International Journal of Approximate Reasoning 187 (2025) 109542

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

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

Diversity among multiple reducts computation with application to ensemble of classification model

Abhimanyu Bar [a,*], Anil Kumar [b], G.Y. Phani Kumar [b], P.S.V.S. Sai Prasad [b,*]

[a] School of Computer Science Engineering and Technology, Bennett University, Greater Noida, 201310, UP, India
[b] School of Computer and Information Sciences, University of Hyderabad, Hyderabad, 500046, Telangana, India

ARTICLE INFO

Keywords:
Feature selection
Rough set theory
Multiple reducts
Discernibility matrix
Ensemble classification
Diversity in ensemble

ABSTRACT

In rough set–based feature selection, the discernibility matrix provides a mathematical framework for computing all or multiple reducts. However, for applications such as ensemble model induction, there is no need to generate an exhaustive set of reducts; instead, selecting a smaller subset with better individual performance and sufficient diversity tends to more effective. Diversity among base classifiers is critical for improving the predictive performance of ensemble models, yet most existing rough set–based methods do not explicitly address this aspect when generating multiple reducts. To overcome this limitation, this paper proposes two strategies that embed diversity directly into the reduct generation process. The first introduces a novel partition refinement cardinality heuristic that selects mutually exclusive reducts with maximum partition cardinality differences to promote classifier diversity. The second presents an efficient adaptation of an existing least overlap heuristic, combined with an incremental construction of the absorbed discernibility matrix to ensure scalability for large datasets where conventional discernibility matrix construction is infeasible. Finally, empirical analysis with state-of-the-art algorithms demonstrates that the diverse reducts generated by the proposed methods successfully achieve their goal of enhancing ensemble model performance through improved diversity and predictive accuracy.

1. Introduction

The Rough Set Theory (RST), introduced by Prof. Zdzisław Pawlak [1], has emerged as a powerful soft-computing framework for dimensionality reduction in Knowledge Discovery in Databases (KDD). Since RST requires no prior assumptions about data distributions and inherently handles vagueness and uncertainty, leading to its widespread adoption in areas such as cost-sensitive feature reduction [2,3], knowledge acquisition [4–7], and approximate reasoning [8,9], etc. A fundamental to RST is the notion of reduct: a minimal subset of attributes that preserves the full discriminative power of the original feature set, ensuring that objects from different decision classes remain distinguishable.

A decision system can produce a large set of reducts—often numbering in the hundreds or thousands—each of which defines a minimal subset of attributes sufficient to induce a valid classification model. In 1992, Skowron et al. [10] introduced the concept of the Discernibility Matrix (DM) for identifying all possible reducts in a given decision system. This foundational idea subsequently

* Corresponding authors.
E-mail addresses: abhimanyu.bar@bennett.edu.in (A. Bar), anilhcu@outlook.com (A. Kumar), phani.stark@gmail.com (G.Y. Phani Kumar), saics@uohyd.ac.in (P.S.V.S. Sai Prasad).

https://doi.org/10.1016/j.ijar.2025.109542
Received 23 May 2025; Received in revised form 23 July 2025; Accepted 31 July 2025

Available online 6 August 2025

enabled a range of developments—including algorithms for computing all/multiple reducts [11,12], approaches to compute optimal and near-optimal reduct [13,14] and many more.

In scenarios where model performance is a primary objective, computing multiple reducts offers a practical and effective alternative to relying solely on a single reduct or attempting to generate all possible reducts. The exhaustive computation of all reducts becomes computationally infeasible as the dimensionality of the dataset increases, significantly limiting its applicability in real-world settings. In contrast, selecting a small number of diverse reducts not only alleviates the computational burden but also retains the representational power necessary for accurate classification. While a single reduct may be sufficient for training an individual classifier, ensemble learning strategies inherently require a collection of strong and diverse base classifiers to achieve optimal performance. Numerous experimental and theoretical studies have consistently demonstrated the potential of ensemble methods to significantly enhance classification accuracy [15–18]. However, the success of an ensemble model is contingent on several key factors. First, inducing an ensemble by aggregating predictions from all possible base classifiers is computationally prohibitive due to its exponential complexity, rendering it unscalable even for moderately sized datasets [19,20]. Second, constructing ensembles with too few or highly similar base classifiers often fails to capitalize on the strengths of ensemble learning [21]. Third, even the presence of highly generalizable individual classifiers may not compensate for the errors made by other base models; rather, it is the complementary nature of classifiers—those offering different perspectives on the data—that drives ensemble effectiveness [22,23]. Thus, promoting diversity—quantified by the degree of disagreement among base classifiers induced from distinct feature subsets—is essential for building robust ensemble models [20,24]. Focusing on the generation of a small set of diverse, high-quality reducts not only addresses computational constraints but also facilitates stronger ensemble performance, highlighting a promising direction in rough set-based model construction.

In RST, each reduct characterizes the decision space from a distinct perspective, preserving the original dataset's discernibility without information loss [1]. Classifiers trained on reducts exhibit generalization capabilities comparable to those built using traditional feature selection techniques, suggesting that ensembles based on reducts can leverage inherent diversity to achieve superior performance. However, not all reducts contribute effectively to ensemble diversity; reducts with similar attribute compositions may induce highly correlated classifiers, diminishing ensemble benefits. Following the "many could be better than all" principle [19], selecting a subset of complementary, diverse reducts offers a more promising path toward constructing robust ensembles. Recent studies in rough set-based ensemble learning [22,25–29] typically generate multiple reducts over different sampled spaces, then inducing base classifiers using them. Although multiple reducts inherently offer some diversity in an ensemble, existing methods rarely establish explicit criteria to ensure sufficient disagreement among them, which is essential for constructing more effective ensemble models. As a result, there remains a significant risk of generating similar or redundant classifiers, thereby limiting the effectiveness of the ensemble. This limitation highlights the need for a more principled approach that explicitly incorporates diversity during the process of multiple reduct computation—before model induction—so that the resulting reducts can lead to more effective ensemble models.

Although there is no universally accepted definition of diversity among base classifiers, this work adopts a subjective perspective by defining diversity as the selection of reducts that are more likely to induce a better-performing ensemble of classification models. To address this objective, authors proposed a method to compute multiple reducts with minimal attribute overlap [11]. The heuristic, known as "$MRGA$ based multiple reducts computation ($MRGA\_MRC$)", explicitly promotes diversity during reduct generation by minimizing common attributes among the selected reducts. While this approach offers a more systematic mechanism for constructing high-performing rough set-based ensembles, its definition of diversity—based solely on minimal overlap—may not be sufficient when generalizability is the primary goal. Therefore, more robust heuristics are needed to identify reducts that not only exhibit minimal overlap but also contribute complementary perspectives, thereby enhancing the ensemble's overall generalization capability.

To enhance both the scalability of algorithms and enforce diversity among multiple reducts, this article introduces the following key contributions:

1. A novel Partition Refinement Cardinality-based heuristic is proposed for selecting diverse multiple reducts, and a corresponding algorithm is developed by integrating this heuristic with an Incremental way of computing Absorbed discernibility Matrix[1] framework.
2. To leverage the advantages of Incremental way of computing Absorbed discernibility Matrix, a scalable version of the existing Minimum Overlap Multiple Reduct Computation ($MRGA\_MRC$) algorithm [11] is reproduced as Least Overlap Multiple Reducts ($LOMR$).

The remainder of this article is organized as follows: Section 2 presents the preliminaries necessary to understand the proposed concepts. Section 3 discusses the base work, as one of the proposed algorithms is a scalable extension of it. Section 4 provides a detailed explanation of the proposed methods, along with illustrative examples. Section 5 reports the results of a comparative experimental study conducted to validate the effectiveness of the proposed approaches. Finally, Section 6 concludes the paper with a summary of the contributions, their impact on feature selection, and potential directions for future research.

[1] This work is part of the author's Ph.D. thesis and introduces a framework that can be seamlessly integrated into any existing method utilizing the Discernibility Matrix (DM) or Absorbed Discernibility Matrix (ADM) [30]. It offers a scalable alternative for datasets where full DM construction is computationally infeasible. Consequently, both the reproduced and newly proposed algorithm in this study are designed using this scalable framework.

Table 1
Decision System (DT).

$Objects$	$Conditional\ Attributes\ (A)$				$Class(D)$
U	a_1	a_2	a_3	a_4	d
o_1	1	0	1	1	0
o_2	1	2	0	1	1
o_3	0	1	0	2	0
o_4	1	1	1	0	1
o_5	1	0	2	2	1
o_6	2	1	2	2	0
o_7	2	2	2	2	1
o_8	0	0	0	1	1

2. Preliminaries

In this section, we present a concise review of the key notations, definitions, and preliminaries that form the foundation of this work, providing clarity for the concepts and methods discussed.

2.1. Decision system

A *Decision System*, often referred to as a *Decision Table (DT)*, is a structured form of data representation used for classification or decision-making tasks. In this representation, each row corresponds to an object (or instance), and each column corresponds to an attribute. It is typically defined as a tuple $DT = (U, A \cup D, \{V_a\}_{a \in A \cup D}, \{f_a\}_{a \in A \cup D})$, where U denotes a non-empty finite set of objects (also called universe of objects), A is a finite set of conditional attributes that describe the objects, and $D = \{d\}$ is the decision attribute representing the outcome or class label. For each attribute $a \in A \cup D$, there exists a function $f_a : U \rightarrow V_a$, which assigns a value from the domain V_a to each object in U. An example of a consistent decision system is provided in Table 1, which is used to demonstrate the concepts throughout the article.

A decision system DT is considered *consistent* if every object in the set U can be uniquely assigned to a decision class from D, using the information provided by the conditional attributes set A. For non-consistent data, consistency is achieved by applying $MDLP$ [31] discretization before further use.

2.2. Granular representation

In rough set theory, knowledge is modeled through granules formed by grouping objects into equivalence classes based on an *indiscernibility relation* [1]. This relation captures the idea that two objects are indistinguishable if they share the same values for a selected set of attributes. Within a decision system DT, the indiscernibility relation defines an equivalence relation over the universe U, where, for a subset of conditional attributes $B \subseteq A$, any two objects o_i and o_j in U are considered indiscernible if they have identical values for all attributes in B. These equivalence classes provide the basis for approximating concepts, especially when dealing with imprecise or uncertain information. Formally, this is expressed as:

$$IND(B) = \{(o_i, o_j) \in U \times U \mid \forall a \in B, \quad a(o_i) = a(o_j)\}. \tag{1}$$

The indiscernibility relation $IND(B)$ partitions the universe of objects U into a set of mutually disjoint subsets, known as equivalence classes [1]. Each equivalence class is referred to as a *granule*. The collection of all such granules forms a *granular space*, which is induced by the attribute subset B [1]. This granular structure is formally represented as $U/IND(B)$ or simply U/B.

2.3. Reducts using discernibility matrix

In RST, multiple reducts are typically derived using a fundamental concept called the *discernibility matrix* [10]. Each entry in this matrix is determined based on a specific relation known as the *discernibility relation* [10].

2.3.1. Discernibility relation

In a decision system, let $B \subseteq A$ be a subset of conditional attributes, and $d \in D$ be the decision attribute. The discernibility relation, denoted as $DIR(B)$, is defined over pairs of objects from the universe U. Two objects o_i and o_j are considered to be in a discernibility relation if they belong to different decision classes (i.e., $d(o_i) \neq d(o_j)$) and differ in at least one attribute from the subset B [10,32]. Formally, the discernibility relation is defined as:

$$DIR(B) = \{(o_i, o_j) \in U \times U \mid \exists a \in B, a(o_i) \neq a(o_j), \quad if \quad d(o_i) \neq d(o_j)\}. \tag{2}$$

2.3.2. Discernibility matrix

The collection of all discernible clauses obtained using Equation (2) over the object space U with respect to the attribute set A is organized into a square matrix of size $U \times U$, which is referred to as the *discernibility matrix* (DM) [10]. In a given decision

Table 2
Discernibility Matrix (DM).

U	1	2	3	4	5	6	7	8
1								
2	$\{a_2, a_3\}$							
3	$\emptyset$	$\{a_1, a_2, a_4\}$						
4	$\{a_2, a_4\}$	$\emptyset$	$\{a_1, a_3, a_4\}$					
5	$\{a_3, a_4\}$	$\emptyset$	$\{a_1, a_2, a_3\}$	$\emptyset$				
6	$\emptyset$	$\{a_1, a_2, a_3, a_4\}$	$\emptyset$	$\{a_1, a_3, a_4\}$	$\{a_1, a_2\}$			
7	$\{a_1, a_2, a_3, a_4\}$	$\emptyset$	$\{a_1, a_2, a_3\}$	$\emptyset$	$\emptyset$	$\{a_2\}$		
8	$\{a_1, a_3\}$	$\emptyset$	$\{a_2, a_4\}$	$\emptyset$	$\emptyset$	$\{a_1, a_2, a_3, a_4\}$	$\emptyset$	

system, the DM is a square matrix, where each entry captures the set of conditional attributes that can distinguish between pairs of objects from different decision classes. For any pair of objects $(o_i, o_j) \in U$, the matrix entry (or clause) $e(o_i, o_j)$ is defined as the set of attributes $a \in A$ for which the objects have different values, provided that their decision values differ. Otherwise, the entry is empty set. Formally, the discernibility matrix is given as:

$$DM = [e(o_i, o_j) \subseteq A]_{i,j \in \{1,2,\ldots,|U|\}}, \tag{3}$$

where each element $e(o_i, o_j)$ is computed as:

$$e(o_i, o_j) = \begin{cases} \{a \in A \mid a(o_i) \neq a(o_j)\}, & \text{if } d(o_i) \neq d(o_j), \\ \emptyset, & \text{otherwise.} \end{cases}$$

Example 2.1. To illustrate the construction and interpretation of the discernibility matrix, a *Decision-Inspired Discernibility Matrix* is developed based on the example decision system given in Table 1 and is presented in Table 2. In the framework of rough set theory, feature selection remains invariant under the permutation of attributes, indicating that reordering the attributes does not influence the resulting reducts. Moreover, the discernibility between any two objects (o_i, o_j) is symmetric, meaning the set of attributes that differentiate o_i from o_j is the same as that for o_j from o_i, formally expressed as $e(o_i, o_j) = e(o_j, o_i)$. Due to this symmetric property, the discernibility matrix exhibits duplicate entries across its main diagonal. To reduce redundancy and enhance readability, only the lower triangular portion of the matrix is filled with discernibility entries, while the upper triangular part is left blank. This approach retains all the essential information required for analysis while streamlining the matrix for clarity and efficiency.

2.3.3. Discernibility function and reduct computation

The *discernibility function*, denoted by $f(DM)$, is a function of Boolean expression formulated to identify minimal subsets of attributes—referred to as *reducts*—from the discernibility matrix. This function is constructed by collecting all non-empty entries $e(o_i, o_j)$ of the DM, where each entry represents the set of attributes that differentiate two objects o_i and o_j belonging to distinct decision classes [10,33]. Formally, the $f(DM)$ is defined as:

$$f(DM) = \wedge\{\vee e(o_i, o_j) | \forall o_i, o_j \in U, e(o_i, o_j) \neq \emptyset\}. \tag{4}$$

By applying Boolean simplification techniques such as distribution, absorption, and the elimination of redundant terms, the function can be minimized to a set of *prime implicants*. The resulting prime implicants form a reduct—that is, a minimal subset of attributes capable of distinguishing all relevant object pairs while preserving the decision-making power of the original attribute set. [10,32–34] provided that each reduct satisfies the conditions specified in Definition 2.1.

Definition 2.1. (Reduct) For a discernibility matrix DM, a subset of attributes $B \subseteq A$ in a given decision system DT is defined as a *reduct* if it satisfies the following two conditions:

1. $\forall (o_i, o_j) \in U \times U | e(o_i, o_j) \neq \emptyset \implies B \cap e(o_i, o_j) \neq \emptyset$. [Jointly Sufficient]
2. $\forall a \in B \quad \exists (o_i, o_j) \in U \times U \ | e(o_i, o_j) \neq \emptyset \wedge ((B - \{a\}) \cap e(o_i, o_j) = \emptyset)$. [Individually Necessary]

3. Base work

This section provides a brief overview of the method proposed in [11], which forms the foundation for the current study. The proposed work builds upon this framework by introducing a more efficient and scalable adaptation of its reduct computation mechanism. The original approach follows the conventional strategy of computing all possible reducts and applies a heuristic to select a subset for constructing an ensemble classification model. It begins with the construction of the Absorbed Discernibility Matrix (ADM), followed by a selection process guided by a least-overlap criterion to identify diverse reducts. For completeness, a scalable reformulation of this algorithm is presented in Section 4.3.2.

Table 3
Absorbed Discernibility Matrix (ADM). (For interpretation of the colors in this table, the reader is referred to the web version of this article.)

U	1	2	3	4	5	6	7	8
1								
2	$\cancel{\{a_2,a_3\}}\xrightarrow{\{a_2\}}$							
3	$\emptyset$	$\cancel{\{a_1,a_2,a_4\}}\xrightarrow{\{a_2\}}$						
4	$\cancel{\{a_2,a_4\}}\xrightarrow{\{a_2\}}$	$\emptyset$	$\cancel{\{a_1,a_3,a_4\}}\xrightarrow{\{a_3,a_4\}}$					
5	$\mathbf{\{a_3,a_4\}}$	$\emptyset$	$\cancel{\{a_1,a_2,a_3\}}\xrightarrow{\{a_2\}}$	$\emptyset$				
6	$\emptyset$	$\cancel{\{a_1,a_2,a_3,a_4\}}\xrightarrow{\{a_2\}}$	$\emptyset$	$\cancel{\{a_1,a_3,a_4\}}\xrightarrow{\{a_3,a_4\}}$	$\cancel{\{a_1,a_2\}}\xrightarrow{\{a_2\}}$			
7	$\cancel{\{a_1,a_2,a_3,a_4\}}\xrightarrow{\{a_2\}}$	$\emptyset$	$\cancel{\{a_1,a_2,a_3\}}\xrightarrow{\{a_2\}}$	$\emptyset$	$\emptyset$	$\mathbf{\{a_2\}}$		
8	$\mathbf{\{a_1,a_3\}}$	$\emptyset$	$\cancel{\{a_2,a_4\}}\xrightarrow{\{a_2\}}$	$\emptyset$	$\emptyset$	$\cancel{\{a_1,a_2,a_3,a_4\}}\xrightarrow{\{a_2\}}$	$\emptyset$	

3.1. *Absorbed discernibility matrix*

In its conventional form, the Discernibility Matrix (DM) comprehensively captures all possible discernibility relations among objects in a dataset. However, this exhaustive representation frequently includes clauses that can be inferred from smaller discernibility sets, which introduces unnecessary computational overhead. Such redundancy highlights the necessity of systematically eliminating superfluous clauses to enhance the overall efficiency and reduce the computational complexity of the reduct generation process. The theoretical foundation for this refinement lies in the absorption law from propositional logic, which asserts that for any two sets P and Q, the expression $P \wedge (P \vee Q) \equiv P$ implies that P alone retains the full distinguishing capacity of the entire expression. Within the framework of Rough Set Theory, a comparable condition applies: when two discernibility sets $e(o'_i, o'_j)$ and $e(o_i, o_j)$ satisfy $e(o'_i, o'_j) \subset e(o_i, o_j)$, the smaller set suffices to discern the relevant object pairs, as indicated by $e(o'_i, o'_j) \cap e(o_i, o_j) = e(o'_i, o'_j)$. Accordingly, $e(o'_i, o'_j)$ absorbs $e(o_i, o_j)$, enabling the larger clause to be replaced with its subset without compromising the generalizability of the matrix. This refinement process is formally referred to as the *absorbed operation* [3,35].

Through the systematic application of the absorbed operation across the DM, any discernibility clause that constitutes a strict superset of another can be effectively eliminated. This iterative simplification yields an optimized matrix composed exclusively of *prime implicants*, which are the minimal and non-redundant discernibility sets necessary for distinguishing objects [32]; this refined structure is referred to as the Absorbed Discernibility Matrix (ADM) [3,32,35]. Formally, for a given discernibility matrix, the ADM is defined as:

$$ADM = \{\, e \in DM \mid \nexists\, e' \in DM,\ e' \neq e, \text{ and } e' \subset e \,\}. \tag{5}$$

Where, each element "e" in ADM represents a discernibility set that is not a proper superset of any other set within the original DM. Therefore, the ADM contains only irreducible clauses that are sufficient to generate all valid reducts without any loss of generality [32,35].

Example 3.1. To illustrate the construction of ADM, Table 3 presents the absorption process applied to the discernibility matrix (DM) provided in Table 2. In this table, the notation $\cancel{\{X,Y\}}\xrightarrow{\{X\}}$ is used to indicate that the clause $\{X\}$ absorbs $\{X,Y\}$. The remaining clauses that are not absorbed during this process are highlighted in bold face and collectively form the final ADM.

The simplification process begins by identifying the smallest discernibility set in the DM, specifically $(o_7, o_6) = \{a_2\}$. This element is found to be a subset of several other discernibility sets, including $\{a_2, a_3\}$, $\{a_1, a_2, a_4\}$, $\{a_2, a_4\}$, $\{a_1, a_2, a_3\}$, $\{a_1, a_2, a_3, a_4\}$, and $\{a_1, a_2\}$. Consequently, these larger sets are absorbed and reduced by $\{a_2\}$ in accordance with the absorption principle. By repeatedly applying this subset-based reduction, the DM is progressively simplified, ultimately retaining only three irreducible clauses: $\{a_1, a_4\}$, $\{a_2\}$, and $\{a_1, a_3\}$. These final discernibility sets, indicated in the table using red boldface, constitute the absorbed discernibility matrix, representing the minimal form sufficient for deriving all valid reducts without any redundant clauses.

3.2. *$MRGA$ based multiple reducts computation ($MRGA\_MRC$)*

The computation of diverse multiple reducts involves two main steps: generating a set of candidate reducts, as outlined in Section 3.2.1, and selecting a diverse subset using the heuristic described in Section 3.2.2.

3.2.1. *Construction of candidate reducts*

Once the ADM is constructed, its clauses are sorted in non-decreasing order of their lengths to facilitate the identification of minimal reducts that maintain a non-trivial intersection with each clause. The resulting ordered list is referred to as the Absorbed Discernibility List (ADL). Formally, the sorted list is represented as:

$$ADL = \{e_i \in ADM \,|\, |e_i| \leq |e_{i+1}|, \forall i = 1, 2, 3, \dots\}. \tag{6}$$

Then, the process of reduct computation begins by initializing the set of multiple reducts to $RED = \emptyset$. Let $B \subseteq RED$ represent the set of reducts that cover the first i clauses $(e_1, e_2, \dots, e_i)$ of ADL. Then each reduct in B maintains a non-trivial intersection with these clauses, hence satisfies reduct properties of Definition 2.1. To extend this coverage to the next clause (e_{i+1}) of ADM, the algorithm considers construction of two subsets, S_{i+1} and T_{i+1}. The set S_{i+1} contains those reducts from B that already have a non-trivial intersection with e_{i+1}, and thus inherently satisfy the reduct condition for the clauses e_1 through e_{i+1}. For the remaining reducts in $B - S_{i+1}$ are extended by augmenting each attribute of e_{i+1} to form a set of possible candidate reducts called T_{i+1} that satisfies reduct properties. The union $RED = S_{i+1} \cup T_{i+1}$ then constitutes the set of reducts that cover clauses from e_1 to e_{i+1}.

3.2.2. Least overlap multiple reduct heuristic

When the number of reducts in RED exceeds n, a selection process is triggered to enforce diversity among the chosen n-reducts. The process begins by selecting the reduct with the smallest cardinality, which is added to the final set of selected reducts, denoted as $SRED$. The remaining $n - 1$ reducts are chosen iteratively using the least-overlap heuristic, denoted as $LOMR_B$. This heuristic is defined as:

$$LOMR_B = \max_{S \in SRED} |B \cap S|. \tag{7}$$

Here, $LOMR_B$[2] represents the maximum overlap of a candidate reduct B with any of the reducts already included in $SRED$. At each step, the reduct with the minimum value of $LOMR_B$ is selected as the next member of $SRED$, thereby ensuring that the selected set exhibits maximal diversity. This process continues iteratively until either the desired number of reducts, denoted as n-reducts, is obtained, or all clauses in the ADL have been processed.

3.2.3. Limitations of DM based reducts computation algorithm

Despite achieving their intended objectives, existing DM based multiple reduct computation algorithms [35–39], including the referenced $MRGA\_MRC$ [11], exhibit notable limitations that hinder their scalability and effectiveness in producing truly diverse reducts.

1. First, a substantial computational burden in these methods arises not from the reduct computation itself, but from the preliminary construction of ADM [11,35] or its variants [36]. This is particularly problematic for those datasets, where the complete construction of the DM—a prerequisite for generating the ADM—can exceed available memory resources. As a result, such approaches face serious scalability issues, limiting their applicability in real-world, high-dimensional decision systems.
2. Second, the evaluation of diversity among multiple reducts has predominantly been based on heuristics that consider attribute overlap or disjointedness [11]. However, this perspective overlooks an important semantic aspect of reducts: their induced partitions. Two reducts that are disjoint in terms of attributes may still generate identical partitions, thereby having the same impact on decision outcomes. Consequently, diversity based purely on syntactic criteria (e.g., attribute overlap) does not guarantee functional or decision-level diversity. A more meaningful diversity assessment should account for differences in the partitions induced by the reducts, reflecting their actual influence on the decision process.

These limitations have motivated our proposed contributions: (1) the development of a simplified and incremental DM construction strategy that avoids full pre-generation without compromising discernibility information, and (2) the introduction of a novel diversity heuristic that quantifies the distance between the partitions induced by different reducts, offering a more principled approach to measuring reduct diversity.

4. Proposed work

This work proposes a novel framework for computing diverse multiple reducts by building upon the foundational structure of the existing $MRGA\_MRC$ algorithm, as discussed in Section 3. While $MRGA\_MRC$ effectively explores the space of least-overlap reducts, it incurs computational overhead during preprocessing and lacks a principled mechanism for ensuring diversity among the selected reducts. To address these limitations, the proposed method introduces two significant advancements. First, it incorporates an incremental approach for computing ADM, which enhances scalability without compromising the quality of discernibility representation. Second, the original least-overlap diversity heuristic is replaced with a novel partition refinement heuristic designed to maximize dissimilarity among reducts, thereby improving the representational richness of the resulting ensemble model. These enhancements collectively contribute to a more effective and computationally efficient approach for generating diverse multiple reducts, making the framework well-suited for decision systems where the construction of the full discernibility matrix is infeasible.

[2] The notation L_B used in [11] has been renamed to $LOMR_B$ for improved clarity and will be consistently used throughout this article.

4.1. Incremental absorbed discernibility matrix ($IADM$)

The **Incremental Absorbed Discernibility Matrix ($IADM$)** is a reduced form of DM, constructed incrementally during the generation of the DM without requiring storage of all its clauses in memory. Let DM^k denote the first k entries generated in the process of constructing the full DM. Then, $IADM^k$ represents the absorbed discernibility matrix derived incrementally from DM^k. To efficiently manage intermediate storage, the clauses in $IADM^k$ are maintained in a priority queue, sorted by clause length. The formal definition of $IADM^k$ is given by Eq. (8):

$$IADM^k = \left\{ e \mid e \in DM^k \wedge \nexists e' \in DM^k \text{ s.t. } e' \neq e \wedge e' \subset e \right\}. \tag{8}$$

It is straightforward that $IADM^1 = DM^1$. Suppose $IADM^k$ has already been computed. To obtain $IADM^{k+1}$, we only need to compare the next clause e_{k+1} from the discernibility matrix with the current $IADM^k$. If e_{k+1} is absorbed by any clause in $IADM^k$, then no update is needed, and $IADM^{k+1} = IADM^k$. Otherwise, we check whether e_{k+1} absorbs any existing clauses in $IADM^k$; if so, those clauses are removed. The updated $IADM^{k+1}$ then consists of the remaining clauses in $IADM^k$ along with e_{k+1}. This procedure is repeated for each clause in the discernibility matrix until the final clause e_n is processed, resulting in $IADM^n$, which serves as the desired ADM. For completeness, the pseudocode for the incremental construction of the Absorbed Discernibility Matrix is presented in Algorithm 1.

Algorithm 1: IADM.

```
Input  : Decision system ($DT$)
Output: $IADM$
$k = 0$; $IADM^0 = \emptyset$;
for Each $o_i \in U$ do
    for Each $o_j \in U$ do
        if $d(o_i) \neq d(o_j)$ then
            $e = \{a | a \in A \wedge a(o_i) \neq a(o_j)\}$;
            if $e \neq \emptyset$ then
                $k = k + 1$; $isAbsorbed = False$ ;
                for Each $e' \in IADM^{k-1}$ do
                    if $e' \subseteq e$ then
                        $IADM^k = IADM^{k-1}$;
                        $isAbsorbed = Ture$; $break$;
                    else
                        if $e \subseteq e'$ then
                            $IADM^{k-1} = IADM^{k-1} - \{e'\}$;
                        end
                    end
                end
                if $isAbsorbed == False$ then
                    $IADM^k = IADM^{k-1} \cup \{e\}$;
                end
            end
        end
    end
end
$IADM = IADM^k$;
$Return$  $IADM$;
```

Example 4.1. To illustrate the incremental construction process of the discernibility matrix, we consider the decision system provided in Table 1. Using the established formula for generating discernibility clauses, we derive a set $\{e_1, e_2, \ldots, e_n\}$, where each e_k corresponds to the k^{th} clause in the discernibility matrix. For a system comprising eight objects, the maximum possible number of such clauses is $\frac{8\times(8-1)}{2} = 28$. The resulting discernibility matrix based on this dataset is displayed in Table 2 for reference. To enhance understanding of the incremental procedure, Table 4 details the absorption steps involved. Here, the clause index k, the corresponding clause e_k, and the updated incremental discernibility matrix at step k are shown. Notably, the notation $\overset{\{a_2,a_4\}}{\{\cancel{a_1, a_2, a_4}\}}$ indicates that the clause $\{a_2, a_4\}$ absorbs $\{a_1, a_2, a_4\}$, leading to the removal of the clause from the queue during the absorption phase.

The process begins by computing the first clause of the discernibility matrix, $e_1 = \{a_2, a_3\}$, using the defined equation. Since no prior clauses exist, this clause initializes the incremental absorbed discernibility matrix as $IADM^1 = \{\{a_2, a_3\}\}$. For each subsequent non-empty clause, an absorption check is performed against the existing $IADM$ clauses, as described previously. The second clause, $e_2 = \emptyset$, is empty and therefore ignored. The third clause, $e_3 = \{a_1, a_2, a_4\}$, is evaluated to determine if it absorbs or is absorbed by any clause in $IADM^2$. Since $\{a_1, a_2, a_4\} \nsubseteq \{a_2, a_3\}$, no absorption occurs, and e_3 is added to form $IADM^3 = \{\{a_2, a_3\}, \{a_1, a_2, a_4\}\}$. When processing the fourth clause $e_4 = \{a_2, a_4\}$, it is found that $\{a_2, a_4\} \subset \{a_1, a_2, a_4\}$, meaning e_4 absorbs $\{a_1, a_2, a_4\}$, which is then removed from the matrix. This iterative absorption continues through all 28 clauses, culminating in the final incremental absorbed discernibility matrix, $IADM^{28} = \{\{a_2\}, \{a_3, a_4\}, \{a_1, a_3\}\}$, which encapsulates the essential distinguishing attributes for the dataset.

Table 4
Illustration of $IADM$ operations for Table 1.

k	e_k	$IADM^k$
1	$\{a_2, a_3\}$	$\{\{a_2, a_3\}\}$
2	$\emptyset$	$\{\{a_2, a_3\}\}$
3	$\{a_1, a_2, a_4\}$	$\{\{a_2, a_3\}, \{a_1, a_2, a_4\}\}$
4	$\{a_2, a_4\}$	$\{\{a_2, a_3\}, \{a_2, a_4\}, \cancelto{\{a_2,a_4\}}{\{a_1, a_2, a_4\}}\}$
5	$\emptyset$	$\{\{a_2, a_3\}, \{a_2, a_4\}\}$
6	$\{a_1, a_3, a_4\}$	$\{\{a_2, a_3\}, \{a_2, a_4\}, \{a_1, a_3, a_4\}\}$
$\vdots$	$\vdots$	$\vdots$
28	$\{a_1, a_2, a_3, a_4\}$	$\{\{a_2\}, \{a_1, a_3\}, \{a_3, a_4\}, \cancelto{\{a_3,a_4\}}{\{a_1, a_2, a_3, a_4\}}\}$

Bit-representation of $IADM$: Usually the $IADM$ is organized as a list of clauses, where each clause represents a subset of attributes that differentiates a specific object pair. However, this format can be streamlined by utilizing certain characteristics of the input and the algorithm. First, the algorithm is designed to be invariant to both duplicate entries and how often they occur, making repetition irrelevant. Second, since each clause must intersect with the reduct through at least one shared attribute—regardless of the originating object pair—the explicit storage of object indices is unnecessary. These indices merely reflect which objects are distinguished by the clause's attribute set and are not essential for the reduct computation.

Considering these characteristics, the $IADM$ can be more effectively modeled as a collection of distinct attribute subsets rather than as a conventional representation. To enhance implementation flexibility and reduce memory overhead, each clause can be encoded using a bit-wise representation. Specifically, each attribute subset is transformed into an N-bit binary vector, where N denotes the total number of conditional attributes in the set A. In this representation, the k^{th} bit is set to 1 if the attribute "k" discerns an object pair in the corresponding clause, and to 0 otherwise. For example, let the conditional attribute set be $A = \{a_1, a_2, a_3, a_4, a_5\}$ for the given decision system over which $IADM$ is constructed. If a clause within the $IADM$ contains the subset $\{a_2, a_4, a_5\}$, it can be encoded as a binary-array $\{0, 1, 0, 1, 1\}$, where each bit position corresponds to the presence (1) or absence (0) of an attribute in A. Using this approach, the entire $IADM$ can be efficiently represented as a collection of binary-array, eliminating the need for explicit indexing and providing a compact alternative to the traditional matrix-based format. For example, if the $IADM$ is given, then it can be represented as bit-array as follows:

$$\begin{aligned} IADM &= \{\{a_1, a_3\}, \{a_1, a_5\}, \{a_2, a_3, a_5\}, \{a_1, a_2, a_4\}\} \\ &= \{\{1, 0, 1, 0, 0\}, \{1, 0, 0, 0, 1\}, \{0, 1, 1, 0, 1\}, \{1, 1, 0, 1, 0\}\} \end{aligned}$$

Here, each entry is encoded as a binary vector of length $N = |A| = 5$, where each bit is set to 1 if the corresponding attribute is present in the subset, and 0 otherwise.

4.2. Diversity heuristic in rough set theory

In the construction of rough set–based ensemble models, selecting multiple reducts that are not only minimal but also diverse is critical for maximizing ensemble accuracy and robustness. Existing strategies for enforcing diversity among multiple reducts have predominantly relied on measuring attribute-level overlap. For instance, the Least Overlap Multiple Reduct ($LOMR$) heuristic selects reducts iteratively by minimizing their maximum overlap with already chosen reducts, thereby favoring attribute sets that share as few common attributes as possible [11]. Similarly, the widely adopted Jaccard similarity–based heuristic (DR) computes diversity as the complement of the ratio of the intersection to the union of two reducts' attribute sets, assuming that lower intersection implies higher functional difference [29].

While these heuristics successfully ensure syntactic diversity, they overlook a crucial semantic dimension: the partitions that the reducts induce. In rough set theory, the true discernibility power of a reduct is defined by the partitions it generates (granules)—not merely by the attribute set itself. As a result, two reducts that share no attributes may still induce identical partitions, providing no genuine diversity in decision-making [40]. Therefore, diversity measures based solely on overlap can lead to hidden redundancy. These limitations underline the necessity for a novel diversity heuristic that extends beyond set-theoretic overlap and directly quantifies the difference between the induced partitions of candidate reducts. By evaluating functional distinctions at the granular level, such a partition-based heuristic would provide a more principled and theory-consistent foundation for selecting truly diverse reducts in rough set–based ensemble models.

4.2.1. Partition refinement cardinality (PRC)

The partition refinement cardinality measures is the change in the number of equivalence classes induced over the universe U when a set of attributes $Y \subseteq A \cup D$ is added to an existing attribute set $X \subseteq A \cup D$. Formally, it can be derived as:

$$P_r(X, Y) = |U/IND(X \cup Y)| - |U/IND(X)|\,. \tag{9}$$

Here, $IND(Z)$ denotes the indiscernibility relation induced by the attribute set Z, and $U/IND(Z)$ represents the corresponding partition of the universe over Z. The $P_r(X, Y)$ measure captures the number of new splits that occur in the partition defined by X

when the attributes in Y are added. When the partition created by Y closely resembles that of X, the value of $P_r(X,Y)$ tends to be small. On the other hand, if the partitions produced by X and Y differ substantially, the refinement value increases accordingly. Therefore, $P_r(Y,X)$ quantifies how many additional splits arise in the partition induced by Y when attributes from X are included. Since the sizes of the partitions $U/IND(X)$ and $U/IND(Y)$ can vary, $P_r(X,Y)$ does not necessarily equal $P_r(Y,X)$.

The value $P_r(X,Y)$ serves as a measure of how much the partition formed by the attribute set X is refined when attributes from Y are incorporated. This refinement captures the structural difference between the partitions $U/IND(X)$ and $U/IND(Y)$, providing a way to quantify their diversity. When $P_r(X,Y) \neq P_r(Y,X)$, it reflects an asymmetry in the refinement process, indicating a significant difference between the two partitions. This insight leads to the development of a heuristic called *Partition Refinement Cardinality* (PRC), which is used to evaluate the diversity between reducts. Given a candidate reduct $B \in$ AllReduct and a set of already selected diverse reducts $R \in$ SRED, the PRC score for B is defined as follows:

$$PRC_B = \min_{R\in \text{SRED}} \left\{ \min\left(P_r(R,B), P_r(B,R)\right)\right\}. \tag{10}$$

Here, the "inner-min" evaluates the lower of the two refinement values $P_r(R,B)$ and $P_r(B,R)$, offering a balanced assessment of the partition difference. The "outer-min" identifies the closest similarity between B and any reduct in the set $SRED$. If B is found to be similar to even one of the existing reducts, it is considered insufficiently diverse and therefore not suitable for inclusion. Finally, the reduct B^* that maximizes this diversity score is selected:

$$PRC_{B^*} = \max_{B\in RED} PRC_B. \tag{11}$$

This B^* is then added to the set $SRED$ as it represents the candidate with the highest diversity relative to the current collection.

4.2.2. Theoretical distinction among existing and proposed diversity heuristics

Compared to overlap-based heuristics such as $LOMR$ [11] and DR [29], PRC offers the following benefits:

1. **Semantic Relevance:** PRC reflects the true decision-making impact of an attribute set by examining how it segments the data (that is granular representation).
2. **Redundancy Elimination:** It prevents misleading conclusions about diversity when attribute sets have zero overlap but identical induced partitions.
3. **Data-Driven Adaptability:** PRC adapts to the dataset's underlying structure, producing more robust and genuinely diverse multiple reducts for ensemble models.

In summary, the PRC heuristic addresses a critical gap in classical difference measures by redefining diversity in a way that is theoretically sound and practically meaningful. Its alignment with the rough set philosophy makes it a significant advancement for constructing scalable, diverse, and effective reduct-based models.

4.3. Diverse multiple reducts computation

This work introduces two algorithms for computing diverse multiple reducts using the concept of $IADM$ discussed in Section 4.1. The first algorithm, employing the proposed PRC heuristic discussed in Section 4.2.1, is named as Partition Refinement Cardinality based Multiple Reducts ($PRCMR$). The second algorithm, reproduced as Least overlap Multiple Reducts ($LOMR$), is a scalable adaptation of the $MRGA\_MRC$ method originally proposed in [11]. The $LOMR$ algorithm is reproduced to demonstrate the applicability of $IADM$ over conventional ADM-based techniques and also serves as a state-of-the-art benchmark for performance comparison.

4.3.1. Partition refinement cardinality based multiple reducts: $PRCMR$

To promote diversity among the computed reducts, a Partition Refinement Cardinality-based multiple reduct computation algorithm, termed $PRCMR$, is proposed. This algorithm integrates the $IADM$ with the PRC heuristic to effectively guide the selection of diverse reducts. The complete procedure is presented in the pseudocode given in Algorithm 2.

The algorithm operates through a structured sequence of steps to ensure the efficient generation of diverse reducts. It begins with the construction of the $IADM$ using the approach detailed in Section 4.1 [***Ref. Line No.: 1***]. From the resulting non-empty clauses, a pool of candidate reducts, denoted as RED, is formed, as outlined earlier in this section [***Ref. Line No.: 4–7***]. If the number of candidates in RED exceeds the user-specified threshold "n", the algorithm first selects the reduct with the smallest cardinality as the initial member of the diverse reduct set $SRED$. To maximize diversity, the remaining $n-1$ reducts are then identified using the PRC heuristic described in Section 4.2.1 [***Ref. Line No.: 8–26***]. This selection process continues until the desired number of reducts is obtained or all clauses have been processed. The algorithm then outputs the final set of diverse reducts [***Ref. Line No.: 27***].

4.3.2. Least overlap based multiple reducts: $LOMR$

To address the limitations of the original $MRGA\_MRC$ algorithm [11]—particularly its inefficient memory usage during ADM construction, as discussed in Section 3.2—this work presents a scalable improvement called the "Least Overlap Multiple Reducts ($LOMR$)" algorithm. The proposed algorithm incorporates the $IADM$ method, detailed in Section 4.1, to replace the traditional matrix construction used in the original approach. By leveraging $IADM$, $LOMR$ significantly enhances space efficiency and reduces

Algorithm 2: $PRCMR$.

```
   Input  : Decision system ($DT$), $n$: Number of reducts required to obtain.
   Output: $RED$: $n$-diverse reducts
   /* Find $IADM$ using Algorithm 1                                  */
1  $IADM = IADM(DT)$;
   /* Computation of Candidate Reducts                               */
2  $RED = \emptyset$;
3  for Each $e_i$ in $IADM$ do
4      $S_i = \{B \in RED | B \cap e_i \neq \emptyset\}$;
5      $T_i = \bigcup_{a\in e_i} \bigcup_{B\in RED \wedge B\cap e_i=\emptyset}\{B \cup \{a\}\}$;
6      $MIN_i = \{B \in T_i | min(B, IADM, i) = TRUE\}$;
7      $RED = S_i \cup MIN_i$;
       /* Multiple Reducts Selection                                 */
8      $SRED = \emptyset$;
9      if $|RED| > n$ then
10         Sort reducts of $RED$ based on induce partition size;
11         $SRED = SRED \cup RED(1)$;
12         for Each $j$ in $2$ to $n$ do
13             $BestCount = 0$; $CurrentBest = \emptyset$;
14             for Every $B$ in $RED - SRED$ do
15                 $PRCMR_B = \min_{R\in SRED}(min(P_r(B,R), P_r(R,B)))$;
16                 if $PRCMR_B > BestCount$ then
17                     $BestCount = PRCMR_B$;
18                     $CurrentBest = B$;
19                 end
20             end
21             $SRED = SRED \cup \{B\}$;
22             $RED = RED - \{CurrentBest\}$;
23         end
24         $RED = SRED$;
25     end
26 end
27 Return $RED$;
```

computation time. Since $LOMR$ follows the structural design of the original $MRGA\_MRC$ algorithm, only the initial matrix construction phase is redesigned, while the subsequent phases are kept intact with necessary modifications to ensure compatibility with the new approach and the pseudocode given in Algorithm 3.

The execution of this algorithm begins with the construction of the $IADM$ following the procedure described in Section 4.1 [***Ref. Line No.: 1***]. Based on the resulting non-empty clauses, a set of candidate reducts, denoted as RED, is generated according to the criteria specified in Definition 2.1 [***Ref. Line No.: 4–7***]. If the number of candidate reducts exceeds the user-defined target n, the algorithm first selects the reduct with the smallest cardinality as the initial member of the final reduct set, $SRED$. The remaining $n-1$ reducts are then determined using the Least Overlap heuristic, $LOMR_B$, defined in Eq. (7) [***Ref. Line No.: 8–27***]. This process iterates until the required n reducts have been identified or all clauses within the $IADM$ have been examined, after which the algorithm returns the resulting set of diverse multiple reducts [***Ref. Line No.: 28***].

4.4. Computational complexity

In discernibility matrix-based reduct computation approaches, the primary computational overhead arises during the construction of the discernibility matrix itself, rather than from the heuristic or search strategies applied afterward. Therefore, the overall computational complexity of such algorithms is predominantly determined by the complexity of constructing the discernibility matrix [32,34,41]. Here, in the worst-case scenario, the absorption mechanism used during $IADM$ construction may fail to eliminate any clauses, leading to $O(|U|^2)$ entries being retained. Each comparison or absorption attempt involves a cost of $O(|U|^2 \cdot |A|)$, resulting in a total time complexity of $O(|U|^4 \cdot |A|)$. However, in the best case, if every new clause absorbs all previous ones, only a single clause remains at each step, reducing the complexity to $O(|U|^2 \cdot |A|)$.

Both the $LOMR$ and $PRCMR$ algorithms fundamentally depend on the construction of the Incremental Absorbed Discernibility Matrix ($IADM$). According to the findings in [42], the worst-case space complexity for constructing the $IADM$ is $O(|U|^2|A|)$, while the corresponding time complexity reaches $O(|U|^4 \cdot |A|)$. After the $IADM$ is generated, deriving all valid reducts—essentially the prime implicants—becomes an NP-hard task with a complexity of $O(2^{|A|})$ as shown in [10]. Consequently, the total time complexity for the proposed algorithms falls between $O(|U|^4 \cdot |A|)$ and $O(2^{|A|})$.

Although a precise or best-case time complexity expression for $LOMR$ and $PRCMR$ has not yet been formalized, it remains a subject for future research. However, an analytical comparison of their components suggests that $PRCMR$ is likely to require more computation time than $LOMR$, due to the inclusion of partition refinement and set intersection operations. Partition refinement adds a cost of $O(|A||U|\log|U|)$, while set intersection contributes a cost of $O(|A|)$. Since it is challenging to analytically quantify the effect of the optimization techniques integrated into the proposed algorithm, a practical evaluation has been carried out instead. This evaluation is designed to systematically assess how the introduced strategies influence the algorithm's performance, particularly

Algorithm 3: $LOMR$.

```
Input  : $DT$: Decision system, $n$: Number of reducts required to obtain.
Output: $RED$: $n$-diverse reducts
/* Find $IADM$ using Algorithm 1                                        */
$IADM = IADM(DT)$;
/* Computation of Candidate Reducts                                     */
$RED = \emptyset$;
for Each $e_i$ in $IADM$ do
    $S_i = \{B \in RED | B \cap e_i \neq \emptyset\}$;
    $T_i = \bigcup_{a \in e_i} \bigcup_{B \in RED \wedge B \cap e_i = \emptyset} \{B \cup \{a\}\}$;
    $MIN_i = \{B \in T_i | min(B, IADM, i) = TRUE\}$;
    $RED = S_i \cup MIN_i$;
    /* Multiple Reducts Selection                                       */
    $SRED = \emptyset$ ;
    if $|RED| > n$ then
        Sort $RED$ by increasing order of reduct size;
        $SRED = SRED \cup RED(1)$;
        for Every $j$ in 2 to $n$ do
            $BestCount = |\mathbb{A}|$;
            $CurrentBest = \emptyset$;
            for Every $B$ in $RED - SRED$ do
                $LOMR_B = \max_{S \in SRED} |B \cap S|$;
                if $LOMR_B < BestCount$ then
                    $BestCount = LOMR_B$;
                    $CurrentBest = B$;
                end
            end
            $SRED = SRED \cup \{CurrentBest\}$;
            $RED = RED - \{CurrentBest\}$;
        end
        $RED = SRED$;
    end
end
Return $RED$;
```

in terms of memory usage and execution time. By focusing on experimental results, the analysis offers concrete evidence of the improvements achieved in space and time efficiency, which are otherwise difficult to establish through purely theoretical means.

5. Experimental evaluation

To verify the effectiveness of the proposed methods, this section presents a series of experimental studies comparing them with existing state-of-the-art algorithms. The experiments are organized into four focused analyses, each designed to highlight the limitations of existing methods and the advantages introduced by the proposed solutions.

1. Section 5.2 demonstrates the computational efficiency of $IADM$ relative to the traditional absorbed discernibility matrix.
2. Section 5.3 investigates the scalability of $IADM$ relative to ADM, particularly when applied to large-scale datasets.
3. Section 5.4 evaluates the performance of the proposed multiple reduct algorithms in terms of runtime and memory usage, demonstrating their practical scalability.
4. Section 5.5 examines the effectiveness of the reducts generated by the proposed methods in supporting classification tasks within ensemble learning frameworks.

Collectively, these experiments demonstrate the practical utility and performance benefits of the proposed approaches.

5.1. *Evaluation framework and setup*

The experimental evaluation was carried out on a desktop machine running Ubuntu 18.04.5 LTS (64-bit), featuring a 12-core Intel® Core™ i7-8700 CPU operating at 3.20 GHz, and equipped with 32 GB of RAM. All algorithms, the proposed as well as state-of-the-art algorithms, were implemented in the MATLAB environment. To conduct a comprehensive performance assessment, a total of Ten benchmark datasets were employed. Five datasets—Austra, Car, Ring, Satimage and KddCup—were sourced from the KEEL Data-Mining repository [43]. In addition, four datasets—Dna, Wdbc, Wine and OptDigit—were obtained from the UCI Machine Learning Repository [44], while the Hawks dataset was collected from the R-Data Sets archive hosted on GitHub [45]. Since classical Rough Set Theory is designed to handle categorical or discretized data, all datasets used in the study were either inherently categorical or converted using the "MDLP"' discretization method [31]. Detailed specifications of the datasets are presented in Table 5.

A comprehensive comparative experimental study was conducted to evaluate the effectiveness of the proposed $PRCMR$ and $LOMR$ algorithms in comparison with several established state-of-the-art algorithms, including $MRGA\_MRC$ ($MRGA$-based Multiple Reducts Computation) [11], DR (Select Diverse Reducts) [29], $BBFS$ (Blind Breadth First Search) [38], and $FSPA$ (Feature

Table 5
Description of benchmark data sets.

Sl. No.	Dataset	Objects	Attributes	Classes
1	Austra	690	15	2
2	Dna	318	16	8
3	Hawks	908	19	5
4	Wine	178	14	3
5	Wdbc	569	31	2
6	Car	4655	70	5
7	Ring	4000	21	2
8	Satimage	4000	37	6
9	KddCup	494021	40	27
10	OptDigit	56200	65	6

Table 6
Computational gain of $IADM$ over Traditional ADM.

Dataset	DM_{size}	$IADM$			ADM			Percent of Gain		
		AO_{IADM}	$IADM_{size}$	$T(sec)_{IADM}$	AO_{ADM}	ADM_{size}	$T(sec)_{ADM}$	$AoG\%$	$DMsR\%$	$CtG\%$
Austra	117581	27715	23	0.46	118759	23	2.19	76.66	99.98	79.11
Dna	39990	96713	161	0.27	124166	161	0.43	22.11	99.60	36.78
Hawks	137643	24707	168	0.37	220309	168	0.51	88.79	99.88	28.39
Wine	10429	2680	38	0.05	14273	38	0.07	81.22	99.64	31.88
Wdbc	75684	320514	649	0.54	1015738	649	1.44	68.45	99.14	62.52
Car	4526366	239789794	6087	663.67	1112876823	6087	1454.24	78.45	99.87	54.36
Ring	3999039	62278657	4792	82.78	61754051	4792	8680.48	−0.85	99.88	99.05
Satimage	6486253	191383530	16459	217.63	297971865	16459	33285.70	35.77	99.75	99.35

Selection based on the Positive Approximation) [46]. In this study, $MRGA\_MRC$ is treated as the base algorithm and serves as the primary representative of ADM-based approaches. The DR algorithm is included as a benchmark for rough set reduct–based ensemble learning. $BBFS$ represents a straightforward DM-based approach for computing minimal or shortest-length reducts, thereby highlighting the impact of minimal reduct–driven ensembles. In contrast, $FSPA$ is a greedy single reduct algorithm, incorporated to emphasize the comparative performance differences between single reduct methods and multiple reduct–based ensemble strategies. To ensure an unbiased evaluation and maintain consistency across all ensemble classification models, the number of reducts used to construct base classifiers was uniformly set to 15 for all algorithms, except $FSPA$.

5.2. *Evaluating the computational advantage of $IADM$ over traditional ADM*

A key part of this work is induction of *IADM*, which improves the scalability over the conventional *ADM*. This section presents a detailed experimental comparison to highlight the computational advantages of using *IADM* in reducts computation. Table 6 summarizes the evaluation results. The column labeled DM_{size} shows the total number of discernibility matrix entries generated for each dataset. For both methods, AO_{IADM} and AO_{ADM} denote the number of absorption operations performed during matrix construction. The sizes of the resulting matrices used in reduct computation are recorded as $IADM_{size}$ and ADM_{size}, respectively. The total time taken to complete the computation is reported under $T(sec)_{IADM}$ and $T(sec)_{ADM}$. Finally, the table presents three key performance metrics that together demonstrate the achieved efficiency improvements: Absorption Operation Gain (AoG), Discernibility Matrix Size Reduction ($DMsR$), and Computational Time Gain (CtG), which are calculated using Equations (12), (13), and (14), respectively.

$$AoG = \frac{AO_{DM} - AO_{IADM}}{AO_{DM}} \times 100. \tag{12}$$

$$DMsR = \frac{DM_{size} - IADM_{size}}{DM_{size}} \times 100. \tag{13}$$

$$CtG = \frac{T(sec)_{ADM} - T(sec)_{IADM}}{T(sec)_{ADM}} \times 100. \tag{14}$$

Analysis of Results: Table 6 clearly shows that the *IADM* approach achieves a notable gain in reducing absorption operations, with values ranging from −0.85% to 88% across the benchmark datasets. This performance improvement can be attributed to two main factors. First, *IADM* processes clauses in ascending order of their lengths, allowing newly encountered clauses to be more readily absorbed by existing, shorter ones. This strategic ordering significantly reduces redundant absorption operations during matrix construction. In contrast, the traditional *ADM* applies the absorption principle only after building the full discernibility matrix. Due to the larger number of clauses, as indicated by the DM_{size} metric, this leads to substantially more operations and higher computational overhead.

Moreover, the reduced frequency of absorption operations in *IADM* directly improves execution time. The computational time gains (CtG) observed, ranging from 28% to 99%, align closely with the significant reduction in discernibility matrix size ($DMsR$),

Table 7
Scalability of $IADM$ over ADM.

Dataset	Objects (Sample)	Attributes	DM_{size}	$IADM$			ADM		
				AO_{IADM}	$IADM_{size}$	T_{sec}	AO_{ADM}	ADM_{size}	T_{sec}
KddCup	1000	40	284635	805662	56	1.85	1278220	56	43.26
KddCup	2000	40	1181382	2576780	98	5.43	7537032	98	1305.61
KddCup	3000	40	2640112	10622454	105	15.52	16187133	105	7770.32
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
KddCup	55000	40	$3.4339e+10$	11061433	66	2305.84	—	—	—
OptDigit	500	65	112119	365603029	32081	379.68	1077233294	32081	1380.49
OptDigit	1000	65	449776	2474852238	84143	2566.88	7430826783	84143	9497.10

which exceeds 99% for all datasets. This demonstrates that the compactness of the matrix is a key factor driving the time efficiency of the *IADM* method. In contrast, the traditional *ADM* approach requires full matrix construction and extensive absorption operations on larger clause sets, adding substantial time complexity. Consequently, *ADM* is less scalable and less practical for large-scale or time-sensitive reduct computations.

In summary, the proposed *IADM* method offers superior computational performance in both time and space, making it a more practical and efficient alternative for discernibility matrix–based reduct computation, especially in tasks that require generating multiple, optimal, or near-optimal reducts.

5.3. Scalability assessment of $IADM$ versus traditional ADM

To evaluate the scalability of *IADM*, a comparative experiment was conducted against the conventional *ADM*. The “KDDCup” and “OptDigit” datasets were selected for their large size and suitability for benchmarking large-scale data processing methods. The results, summarized in Table 7, provide clear insights into the computational performance of both approaches as the dataset size increases.

In this study, different samples from both datasets, each containing varying numbers of instances, were used as indicated in the “Objects (Sample)” column of the table. The remaining table details, including DM_{size}, $IADM_{size}$, AO_{ADM}, AO_{IADM}, and T_{sec}, follow the same format as in Table 6. Since the results illustrate performance for increasingly larger sample sizes, the ellipsis symbol (⋮) denotes the continuation of experiments with larger samples, while a double dash (−−) indicates that the *ADM* method failed to complete execution due to memory limitations.

Analysis of Results: The results in Table 7 show that for *KDDCup* samples of 1000, 2000, and 3000 instances, *IADM* requires significantly fewer absorption operations than *ADM* to generate the final matrix. This reduction leads directly to improved computational efficiency: *IADM* completes these samples in 1.85 to 15.52 seconds, while *ADM* takes substantially longer, from 43.26 to 7770.32 seconds for the same tasks. Due to space constraints, results for all tested sizes are not shown here, but similar trends were consistently observed across additional runs, confirming the robustness of the findings. Notably, for a larger sample of 55000 instances, *ADM* failed due to memory demands of about 3.4339×10^{10} units needed to store the full discernibility matrix. In contrast, *IADM* processed the same sample successfully in 2305.84 seconds, performing 11061433 absorption operations and producing only 66 final clauses. Similar scalability benefits were observed for the *OptDigit* dataset as well, demonstrating that *IADM* is both practical and efficient for large-scale and high-dimensional data.

In conclusion, the empirical evidence suggests that *IADM* is capable of handling datasets that exceed the memory constraints of traditional *ADM*. Since *IADM* requires only the final set of simplified clauses to be held in memory, while *ADM* necessitates storing the entire discernibility matrix, the proposed method significantly broadens the scope of applicability for reduct computation. The frequent observation that $|IADM| \ll |DM|$ further reinforces this conclusion. Consequently, algorithms that rely on *ADM* or the discernibility matrix as a computational foundation can benefit from enhanced scalability and efficiency by adopting *IADM*, thereby enabling their use in larger and more complex data environments.

5.4. Comparative evaluation of computational performance

To evaluate the computational efficiency of the proposed multiple reduct computation algorithms $PRCMR$ and $LOMR$ in comparison with other methods, the experimental results are summarized in Table 8. For each algorithm, the table reports two key performance metrics: the average length of the computed reducts ($15 - reducts$), denoted as $RL(\mu \pm \sigma)$, and the total execution time in seconds (T_{sec}). Since $FSPA$ generates only a single reduct, its reduct length is reported without a standard deviation. Additionally, for some datasets, certain algorithms were unable to complete execution due to resource constraints such as excessive memory usage or prolonged runtime. These cases are indicated by “—” in the corresponding table entries, signifying that the method did not scale for those datasets.

Analysis of Results: An analysis of the computational time reported in Table 8 reveals several important trends. As dataset size increases, the time required by the ADM-based $MRGA\_MRC$ algorithm to compute reducts grows significantly, eventually surpassing the execution time of the proposed $IADM$-based approaches. When compared to the exhaustive DM-based DR and *BBFS* algorithms, the proposed methods demonstrate better efficiency on smaller datasets. However, on moderately larger datasets such as *Wdbc*, where the complexity of the search space increases, DR and *BBFS* incurs a considerably higher computational cost than

Table 8
Computational time gain of the proposed multiple reducts approaches over state-of-the-art algorithms.

Dataset	$PRCMR$		$LOMR$		$MRGA\_MRC$		DR		$BBFS$		$FSPA$	
	$RL(\mu \pm \sigma)$	T_{sec}	$RL(\mu \pm \sigma)$	T_{sec}	$RL(\mu \pm \sigma)$	T_{sec}	$RL(\mu \pm \sigma)$	T_{sec}	$RL(\mu \pm \sigma)$	T_{sec}	RL	T_{sec}
Austra	5.13 ± 1.30	4.03	3.73 ± 0.59	0.69	3.67 ± 0.49	2.35	4.40 ± 1.12	3.49	3.67 ± 0.49	2.34	3	0.28
Dna	6.29 ± 0.47	5.40	6.13 ± 0.35	1.92	6.07 ± 0.27	1.90	4.93 ± 1.22	16.66	5.67 ± 0.49	0.66	6	0.31
Hawks	5.60 ± 0.91	29.47	4.40 ± 1.06	1.83	4.40 ± 0.99	1.97	4.93 ± 1.22	16.93	3.87 ± 0.35	1.88	3	0.32
Wine	4.73 ± 1.03	1.58	3.13 ± 0.35	0.76	3.13 ± 0.35	0.75	4.13 ± 1.06	1.27	3.00 ± 0.00	0.09	3	0.07
Wdbc	9.53 ± 1.25	11.81	9.07 ± 0.70	4.13	8.53 ± 0.74	4.48	8.53 ± 1.13	2997.43	7.00 ± 0.00	1147.36	7	0.23
Car	6.57 ± 1.65	1677.19	6.08 ± 2.33	747.03	6.00 ± 2.39	1471.25	—	—	—	—	1	0.37
Ring	11.53 ± 0.64	628.60	11.20 ± 0.41	85.28	11.20 ± 0.41	8591.51	—	—	—	—	10	2.14
Satimage	8.73 ± 0.46	1175.34	7.53 ± 0.52	241.45	7.93 ± 0.46	33616.52	—	—	—	—	7	3.71

Table 9
Compared results of the Naive Bayes (NB) classification models.

Dataset	$PRCMR$	$MRGA\_MRC$		DR		$BBFS$		$FSPA$	
	$\mu \pm \sigma$	$\mu \pm \sigma$	p_{val}	$\mu \pm \sigma$	p_{val}	$\mu \pm \sigma$	p_{val}	$\mu \pm \sigma$	p_{val}
Austra	74.35 ± 4.32	63.91 ± 4.35	$\mathbf{0.00^{+}}$	65.94 ± 8.37	$\mathbf{0.00^{+}}$	64.20 ± 4.69	$\mathbf{0.00^{+}}$	58.99 ± 4.53	$\mathbf{0.00^{+}}$
Dna	85.78 + 2.71	80.02 + 5.12	$\mathbf{0.00^{+}}$	79.68 + 5.10	$\mathbf{0.00^{+}}$	79.47 + 4.98	$\mathbf{0.00^{+}}$	79.95 + 2.49	$\mathbf{0.00^{+}}$
Hawks	85.21 ± 3.91	79.85 ± 4.58	$\mathbf{0.01^{+}}$	79.54 ± 3.94	$\mathbf{0.00^{+}}$	79.62 ± 4.31	$\mathbf{0.01^{+}}$	80.93 ± 4.72	$\mathbf{0.04^{+}}$
Wine	88.40 ± 9.90	66.56 ± 12.57	$\mathbf{0.00^{+}}$	72.14 ± 19.95	$\mathbf{0.03^{+}}$	65.79 ± 10.44	$\mathbf{0.00^{+}}$	60.85 ± 12.65	$\mathbf{0.00^{+}}$
Wdbc	94.69 ± 1.94	94.38 ± 1.78	0.72	95.00 ± 3.01	0.78	95.61 ± 2.25	0.34	95.68 ± 2.99	0.39
Car	82.49 ± 4.91	83.57 ± 4.04	0.60	–	–	–	–	99.98 ± 0.07	$\mathbf{0.00^{-}}$
Ring	96.55 ± 0.57	–	–	–	–	–	–	93.84 ± 1.20	$\mathbf{0.00^{+}}$
Satimage	77.08 ± 5.41	–	–	–	–	–	–	80.50 ± 1.89	0.07
Average	**85.56**	78.04		78.46		76.93		81.34	
$\mathbf{W/L/T/N_R}$			4/0/2/2		4/0/1/3		4/0/1/3		4/1/3/0

The "$W/L/T/N_R$" represents, the number of data sets, where the concern algorithm win, lose, Tie or unable to run (no results) respectively.

the proposed methods, highlighting its lack of scalability to larger datasets. Between the two $IADM$-based approaches, $LOMR$ consistently outperforms $PRCMR$ in terms of execution time across all datasets. This can be attributed to more effective use of the $IADM$ framework and the relatively lower computational overhead of the $LOMR$ (or $L_{\mathbb{B}}$) heuristic compared to the PRC heuristic. Whereas, the greedy sequential forward selection algorithm $FSPA$ is not comparable to the other algorithms, as it is only a single reduct algorithm, hence it naturally takes a lesser amount of time to obtain a reduct. Furthermore, since the proposed approaches are not constrained to generating minimal-size reducts, the average length of the reducts does not hinder their efficiency in producing diverse and useful sets of multiple reducts.

5.5. Significance of diversity through classification analysis

This section presents a classification-based evaluation to demonstrate the effectiveness of the diversity criterion employed in the proposed $MRCMR$ algorithm, in comparison with the base approach $LOMR/MRGA\_MRC$ and other state-of-the-art algorithms. A ten-fold cross-validation approach was applied to benchmark datasets; in each iteration, nine folds were used to generate reducts and train the classification models, while the remaining fold was reserved for testing. Since $LOMR$ is a scalable variant of $MRGA\_MRC$ and generates identical reduct sets, only $MRGA\_MRC$ is included in the results as their representative, as shown in Tables 9 to 10. Two widely used classifiers—Naive Bayes (NB) and Support Vector Machine (SVM) with a radial basis function kernel—were employed to induce models. For methods that produce multiple reducts, including $PRCMR$, $LOMR$, DR, and $BBFS$, ensemble classification models were constructed by training 15-base classifiers on the obtained 15-diverse reducts and aggregating their predictions through majority voting [47]. In contrast, $FSPA$, which produces a single reduct, was used to train a standard (non-ensemble) classification model. The classification performance across all folds is reported as the mean and standard deviation ($\mu \pm \sigma$). To determine the statistical significance of any observed differences, an unpaired Student's t-test was performed, and the resulting p-values (p_{val}) are reported. A p-value ($p_{val} \leq 0.05$) indicates a statistically significant difference; such cases are highlighted in bold, with superscript symbols "+" or "−" indicating whether the proposed algorithm performed better or worse than the compared method, respectively. For all other cases, performance is considered comparable. Each result table also reports the overall average accuracy and provides a summary indicator ($W/L/T/N_R$) that shows how often the proposed method outperformed, underperformed, tied with, or did not produce results for the evaluated datasets.

Analysis of Results: A detailed analysis of the results in Tables 9 to 10 highlights the clear advantage of the proposed $PRCMR$, which uses partition refinement cardinality as its diversity criterion. Across datasets, $PRCMR$ consistently outperforms the base $MRGA\_MRC$ and the state-of-the-art DR and $BBFS$ methods. Notably, it achieves statistically significant gains on four datasets—Austra, Dna, Hawks, and Wine—with both Naive Bayes and SVM classifiers. In comparison, $MRGA\_MRC$ completed on six datasets, while DR and $BBFS$ ran on five each, but neither showed broader or consistent improvement. On all other datasets, $PRCMR$ matched or surpassed competing methods. These results confirm that the proposed criterion yields more robust and effective reduct

Table 10
Compared results of the Gaussian kernel-based SVM classification models.

Dataset	$PRCMR$	$MRGA\_MRC$		DR		$BBFS$		$FSPA$	
	$\mu \pm \sigma$	$\mu \pm \sigma$	p_{val}	$\mu \pm \sigma$	p_{val}	$\mu \pm \sigma$	p_{val}	$\mu \pm \sigma$	p_{val}
Austra	72.32 ± 4.76	64.78 ± 5.64	$\mathbf{0.00}^+$	65.36 ± 8.55	$\mathbf{0.03}^+$	62.90 ± 5.89	$\mathbf{0.00}^+$	55.36 ± 5.75	$\mathbf{0.00}^+$
Dna	91.52 ± 3.36	80.78 ± 4.55	$\mathbf{0.00}^+$	80.79 ± 4.28	$\mathbf{0.00}^+$	81.33 ± 4.31	$\mathbf{0.00}^+$	78.90 ± 4.31	$\mathbf{0.00}^+$
Hawks	92.32 ± 4.05	80.51 ± 4.90	$\mathbf{0.00}^+$	80.41 ± 4.15	$\mathbf{0.00}^+$	80.51 ± 4.07	$\mathbf{0.00}^+$	78.86 ± 4.60	$\mathbf{0.00}^+$
Wine	84.47 ± 11.58	61.91 ± 14.84	$\mathbf{0.00}^+$	70.54 ± 14.91	$\mathbf{0.02}^+$	60.33 ± 15.29	$\mathbf{0.00}^+$	49.91 ± 15.23	$\mathbf{0.00}^+$
wdbc	97.22 ± 2.18	96.53 ± 1.89	0.46	97.86 ± 2.50	0.54	96.37 ± 1.82	0.36	95.30 ± 2.88	0.11
Car	89.32 ± 5.93	88.36 ± 5.40	0.71	–	–	–	–	99.98 ± 0.07	$\mathbf{0.00}^-$
Ring	96.49 ± 0.74	–	–	–	–	–	–	76.81 ± 2.04	$\mathbf{0.00}^+$
Satimage	82.55 ± 1.43	–	–	–	–	–	–	19.20 ± 5.16	$\mathbf{0.00}^+$
Average	**88.27**	78.81		78.99		76.28		69.29	
$\mathbf{W/L/T/N_R}$			4/0/2/2		4/0/1/3		4/0/1/3		6/1/1/0

The “$W/L/T/N_R$” represents, the number of data sets, where the concern algorithm win, lose, Tie or unable to run (no results) respectively.

ensembles than the least-overlap heuristic in $LOMR$, the Jaccard similarity–based diversity in DR, and the breadth-first search–based n-minimal reducts in $BBFS$.

Compared to the single-reduct method $FSPA$, the proposed approach achieved statistically significant improvements on five datasets—Austra, Dna, Hawks, Wine, and Ring—with both classifiers. For the Satimage dataset under the SVM classifier, $PRCMR$ also outperformed $FSPA$. The only case where $FSPA$ showed similar performance was on the Car dataset. These findings highlight the practical benefits of generating diverse multiple reducts for ensemble learning, as opposed to relying on a single reduct.

In summary, the experimental results demonstrate that integrating the proposed $IADM$ framework into multiple reduct computation significantly improves computational efficiency and scalability, especially for datasets where constructing a conventional DM is infeasible. By enabling the processing of larger datasets that exceed the practical limits of traditional ADM methods, $IADM$ broadens the applicability of existing algorithms for single, optimal, and multiple reduct computation. Although the PRC heuristic introduces some additional computational overhead, it effectively enhances ensemble performance by ensuring greater diversity among reducts, thereby fulfilling its intended purpose and surpassing existing diversity criteria in rough set–based ensemble models.

6. Conclusion

This paper has tackled two critical challenges in the domain of rough set–based ensemble learning. First, the scalability bottlenecks of traditional discernibility matrix–based reduct computation. Second, the limitations of conventional overlap-based heuristics in ensuring meaningful diversity among multiple reducts.

Although classical frameworks such as the Absorption Discernibility Matrix and standard discernibility matrix are theoretically capable of generating all possible reducts, their practical application is often restricted by the high computational cost and memory overhead issue associated with constructing and storing entire DM especially when dealing with comparatively larger datasets. To address this, we have proposed an incremental ADM framework that computes absorbed discernibility matrices directly, without first requiring the complete DM. The proposed approach removes the need of store an entire DM in memory, enabling the proposed algorithms to handle datasets that would otherwise be infeasible due to resource constraints.

Alongside this computational improvement, this work also identifies the shortcomings of existing overlap-based diversity criteria, which measure only attribute-level differences and overlook the functional equivalence that can arise when different reducts induce the same partitions. To overcome this, we introduced a novel Partition Refinement Cardinality heuristic that quantifies diversity by examining how partitions differ, better capturing the true generalizability of selected reducts in building diverse base classifiers.

By integrating the $IADM$ framework with the proposed PRC heuristic and the conventional least-overlap method, we developed the new $PRCMR$ and $LOMR$ algorithms for scalable and diverse multiple reduct computation. Both theoretical analysis and experimental results demonstrate that the proposed approach enhances the scalability and practical applicability of rough set–based ensemble classification, even for larger datasets where traditional DM construction becomes impractical.

Looking ahead, while the $IADM$ framework has proven effective for scalable multiple reduct computation, its potential can be further extended to develop robust algorithms for single, optimal, and hybrid reducts computation scenarios. In addition, the PRC heuristic provides a strong starting point for partition-based diversity assessment, but there remains scope to design more advanced or hybrid heuristics that further strengthen the diversity and robustness of ensemble models.

CRediT authorship contribution statement

Abhimanyu Bar: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Anil Kumar:** Writing – review & editing, Software, Data curation, Conceptualization. **G.Y. Phani Kumar:** Software, Resources, Data curation. **P.S.V.S. Sai Prasad:** Writing – review & editing, Visualization, Validation, Project administration, Methodology, Funding acquisition, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Author acknowledges the financial support of Institute of Eminence for University of Hyderabad ($UoH - IoE$) by Ministry of Human Resource Development, Government of India (No: $F11/9/2019 - U3(A)$).

Data availability

Data will be made available on request.

References

[1] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (5) (1982) 341–356.
[2] J. Dai, H. Han, Q. Hu, M. Liu, Discrete particle swarm optimization approach for cost sensitive attribute reduction, Knowl.-Based Syst. 102 (2016) 116–126.
[3] R. Susmaga, Computation of shortest reducts, Found. Comput. Decision Sci. 23 (2) (1998) 119–137.
[4] Q. Zhang, S. Yang, G. Wang, Measuring uncertainty of probabilistic rough set model from its three regions, IEEE Trans. Syst. Man Cybern. Syst. 47 (12) (2016) 3299–3309.
[5] H. Rana, M. Lal, A rough set theory approach for rule generation and validation using rses, Int. J. Rough Sets Data Anal. 3 (1) (2016) 55–70.
[6] P. Gogoi, D.K. Bhattacharyya, J.K. Kalita, A rough set–based effective rule generation method for classification with an application in intrusion detection, Int. J. Secur. Netw. 8 (2) (2013) 61–71.
[7] S. Xia, X. Bai, G. Wang, Y. Cheng, D. Meng, X. Gao, Y. Zhai, E. Giem, An efficient and accurate rough set for feature selection, classification, and knowledge representation, IEEE Trans. Knowl. Data Eng. (2022).
[8] H.S. Own, H. Yahyaoui, Rough set based classification of real world web services, Inf. Syst. Front. 17 (6) (2015) 1301–1311.
[9] Y. Li, L. Zhang, Y. Xu, Y. Yao, R.Y.K. Lau, Y. Wu, Enhancing binary classification by modeling uncertain boundary in three-way decisions, IEEE Trans. Knowl. Data Eng. 29 (7) (2017) 1438–1451.
[10] A. Skowron, C. Rauszer, The Discernibility Matrices and Functions in Information Systems, Springer Netherlands, Dordrecht, 1992, pp. 331–362.
[11] A. Bar, P. Sai Prasad, Multiple reducts computation in rough sets with applications to ensemble classification, in: Proceedings of ICETIT 2019, Springer, 2020, pp. 449–461.
[12] R. Susmaga, Computation of minimal cost reducts, in: International Symposium on Methodologies for Intelligent Systems, Springer, 1999, pp. 448–456.
[13] A. Bar, A. Kumar, P. Sai Prasad, Coarsest granularity-based optimal reduct using a* search, Granul. Comput. (2022) 1–22.
[14] A. Bar, P. Prasad, Approaches for coarsest granularity based near-optimal reduct computation, Appl. Intell. (2022) 1–26.
[15] J. Cao, M. Ahmadi, M. Shridhar, Recognition of handwritten numerals with multiple feature and multistage classifier, Pattern Recognit. 28 (2) (1995) 153–160.
[16] E. Stamatatos, G. Widmer, Automatic identification of music performers with learning ensembles, Artif. Intell. 165 (1) (2005) 37–56.
[17] T.G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Mach. Learn. 40 (2) (2000) 139–157.
[18] Y. Sun, M.S. Kamel, A.K. Wong, Empirical study on weighted voting multiple classifiers, in: International Conference on Pattern Recognition and Image Analysis, Springer, 2005, pp. 335–344.
[19] Z.-H. Zhou, J. Wu, W. Tang, Ensembling neural networks: many could be better than all, Artif. Intell. 137 (1–2) (2002) 239–263.
[20] A. Tsymbal, M. Pechenizkiy, P. Cunningham, Diversity in search strategies for ensemble feature selection, Inf. Fusion 6 (1) (2005) 83–98.
[21] C. Elkan, Boosting and naıve Bayesian learning, in: Proceedings of KDD-97, New Port Beach, CA, 1997.
[22] Q. Hu, D. Yu, Z. Xie, X. Li, Eros: ensemble rough subspaces, Pattern Recognit. 40 (12) (2007) 3728–3739.
[23] J. Kittler, M. Hatef, R.P. Duin, J. Matas, On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 226–239.
[24] J.F. Díez-Pastor, J.J. Rodríguez, C.I. García-Osorio, L.I. Kuncheva, Diversity techniques improve the performance of the best imbalance learning ensembles, Inf. Sci. 325 (2015) 98–117.
[25] F. Jiang, X. Yu, H. Zhao, D. Gong, J. Du, Ensemble learning based on random super-reduct and resampling, Artif. Intell. Rev. 54 (4) (2021) 3115–3140.
[26] Y. Guo, L. Jiao, S. Wang, S. Wang, F. Liu, K. Rong, T. Xiong, A novel dynamic rough subspace based selective ensemble, Pattern Recognit. 48 (5) (2015) 1638–1652.
[27] K. Liu, X. Yang, H. Yu, J. Mi, P. Wang, X. Chen, Rough set based semi-supervised feature selection via ensemble selector, Knowl.-Based Syst. 165 (2019) 282–296.
[28] K. Jackowski, New diversity measure for data stream classification ensembles, Eng. Appl. Artif. Intell. 74 (2018) 23–34.
[29] A. Trabelsi, Z. Elouedi, E. Lefevre, An ensemble classifier through rough set reducts for handling data with evidential attributes, Inf. Sci. 635 (2023) 414–429.
[30] B. Abhimanyu, Novel approaches for optimal near optimal reduct and multiple reducts computation, Ph.D. thesis, University of Hyderabad, 2022, http://hdl.handle.net/10603/511851.
[31] H. Liu, F. Hussain, C.L. Tan, M. Dash, Discretization: an enabling technique, Data Min. Knowl. Discov. 6 (4) (2002) 393–423.
[32] Y. Yao, Y. Zhao, Discernibility matrix simplification for constructing attribute reducts, Inf. Sci. 179 (7) (2009) 867–882.
[33] R. Jensen, Rough set-based feature selection: a review, in: Rough Computing: Theories, Technologies and Applications, 2008, pp. 70–107.
[34] Y. Yao, Y. Zhao, J. Wang, On reduct construction algorithms, in: Transactions on Computational Science II, Springer, 2008, pp. 100–117.
[35] R. Susmaga, Parallel computation of reducts, in: L. Polkowski, A. Skowron (Eds.), Rough Sets and Current Trends in Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 450–458.
[36] V. Rodríguez-Diez, J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, M.S. Lazo-Cortés, J.A. Olvera-López, Minreduct: a new algorithm for computing the shortest reducts, Pattern Recognit. Lett. 138 (2020) 177–184.
[37] D. Chen, S. Zhao, L. Zhang, Y. Yang, X. Zhang, Sample pair selection for attribute reduction with rough set, IEEE Trans. Knowl. Data Eng. 24 (11) (2012) 2080–2093.
[38] M. Choromański, T. Grześ, P. Hońko, Breadth search strategies for finding minimal reducts: towards hardware implementation, Neural Comput. Appl. 32 (18) (2020) 14801–14816.
[39] Q.A. Al-Radaideh, M.A. Al-Abrat, An Arabic text categorization approach using term weighting and multiple reducts, Soft Comput. 23 (14) (2019) 5849–5863.
[40] Z. Pawalk, Rough Sets: Theoretical Aspects of Reasoning About Data, Kluwer Academic Publishers, Dordrecht, 1991.
[41] J. Qian, D. Miao, Z. Zhang, X. Yue, Parallel attribute reduction algorithms using mapreduce, Inf. Sci. 279 (2014) 671–690.

[42] L.G. Nguyen, H.S. Nguyen, On elimination of redundant attributes in decision tables, in: 2012 Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE, 2012, pp. 317–322.

[43] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, J. Mult.-Valued Log. Soft Comput. 17 (2011), https://sci2s.ugr.es/keel/category.php?cat=clas&order=name#sub2.

[44] D. Dua, E. Karra Taniskidou, UCI machine learning repository, http://archive.ics.uci.edu/ml, 2017.

[45] V. Arel-Bundock, Rdatasets: an archive of datasets distributed with R, https://vincentarelbundock.github.io/Rdatasets/datasets.html, 2012.

[46] Y. Qian, J. Liang, W. Pedrycz, C. Dang, Positive approximation: an accelerator for attribute reduction in rough set theory, Artif. Intell. 174 (9–10) (2010) 597–618, https://doi.org/10.1016/j.artint.2010.04.018.

[47] V.S. Sheng, J. Zhang, B. Gu, X. Wu, Majority voting and pairing with multiple noisy labeling, IEEE Trans. Knowl. Data Eng. 31 (7) (2017) 1355–1368.