Pattern Recognition 159 (2025) 111141

Contents lists available at ScienceDirect

Pattern Recognition

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

# Incremental feature selection: Parallel approach with local neighborhood rough sets and composite entropy

Weihua Xu [*], Weirui Ye

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

ARTICLE INFO

*Keywords:*
Composite entropy
Feature selection
Incremental algorithm
Rough set

ABSTRACT

Rough set theory is a powerful mathematical framework for managing uncertainty and is widely utilized in feature selection. However, traditional rough set-based feature selection algorithms encounter significant challenges, especially when processing large-scale incremental data and adapting to the dynamic nature of real-world scenarios, where both data volume and feature sets are continuously changing. To overcome these limitations, this study proposes an innovative algorithm that integrates local neighborhood rough sets with composite entropy to measure uncertainty in information systems more accurately. By incorporating decision distribution, composite entropy enhances the precision of uncertainty quantification, thereby improving the effectiveness of the algorithm in feature selection. To further improve performance in handling large-scale incremental data, matrix operations are employed in place of traditional set-based methods, allowing the algorithm to fully utilize modern hardware capabilities for accelerated processing. Additionally, parallel computing technology is integrated to further enhance computational speed. An incremental version of the algorithm is also introduced to better adapt to dynamic data environments, increasing its flexibility and practicality. Comprehensive experimental evaluations demonstrate that the proposed algorithm significantly surpasses existing methods in both effectiveness and efficiency.

## 1. Introduction

Rough Set Theory [1], proposed by Pawlak in 1982, is a mathematical framework for managing ambiguous data by employing upper and lower approximations. However, traditional rough sets depend on strict inclusion relationships, which restrict their fault tolerance and applicability in complex scenarios. The 0.5 Probabilistic Rough Set [2] addresses these limitations by integrating conditional probabilities and thresholds, thereby enhancing flexibility and fault tolerance. Building on this, Yao et al. developed the Decision-Theoretic Rough Set [3], which further improves robustness by introducing fault-tolerant thresholds. In the era of big data, challenges such as limited labeling and low efficiency emerge. To tackle these issues, Qian et al. proposed Local Rough Sets [4], which improve performance when handling large datasets. Wang et al. extended this approach with Local Neighborhood Rough Sets [5], adapting it to real-world data environments.

As a versatile theory for data analysis and processing, Rough Set Theory has been successfully applied across various fields, including machine learning [6], decision making [7], and outlier detection [8]. Rough sets have also made significant contributions to granular computing, particularly in areas like Concept-Cognitive Learning (CCL). For example, the cognitive process of learning approximate concepts is inspired by the notion of approximation spaces within Rough Set Theory [9]. Additionally, when combined with the Three-Way Decision (3WD) model, which is derived from Rough Set Theory and operates within a dynamic fuzzy framework, CCL has been employed to model real-time cognitive updating procedures [10]. Moreover, the adaptability of Rough Set Theory extends beyond traditional single-label datasets to multi-label [11] and incomplete datasets [12], highlighting its wide applicability.

Among these various applications, one of the most prominent areas of research is feature selection. Traditional feature selection algorithms based on rough sets typically quantify the consistency of information systems by partitioning them into positive, negative, and boundary regions, utilizing discernibility matrices for feature selection [13]. However, as research progressed, these methods revealed certain limitations. To address these challenges, entropy, a fundamental concept

[*] Corresponding author.
*E-mail addresses:* chxuwh@gmail.com (W. Xu), reanye233@gmail.com (W. Ye).

https://doi.org/10.1016/j.patcog.2024.111141
Received 6 August 2024; Received in revised form 15 September 2024; Accepted 27 October 2024
Available online 5 November 2024

in information theory, was introduced to more accurately measure uncertainty in rough sets, leading to significant advancements.

For instance, Sun et al. proposed the Fuzzy Neighborhood Multi-granulation Rough Set (FNMRS) approach, which enhances both pre-processing and classification across diverse datasets by incorporating uncertainty measures to optimize feature selection [14]. Similarly, Wang et al. integrated fuzzy rough approximations with self-information to develop four types of uncertainty measures, aimed at evaluating the classification performance of attribute subsets [15]. Building on these developments, Sang et al. introduced incremental feature selection methods using a novel conditional entropy to handle dynamic ordered data, leading to the creation of the Fuzzy Dominance Neighborhood Rough Set (FDNRS) model [16]. Additionally, Xu et al. presented a local composite entropy approach based on neighborhood rough sets to address feature selection challenges in fuzzy and imbalanced datasets [17]. Expanding on these concepts, Yuan et al. proposed a novel zentropy-based uncertainty measure for feature selection, leveraging granular structures within the knowledge space [18]. Yuan et al. further developed the zentropy-based uncertainty measure for heterogeneous feature selection, an effective and robust method that broadens the practical applications of entropy in rough set-based feature selection [19].

As datasets grow and evolve, the need for rapid and adaptive feature selection has become critical. Researchers have increasingly integrated incremental learning and parallel computing principles into Rough Set Theory. Huang et al. developed an incremental feature selection method for hierarchical classification using fuzzy rough set techniques [20]. Zhao and colleagues extended tolerance Rough Set Theory by introducing the Subrelation Tolerance Class (STC) and developed an algorithm for managing incomplete streaming data during incremental feature selection [21]. They also devised a Consistency Approximation (CA)-based framework to refine fuzzy Rough Set Theory for feature selection [22]. Pan et al. proposed an incremental feature selection algorithm based on weighted dominance-based neighborhood rough sets [23]. Additionally, Zhang et al. created an incremental feature selection mechanism for interval-valued fuzzy decision information systems, using $\lambda$-fuzzy similarity self-information to dynamically update feature selections [24]. Xu et al. presented a generalized incremental multi-granulation neighborhood rough set approach based on a weight partition model in matrix form [25].

Parallel computing also plays a key role in accelerating rough set algorithms for feature selection. Chen et al. explored parallel attribute reduction using Dominance-based Neighborhood Rough Sets (DNRS), which accommodate both numerical and categorical attribute values in multi-criteria decision-making frameworks [26]. Nosheen et al. introduced a parallel method for calculating Dominance-based Rough Set Approach (DRSA) approximation sets without computing dominance relations, employing heuristic guidelines [27]. Finally, Turaga et al. developed the Parallel Algorithm for Computing Probabilistic Rough Set Approximations (PACPRSA), which allows for parallel computation of regions and approximations in probabilistic rough sets [28].

Building on these impressive results, we introduce a novel feature selection algorithm that integrates local neighborhood rough sets with composite entropy. Local neighborhood rough sets excel at processing rough data with numerical attributes, delivering not only satisfactory analysis outcomes but also significantly enhanced efficiency compared to global rough sets. Unlike other forms of information entropy, composite entropy places greater emphasis on quantifying uncertainty and considering decision distribution, thereby improving its capacity to describe uncertainty. As a result, it performs more accurately and efficiently in feature evaluation and selection. By leveraging the strengths of both approaches, we achieve an excellent and efficient feature selection algorithm.

Meanwhile, to address practical needs, we have developed an incremental version of our algorithm for feature selection in dynamic datasets, which can handle evolving datasets with changing data volume and features over time, significantly reducing computational time and resource consumption. We have replaced set operations with matrix operations, allowing for accelerated rough set calculations using scientific computing packages such as NumPy, thus enhancing computational efficiency. Matrix operations are inherently more efficient than set operations due to their optimized algorithms and hardware acceleration, leading to faster processing and better handling of large datasets. Additionally, both algorithms have been optimized with parallel computing to fully exploit the multi-core capabilities of modern CPUs, greatly improving running speed. In summary, this paper presents two parallel, matrix-based feature selection algorithms–one for static datasets and one for dynamic datasets–based on local neighborhood rough sets and composite entropy, with a simplified flowchart of our dynamic data algorithm shown in Fig. 1.

The research contributions are delineated as follows:

(1) Proposed a feature selection algorithm combining local neighborhood rough sets with composite entropy, leveraging the strengths of both to achieve improved feature selection performance. This algorithm was further extended into an incremental version, allowing it to handle dynamic datasets with changing data volumes and features, thus meeting real-world needs. Compared to static algorithms, the incremental approach eliminates the need for recalculations, resulting in higher efficiency.

(2) Innovatively transformed set operations in local neighborhood rough sets and composite entropy into matrix operations, enabling the algorithm to fully utilize the advantages of matrix computation in modern computers. This significantly accelerates the calculation speed of rough sets and composite entropy, greatly enhancing the computational efficiency of the feature selection algorithm.

(3) Introduced parallel computing into rough set-based feature selection algorithms, leveraging the multi-core capabilities of CPUs to distribute the computational workload across multiple resources. This further improves the computational efficiency of the feature selection process.

This paper is organized as follows. Section 2 introduces related works and basics about neighborhood rough set and composite entropy. In Section 3, we explore the use of matrices to handle rough sets, potentially increasing computational speed. In Section 4, we propose two parallel feature selection algorithms, one for static data and another for dynamic data. Both algorithms aim to improve efficiency. Section 5 presents experiments on classification accuracy and reduction efficiency, along with result analysis. Finally, we provide a summary and discuss future research directions in Section 6.

## 2. Preliminaries

In this section, we will introduce some fundamental concepts of local neighborhood rough sets and composite entropy.

### 2.1. Local neighborhood rough set

The concept of information system is utilized in Rough Set Theory to describe data, which can be represented by a 4-tuple $IS = (U, A \cup D, V, f)$. Here, $U = \{x_1, x_2, \dots, x_p\}$ represents the finite set of samples, $A = \{a_1, a_2, \dots, a_q\}$ represents the finite set of conditional attributes, and $D = \{d\}$ is the set of decision attributes. When there are NaN values in the information system, they will be filled as 0 during computation.

Table 1 provides an example of information, where $U = \{x_1, x_2, \dots, x_8\}$ and $A = \{a_1, a_2, \dots, a_9\}$.

Now, assuming the set $B$ is a subset of $A$, the neighborhood class under the relation $R_B^\delta$ is defined as:

$$[x]_{R_B^\delta} = \{y \mid \Delta(x, y) \le \delta,\ x, y \in U\},$$

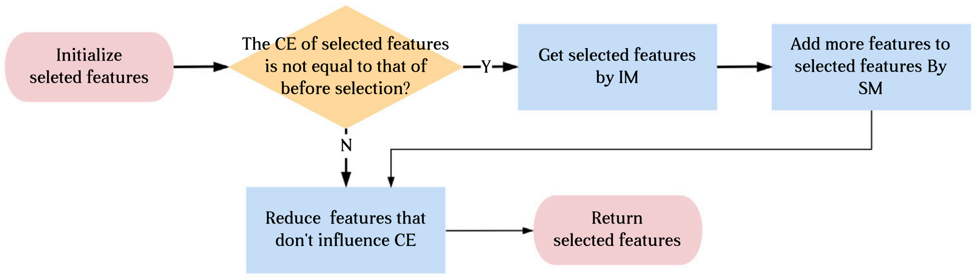
Initialize seleted features
The CE of selected features is not equal to that of before selection?
Y
N
Get selected features by IM
Add more features to selected features By SM
Reduce features that don't influence CE
Return selected features

**Fig. 1.** Simple flowchart of algorithm for dynamic data (Algorithm 4: PMLCE-D).

**Table 1**
An information system.

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ | $a_9$ | $d$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 0.57 | 0.27 | 0.44 | 0.34 | 0.37 | 0.29 | 0.54 | 0.42 | NaN | 1 |
| $x_2$ | 0.37 | 0.21 | 0.46 | 0.56 | 0.51 | 0.31 | 0.44 | 0.66 | NaN | 1 |
| $x_3$ | 0.37 | 0.22 | 0.29 | 0.32 | 0.35 | 0.26 | 0.28 | 0.66 | 0.64 | 1 |
| $x_4$ | 0.30 | 0.36 | 0.59 | 0.99 | 0.97 | 0.39 | 0.70 | 0.32 | 0.30 | 1 |
| $x_5$ | 0.82 | 0.29 | 0.42 | 0.37 | 0.39 | 0.39 | 0.86 | 0.39 | 0.51 | 0 |
| $x_6$ | 0.30 | 0.20 | 0.37 | 0.44 | 0.43 | 0.34 | 0.82 | 0.36 | NaN | 0 |
| $x_7$ | 0.19 | 0.23 | 0.29 | 0.34 | 0.36 | 0.35 | 0.61 | 0.35 | NaN | 0 |
| $x_8$ | 0.15 | 0.29 | 0.33 | 0.39 | 0.39 | 0.62 | 0.92 | 0.31 | 0.26 | 0 |

where $\Delta(x, y)$ is the Euclidean distance between $x$ and $y$, and $\delta$ is a hyper-parameter named neighbor radius. The Euclidean distance between two samples $x$ and $y$ is calculated as:

$$\Delta(x, y) = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2}.$$

Subsequently, the local lower and upper approximations [4], crucial concepts in Rough Set Theory that aid in measuring the uncertainty of an information system, can be computed as follows:

$$\underline{R_B^\delta}(Z_i) = \{x | [x]_{R_B^\delta} \subseteq Z_i, x \in Z_i\},$$
$$\overline{R_B^\delta}(Z_i) = \{x | [x]_{R_B^\delta} \cap Z_i \neq \emptyset, x \in Z_i\}.$$

Furthermore, for $U/D = Z = \{Z_1, Z_2, \ldots, Z_s\}$, the local lower approximation $\underline{R_B^\delta}(Z)$ and upper approximation $\overline{R_B^\delta}(Z)$ are:

$$\underline{R_B^\delta}(Z) = \{\underline{R_B^\delta}(Z_1), \underline{R_B^\delta}(Z_2), \ldots, \underline{R_B^\delta}(Z_s)\},$$
$$\overline{R_B^\delta}(Z) = \{\overline{R_B^\delta}(Z_1), \overline{R_B^\delta}(Z_2), \ldots, \overline{R_B^\delta}(Z_s)\}.$$

### 2.2. Composite entropy

However, even with the utilization of positive and negative regions along with boundary region, we still encounter challenges in accurately measuring the uncertainty of information systems, and the effectiveness of feature selection remains unsatisfactory. To address this limitation, entropy is introduced into Rough Set Theory—a pivotal concept in information theory employed to quantify the uncertainty of possible outcomes of random variables. Through entropy, we can more precisely characterize the uncertainty of information systems, where certainty weakens as entropy increases.

In this paper, we choose to employ composite entropy [17] to assess the uncertainty of this information system, which focuses on the uncertainty measure and decision distribution, improving its ability to describe uncertainty and better evaluate selection features.

If the neighbor radius is denoted as $\delta$, the composite entropy can be expressed as follows:

$$CE(B, Z) = -\sum_{i=1}^{s} \frac{|Z_i|}{|U|} \ln \frac{|\underline{R_B^\delta}(Z_i)|}{|\overline{R_B^\delta}(Z_i)|}.$$

**Proposition 1.** *For $b \in B \subseteq A$, $U/D = Z = \{Z_1, Z_2, \ldots, Z_s\}$, and $B \subseteq B'$, the following properties hold:*

*(1) $CE(B, Z) \geq 0$;*
*(2) $CE(B', Z) \leq CE(B, Z)$;*
*(3) If there exists $Z_i$ such that $\frac{|\underline{R_B^\delta}(Z_i)|}{|\overline{R_B^\delta}(Z_i)|} = 0$ for $i = 1, 2, \ldots, s$, then $CE(B, Z) \to \infty$;*
*(4) If $\frac{|\underline{R_B^\delta}(Z_i)|}{|\overline{R_B^\delta}(Z_i)|} = 1$ for all $Z_i$ ($i = 1, 2, \ldots, s$), then $CE(B, Z) = 0$.*

**Proposition 2.** *For a given set of samples $U = \{x_1, x_2, \ldots, x_p\}$, consider an attribute subset $b \in B \subseteq A$, and let $U/D = Z = \{Z_1, Z_2, \ldots, Z_s\}$. The inner and outer significance measures of attribute $a \in A$ are defined as follows:*

$$IM(b, B, Z) = CE(B - \{b\}, Z) - CE(B, Z),$$
$$SM(b, B, Z) = CE(B, Z) - CE(B \cup \{b\}, Z).$$

*Moreover,*

*(1) A higher value of $IM(b, B, Z)$ or $SM(b, B, Z)$ indicates greater importance of the feature $b$ relative to $B$;*
*(2) If $CE(B - \{b\}, Z) \to \infty$ and $CE(B, Z) \to \infty$, then $IM(b, B, Z) = 0$; If $CE(B, Z) \to \infty$ and $CE(B \cup b, Z) \to \infty$, then $SM(b, B, Z) = 0$.*

## 3. Local neighborhood rough set and composite entropy based on matrix

In this section, we will introduce the matrix-form representation of local neighborhood rough sets and the computation method of composite entropy. By replacing set operations with matrix computations, the calculations of rough sets can be accelerated using scientific computing packages such as Numpy, significantly enhancing the operational speed. Furthermore, this approach enables the possibility of leveraging GPUs for rough set computations, which will further boost the computational efficiency.

The main contents of this section are as follows:

Definition 1 discusses how sets can be converted into matrix form and the definition of submatrices. Our algorithm performs rough set operations through matrix computations, while utilizing submatrices to update the original matrix. This serves as the cornerstone for incremental computation in our algorithm.

Definition 2 delves into the computation method of distance matrices, while Definition 3 builds upon Definition 2 to discuss the calculation of neighborhood matrices. These are fundamental for computing the lower and upper approximations in neighborhood rough sets.

Definition 4 presents the matrix-based computation method for the lower and upper approximations in neighborhood rough sets. By leveraging these approximation matrices, we can calculate the composite entropy according to Definition 5.

To facilitate readers' verification, we have attached an example to each definition.

**Definition 1.** Given a finite field $U = \{x_1, x_2, \dots, x_p\}$ where any subset $Z_i \subseteq U$ can be transformed into a boolean matrix $\mathcal{Z}_i$, where 0 represents false and 1 represents true.

$$\mathcal{Z}_i = \begin{bmatrix} z_{i1} & z_{i2} & \dots & z_{ip} \end{bmatrix}_{j \le p}, z_{ij} = \begin{cases} 0, & z_{ij} \notin Z_i, \\ 1, & z_{ij} \in Z_i. \end{cases}$$

So, $Z = \{Z_1, Z_2, \dots, Z_s\}$ can also be transformed to

$$\mathcal{Z} = \begin{bmatrix} \mathcal{Z}_1 \\ \mathcal{Z}_2 \\ \vdots \\ \mathcal{Z}_s \end{bmatrix} = \begin{bmatrix} z_{11} & z_{12} & \dots & z_{1p} \\ z_{21} & z_{22} & \dots & z_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ z_{s1} & z_{s2} & \dots & z_{sp} \end{bmatrix}.$$

Meanwhile, if $M \subseteq \{i \in \mathbb{Z}^+ \mid i \le s\}$ and $N \subseteq \{j \in \mathbb{Z}^+ \mid j \le p\}$ can be represented as

$$\mathcal{Z}[M, N] = \begin{bmatrix} z_{mn} \end{bmatrix}_{m \in M, n \in N}.$$

**Example 1.** Given a finite field $U = \{x_1, x_2, \dots, x_8\}$ where $Z = \{Z_1, Z_2\}$, $Z_1 = \{x_1, x_2, x_3, x_4\}$, and $Z_2 = \{x_5, x_6, x_7, x_8\}$, we can transform these sets into a matrix as

$$Z = \{Z_1, Z_2\} = \{\{x_1, x_2, x_3, x_4\}, \{x_5, x_6, x_7, x_8\}\},$$

$$\Downarrow$$

$$\mathcal{Z} = \begin{bmatrix} \mathcal{Z}_1 \\ \mathcal{Z}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

If $M = \{1\}$ and $N = \{1, 3\}$, the submatrix of $\mathcal{Z}$ can be shown as

$$\mathcal{Z}[M, N] = \mathcal{Z}[\{1\}, \{1, 3\}] = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

And then, if we let

$$\mathcal{Z}[M, N] = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

The matrix $\mathcal{Z}$ would be updated as

$$\mathcal{Z} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

**Definition 2.** Given an information system $IS = (U, A \cup D, V, f)$ with conditional attributes $B \subseteq A$. Let $G = \{g_1, g_2, \dots, g_v\} \subseteq U$, and $H = \{h_1, h_2, \dots, h_w\} \subseteq U$. When the neighbor radius is $\delta$, the distance matrix between $G$ and $H$ is defined as follows:

$$\mathcal{DM}_B(G, H) = \begin{bmatrix} dm_{11} & dm_{12} & \dots & dm_{1w} \\ dm_{21} & dm_{22} & \dots & dm_{2w} \\ \vdots & \vdots & \ddots & \vdots \\ dm_{v1} & dm_{v2} & \dots & dm_{vw} \end{bmatrix} = \begin{bmatrix} \Delta(g_i, h_j) \end{bmatrix}_{i \le v, j \le w}.$$

**Example 2.** From Table 1, the sample set $U = \{x_1, x_2, \dots, x_8\}$ and conditional attributes set $A = \{a_1, a_2, \dots, a_9\}$. According to Definition 2, we could obtain the distance matrix of the information system when $A = \{a_1, a_2, \dots, a_9\}$

$$\mathcal{DM}_B(U, U) = \begin{bmatrix} 0. & 0.42 & 0.78 & 1.01 & 0.66 & 0.43 & 0.43 & 0.72 \\ 0.42 & 0. & 0.74 & 0.85 & 0.88 & 0.52 & 0.51 & 0.79 \\ 0.78 & 0.74 & 0. & 1.17 & 0.82 & 0.91 & 0.81 & 0.93 \\ 1.01 & 0.85 & 1.17 & 0. & 1.05 & 0.88 & 1.01 & 0.95 \\ 0.66 & 0.88 & 0.82 & 1.05 & 0. & 0.74 & 0.86 & 0.76 \\ 0.43 & 0.52 & 0.91 & 0.88 & 0.74 & 0. & 0.28 & 0.44 \\ 0.43 & 0.51 & 0.81 & 1.01 & 0.86 & 0.28 & 0. & 0.50 \\ 0.72 & 0.79 & 0.93 & 0.95 & 0.76 & 0.44 & 0.50 & 0. \end{bmatrix}.$$

**Definition 3.** Given an information system $IS = (U, A \cup D, V, f)$ with conditional attributes $B \subseteq A$. Let $G = \{g_1, g_2, \dots, g_v\} \subseteq U$, and $H = \{h_1, h_2, \dots, h_w\} \subseteq U$. When the neighbor radius is $\delta$, the distance matrix $\mathcal{DM}_B(G, H) = \begin{bmatrix} dm_{ij} \end{bmatrix}_{i \le v, j \le w}$. The neighborhood matrix between $G$ and $H$ is defined as follows, where 0 represents false and 1 represents true:

$$\mathcal{NM}_B^\delta(G, H) = \begin{bmatrix} nm_{11} & nm_{12} & \dots & nm_{1w} \\ nm_{21} & nm_{22} & \dots & nm_{2w} \\ \vdots & \vdots & \ddots & \vdots \\ nm_{v1} & nm_{v2} & \dots & nm_{vw} \end{bmatrix}, nm_{ij} = \begin{cases} 1, & dm_{ij} \le \delta, \\ 0, & \text{others}. \end{cases}$$

**Example 3.** According to Example 2, the distance matrix is $\mathcal{DM}_B(U, U)$. So, when $\delta = 0.5$, the neighborhood matrix can be computed

$$\mathcal{NM}_B^\delta(U, U) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

**Definition 4.** Given an information system $IS = (U, A \cup D, V, f)$ that consists of samples $U = \{x_1, x_2, \dots, x_p\}$, conditional attributes $B \subseteq A$, and the quotient set between samples and decision attributes $U/D = Z = \{Z_1, Z_2, \dots, Z_s\}$. When the neighbor radius is $\delta$, the set $Z$ can be transformed into matrix $\mathcal{Z} = \begin{bmatrix} z_{ij} \end{bmatrix}_{i \le s, j \le p}$. The neighborhood matrix $\mathcal{NM}_B^\delta(U, U) = \begin{bmatrix} nm_{ij} \end{bmatrix}_{i \le p, j \le p}$ can be computed based on the conditional attribute subset $B$ and neighbor radius $\delta$. Overall, lower and upper approximation matrices can be defined as follows:

$$\underline{\mathcal{R}_B^\delta}(Z) = \begin{bmatrix} \underline{r}_{11} & \underline{r}_{12} & \dots & \underline{r}_{1p} \\ \underline{r}_{21} & \underline{r}_{22} & \dots & \underline{r}_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{r}_{s1} & \underline{r}_{s2} & \dots & \underline{r}_{sp} \end{bmatrix}, \overline{\mathcal{R}_B^\delta}(Z) = \begin{bmatrix} \overline{r}_{11} & \overline{r}_{12} & \dots & \overline{r}_{1p} \\ \overline{r}_{21} & \overline{r}_{22} & \dots & \overline{r}_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \overline{r}_{s1} & \overline{r}_{s2} & \dots & \overline{r}_{sp} \end{bmatrix}.$$

If $Z^{index} = \{Z_1^{index}, Z_2^{index}, \dots, Z_s^{index}\}$, $Z_i^{index} = \{j | z_{ij} = 1, j \in \mathbb{Z}^+\}$, then

$$\underline{\mathcal{R}_B^\delta}(Z)[\{i\}, Z_i^{index}] = \begin{bmatrix} \underline{r}_{ij} \end{bmatrix}_{j \in Z_i^{index}} = \begin{bmatrix} \bigwedge_{k=1}^p (nm_{jk} \to z_{ik}) \end{bmatrix}_{j \in Z_i^{index}},$$

$$\overline{\mathcal{R}_B^\delta}(Z)[\{i\}, Z_i^{index}] = \begin{bmatrix} \overline{r}_{ij} \end{bmatrix}_{j \in Z_i^{index}} = \begin{bmatrix} \bigvee_{k=1}^p (nm_{jk} \wedge z_{ik}) \end{bmatrix}_{j \in Z_i^{index}},$$

in which $\to$ is operator of material implication, $\wedge$ is operator of logical conjunction and $\vee$ is operator of logical disjunction.

Specially, the upper approximation matrix is equivalent to

$$\overline{\mathcal{R}_B^\delta}(Z) = \mathcal{Z} = \begin{bmatrix} z_{ij} \end{bmatrix}_{i \le s, j \le p}.$$

**Example 4.** From Table 1, let $U/D = Z = \{Z_1, Z_2\} = \{\{x_1, x_2, x_3, x_4\}, \{x_5, x_6, x_7, x_8\}\}$, and $Z^{index} = \{Z_1^{index}, Z_2^{index}\} = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}\}$.

We can transform set $Z$ into a matrix:

$$\mathcal{Z} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Next, initialize the lower approximation matrix:

$$\underline{\mathcal{R}_B^\delta}(Z) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

According to Example 3, the neighborhood matrix is $\mathcal{NM}_B^\delta(U, U)$. We can now compute the lower approximation for each class:

$$\underline{\mathcal{R}_B^\delta}(Z)[\{1\}, Z_1^{index}] = \begin{bmatrix} 0 & 1 & 1 & 1 \end{bmatrix}, \underline{\mathcal{R}_B^\delta}(Z)[\{2\}, Z_2^{index}]$$
$$= \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}.$$

The lower and upper approximation matrix are:

$$\underline{\mathcal{R}_B^\delta}(Z) = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}, \overline{\mathcal{R}_B^\delta}(Z)$$
$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix},$$

which are equivalent to:

$$\underline{R}^{\delta}_{B}(Z) = \{\underline{R}^{\delta}_{B}(Z_1), \underline{R}^{\delta}_{B}(Z_2)\} = \{\{x_2, x_3, x_4\}, \{x_5, x_8\}\},$$

$$\overline{R}^{\delta}_{B}(Z) = \{\overline{R}^{\delta}_{B}(Z_1), \overline{R}^{\delta}_{B}(Z_2)\} = \{\{x_1, x_2, x_3, x_4\}, \{x_5, x_6, x_7, x_8\}\}.$$

**Definition 5.** Given an information system $IS = (U, A \cup D, V, f)$ that consists of samples $U = \{x_1, x_2, \dots, x_p\}$, conditional attributes $B \subseteq A$, and the quotient set between samples and decision attributes $U/D = Z = \{Z_1, Z_2, \dots, Z_s\}$. When the neighbor radius is $\delta$, the set $U$ and $Z$ can be transformed into matrices $\mathcal{U} = [u_j]_{j \le p}$ and $\mathcal{Z} = [z_{ij}]_{i \le s, j \le p}$, respectively. Subsequently, the lower and upper approximation matrices $\underline{\mathcal{R}}^{\delta}_{B}(Z) = \left[\underline{r}_{ij}\right]_{i \le s, j \le p}$ and $\overline{\mathcal{R}}^{\delta}_{B}(Z) = \left[\overline{r}_{ij}\right]_{i \le s, j \le p}$ can be computed based on the conditional attribute subset $B$ and neighbor radius $\delta$. Overall, composite entropy based on matrix can be defined as follows:

$$CE(B, Z) = -\sum_{i=1}^{s} \frac{\sum_{j=1}^{p} z_{ij}}{\sum_{j=1}^{p} u_j} ln \frac{\sum_{j=1}^{p} \underline{r}_{ij}}{\sum_{j=1}^{p} \overline{r}_{ij}}.$$

**Example 5.** From Table 1, let $U = \{x_1, x_2, \dots, x_8\}$. It can be transformed into a matrix as follows:

$$\begin{aligned}\mathcal{U} &= \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.\end{aligned}$$

Referring to Example 4, the lower and upper approximation matrices are $\underline{\mathcal{R}}^{\delta}_{B}(Z) = \left[\underline{r}_{ij}\right]_{i \le 2, j \le 8}$ and $\overline{\mathcal{R}}^{\delta}_{B}(Z) = \left[\overline{r}_{ij}\right]_{i \le 2, j \le 8}$. We could also get $\mathcal{Z}$, which is the matrix form of $Z$. Then, We could compute composite entropy

$$CE(A, Z) = -\sum_{i=1}^{2} \frac{\sum_{j=1}^{8} z_{ij}}{\sum_{j=1}^{8} u_j} ln \frac{\sum_{j=1}^{8} \underline{r}_{ij}}{\sum_{j=1}^{8} \overline{r}_{ij}} = -(\frac{4}{8} ln \frac{3}{4} + \frac{4}{8} ln \frac{2}{4}) \approx 0.4904.$$

## 4. Parallel matrix-based feature selection algorithms

In this section, we will introduce two feature selection algorithms proposed by us.

Algorithm 1 outlines the computation method of the neighborhood matrix for static data, while Algorithm 2 utilizes Algorithm 1 to perform parallel feature selection on static data.

Definition 6 introduces the concept of information system updates, and in Proposition 3, we present the updating rules of the neighborhood matrix in a dynamic data environment, which serves as a crucial foundation for our algorithm to achieve incremental updates.

Subsequently, in Algorithm 3, we implement the updating and computation of the neighborhood matrix in a dynamic data environment, and apply it in Algorithm 4 to realize a parallel incremental feature selection algorithm.

### *4.1. Parallel matrix-based feature selection algorithm for static data*

Firstly, we will present a static feature selection algorithm. This algorithm necessitates the recalculation of all neighborhood matrices every time the data is updated, resulting in a high time complexity. Nevertheless, we have integrated parallel computing into this algorithm, thereby enhancing its efficiency to a certain extent.

In Algorithm 1, we compute the neighborhood matrix statically, indicating that each time we perform the calculation on the entire dataset. The time complexity of Algorithm 1 is $O(|U|^2)$.

**Algorithm 1:** Calculate Neighborhood Matrix Statically

```
Input: An Information System $IS = (U, A \cup D, V, f)$, the subset of
       samples $B$ and the neighbor radius $\delta$.
Output: Neighborhood matrix $\mathcal{NM}^{\delta}_{B}(U, U)$.
1 Calculate neighborhood matrix $\mathcal{NM}^{\delta}_{B}(U, U)$ via Definition 3;
2 return $\mathcal{NM}^{\delta}_{B}(U, U)$;
```

In Algorithm 2, we introduce a procedure named PMLCE-S for selecting features in parallel for static data. The steps of Algorithm 2 are outlined in detail. Initially, we compute $IM$ and select attributes with values greater than 0. Next, we calculate $SM$ and choose attributes from the remaining set that maximize $SM$, iteratively, until the entropy of the complete information system is equal to the information system after selection. Finally, we discard attributes for which the entropy remains unchanged even after the selection process. Notably, steps 2 to 9, steps 11 to 15, and steps 19 to 25 can be executed in parallel. The time complexity of Algorithm 2 is $O((|A| + |U||A| + |B|)|U|^2)$.

**Algorithm 2:** PMLCE-S: Parallel Local Neighborhood Rough Set with Composite Entropy for Static Data

```
Input: An Information System $IS = (U, A \cup D, V, f)$, the neighbor
       radius $\delta$.
Output: Selected features $B$.
1  Initialize $B = \emptyset$;
2  do in parallel
3      for $a \in A$ do
4          Compute $im = IM(a, A, D)$ via Proposition 2 and Algorithm 1;
5          if $im > 0$ then
6              $B \leftarrow a$;
7          end
8      end
9  end
10 while $CE(B, D) \neq CE(A, D)$ do          /* Use Algorithm 1 to
     calculate the neighborhood matrix */
11     do in parallel
12         for $a \in (A - B)$ do
13             Compute $sm = SM(a, B, D)$ via Proposition 2 and
                Algorithm 1;
14         end
15     end
16     $a_{max} = \arg\max_{a_{max} \in A - B}(sm)$;
17     $B \leftarrow a_{max}$;
18 end
19 do in parallel
20     for $b \in B$ do
21         if $CE(B - \{b\}, D) = CE(B, D)$ then /* Use Algorithm 1 to
            calculate the neighborhood matrix */
22             $B = B - \{b\}$;
23         end
24     end
25 end
26 return $B$;
```

### *4.2. Parallel matrix-based feature selection algorithm for dynamic data*

Subsequently, we propose a dynamic feature selection algorithm that builds upon Algorithm 1 and Algorithm 2. This algorithm stores the neighborhood matrix calculated each time in the cache. When the data is updated, only the corresponding neighborhood matrix needs to be retrieved from the cache. By utilizing the updated data to update the neighborhood matrix, we achieve a dynamic and incremental algorithm, significantly reducing the time complexity and enhancing the operational efficiency of the algorithm. Additionally, we have incorporated parallel computing into this algorithm to further enhance its operational efficiency.

**Definition 6.** Given an information system $IS = (U, A \cup D, V, f)$, consider an incremental information system $IS' = (U', A' \cup D, V', f')$. The merging of $IS'$ and $IS$ results in a combined information system $\widehat{IS} = (\widehat{U}, \widehat{A} \cup D, \widehat{V}, \widehat{f})$ following these rules:

$$\widehat{U} = U \cup U', \widehat{A} = A \cup A', D = D, \widehat{V} = \widehat{U} \times \widehat{A}, \widehat{f} = \widehat{U} \times D.$$

**Example 6.** Table 1 and Table 2 represent two information systems. We can merge Table 2 with Table 1, resulting in Table 3.

**Table 2**
An incremental information system.

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ | $a_9$ | $d$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 0.57 | 0.27 | 0.44 | 0.34 | 0.37 | 0.29 | 0.54 | 0.42 | 0.62 | 1 |
| $x_2$ | 0.37 | 0.21 | 0.46 | 0.56 | 0.51 | 0.31 | 0.44 | 0.66 | 0.53 | 1 |
| $x_6$ | 0.30 | 0.20 | 0.37 | 0.44 | 0.43 | 0.34 | 0.82 | 0.36 | 0.20 | 0 |
| $x_7$ | 0.19 | 0.23 | 0.29 | 0.34 | 0.36 | 0.35 | 0.61 | 0.35 | 0.37 | 0 |
| $x_9$ | 0.37 | 0.22 | 0.45 | 0.56 | 0.50 | 0.32 | 0.44 | 0.64 | 0.52 | 1 |
| $x_{10}$ | 0.17 | 0.31 | 0.33 | 0.37 | 0.37 | 0.60 | 0.94 | 0.33 | 0.26 | 0 |

**Table 3**
The updated information system.

| | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ | $a_9$ | $d$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 0.57 | 0.27 | 0.44 | 0.34 | 0.37 | 0.29 | 0.54 | 0.42 | 0.62 | 1 |
| $x_2$ | 0.37 | 0.21 | 0.46 | 0.56 | 0.51 | 0.31 | 0.44 | 0.66 | 0.53 | 1 |
| $x_3$ | 0.37 | 0.22 | 0.29 | 0.32 | 0.35 | 0.26 | 0.28 | 0.66 | 0.64 | 1 |
| $x_4$ | 0.30 | 0.36 | 0.59 | 0.99 | 0.97 | 0.39 | 0.70 | 0.32 | 0.30 | 1 |
| $x_5$ | 0.82 | 0.29 | 0.42 | 0.37 | 0.39 | 0.39 | 0.86 | 0.39 | 0.51 | 0 |
| $x_6$ | 0.30 | 0.20 | 0.37 | 0.44 | 0.43 | 0.34 | 0.82 | 0.36 | 0.20 | 0 |
| $x_7$ | 0.19 | 0.23 | 0.29 | 0.34 | 0.36 | 0.35 | 0.61 | 0.35 | 0.37 | 0 |
| $x_8$ | 0.15 | 0.29 | 0.33 | 0.39 | 0.39 | 0.62 | 0.92 | 0.31 | 0.26 | 0 |
| $x_9$ | 0.37 | 0.22 | 0.45 | 0.56 | 0.50 | 0.32 | 0.44 | 0.64 | 0.52 | 1 |
| $x_{10}$ | 0.17 | 0.31 | 0.33 | 0.37 | 0.37 | 0.60 | 0.94 | 0.33 | 0.26 | 0 |

**Proposition 3.** *By merging $IS'$ and $IS$ according to Definition 6, we obtain $\widehat{IS} = (\widehat{U}, \widehat{A} \cup D, \widehat{V}, \widehat{f})$. Additionally, for $G = U' - U = \{g_1, g_2, \ldots, g_v\}$ and $H = U' \cap U = \{h_1, h_2, \ldots, h_w\}$, by utilizing $\mathcal{NM}_B^\delta(U,U) = \left[nm_{ij}\right]_{i\le p, j\le p}$, $\mathcal{NM}_B^\delta(G,U) = \left[nm'_{ij}\right]_{i\le v, j\le p}$ and $\mathcal{NM}_B^\delta(H,U) = \left[nm''_{ij}\right]_{i\le w, j\le p}$, we can dynamically compute $\mathcal{NM}_B^\delta(\widehat{U},\widehat{U})$ as follows:*

$$\mathcal{NM}_B^\delta(\widehat{U},\widehat{U}) = \left[\widehat{nm}_{ij}\right]_{i\le p+v, j\le p+v}, \widehat{nm}_{ij} = \begin{cases} nm'_{ij}, & x_i \in G \vee x_j \in G, \\ nm''_{ij}, & x_i \in H \vee x_j \in H, \\ nm_{ij}, & \text{others.} \end{cases}$$

**Example 7.** From Table 1, Table 2 and Table 3, there are three information system $IS = (U, A \cup D, V, f)$, $IS' = (U', A' \cup D, V', f')$ and $\widehat{IS} = (\widehat{U}, \widehat{A} \cup D, \widehat{V}, \widehat{f})$, where $G = U' - U = \{x_9, x_{10}\}$, $H = \{x_1, x_2, x_6, x_7\}$, and $\delta$ is 0.5.

According to Example 3, the neighborhood matrix is $\mathcal{NM}_B^\delta(U,U)$. Now, Computing $\mathcal{NM}_B^\delta(G,\widehat{U})$ and $\mathcal{NM}_B^\delta(H,\widehat{U})$ via Definition 3

$$\mathcal{NM}_B^\delta(G,\widehat{U}) = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix},$$

$$\mathcal{NM}_B^\delta(H,\widehat{U}) = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

Finally, according to Proposition 3, $\mathcal{NM}_B^\delta(\widehat{U},\widehat{U})$ can be computed by $\mathcal{NM}_B^\delta(U,U)$, $\mathcal{NM}_B^\delta(G,\widehat{U})$ and $\mathcal{NM}_B^\delta(H,\widehat{U})$

$$\mathcal{NM}_B^\delta(\widehat{U},\widehat{U}) = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

By using Algorithm 3, We can dynamically calculate the neighborhood matrix. Initially, we generate a key using a hash function, retrieve the neighborhood matrix, and an increment counter from the cache. If the neighborhood matrix does not exist, we statically compute $\mathcal{NM}_B^\delta(\widehat{U},\widehat{U})$. Subsequently, we save it into the cache along with the increment counter.

If the neighborhood matrix exists and the increment counter does not match the one in the cache, we retrieve $G$ and $H$ from the cache—representing samples that have been updated in the information system. These sets are then used to update the neighborhood matrix in the cache accordingly. Subsequently, we compute $\mathcal{NM}_B^\delta(G,\widehat{U})$ and $\mathcal{NM}_B^\delta(H,\widehat{U})$, further updating the neighborhood matrix to $\mathcal{NM}_B^\delta(\widehat{U},\widehat{U})$.

Finally, in other situations where the increment counter and the $\mathcal{NM}_B^\delta(U,U)$ in the cache match the corresponding values for $\mathcal{NM}_B^\delta(\widehat{U}, \widehat{U})$, the result can be returned directly. This algorithm significantly reduces the time complexity of computing $\mathcal{NM}_B^\delta(\widehat{U},\widehat{U})$ by partially updating the neighborhood matrix stored in the cache. The time complexity of this approach is $O(|G||\widehat{U}| + |H||\widehat{U}|)$.

After testing, we observed that the majority of the time spent on Algorithm 2 is dedicated to calculating $SM$ and $IM$. Thus, by leveraging multiple processes to compute Proposition 2, we can significantly enhance the algorithm's efficiency.

As illustrated in Fig. 2, the attribute $a$ to be computed is initially placed in the input queue. When a process in the pool becomes available, it is assigned to compute either $IM$ or $SM$ for the attribute $a_x$ using the idle process $process_y$. Upon completion of the computation, the result is placed in the output queue, and process $process_y$ is marked as idle and returned to the pool. This parallel computation continues until the input queue is empty. Once all computations are complete, the results are dequeued from the output queue sequentially, finalizing the parallel computation of $IM$ and $SM$.

Algorithm 4 is a parallel feature selection algorithm for the local neighborhood rough set with composite entropy designed for dynamic data. Initially, the selected features $B$ are initialized. If it is the first time training the model, the increment counter $t$ is initialized to record the count of model training. Otherwise, it is incremented by 1. Next, the original information system $IS = (U, A \cup D, V, f)$ is merged with the incremental information system $IS' = (U', A' \cup D, V', f')$ to obtain the updated information system $\widehat{IS} = (\widehat{U}, \widehat{A} \cup D, \widehat{V}, \widehat{f})$. Simultaneously, $G$ and $H$ are computed and stored in the cache, representing added data and updated data, respectively.

If $CE(B', D)$ does not equal $CE(A, D)$, the selected features are recalculated; otherwise, $B'$ remains as the selected features. Finally, attributes whose deletion does not affect composite entropy are removed. Notably, steps 11 to 18, steps 20 to 24, and steps 31 to 37 can be computed in parallel. The time complexity of Algorithm 4 is $O((|\widehat{A}| + |\widehat{U}||\widehat{A}| + |\widehat{B}|)(|G| + |H|)|\widehat{U}|)$.

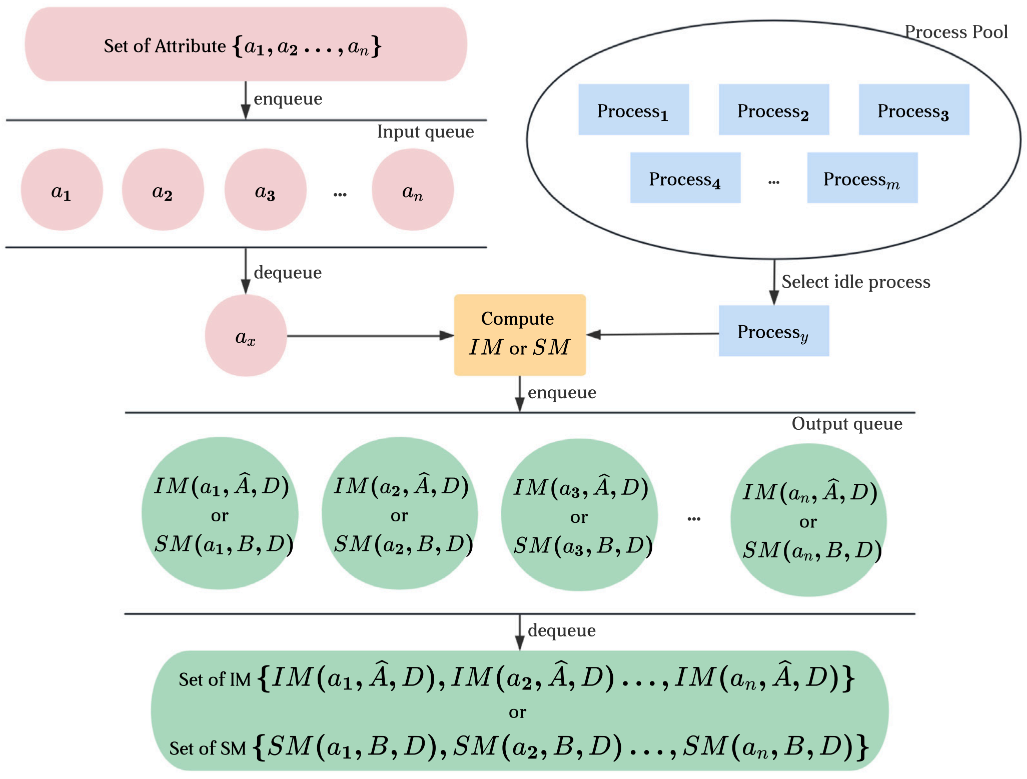
Set of Attribute $\{a_1, a_2 \dots, a_n\}$
enqueue
Input queue
$a_1$
$a_2$
$a_3$
$a_n$
dequeue
$a_x$
Compute $IM$ or $SM$
Process Pool
$\text{Process}_1$
$\text{Process}_2$
$\text{Process}_3$
$\text{Process}_4$
$\text{Process}_m$
Select idle process
$\text{Process}_y$
enqueue
Output queue
$IM(a_1, \hat{A}, D)$ or $SM(a_1, B, D)$
$IM(a_2, \hat{A}, D)$ or $SM(a_2, B, D)$
$IM(a_3, \hat{A}, D)$ or $SM(a_3, B, D)$
$IM(a_n, \hat{A}, D)$ or $SM(a_n, B, D)$
dequeue
Set of IM $\{IM(a_1, \hat{A}, D), IM(a_2, \hat{A}, D) \dots, IM(a_n, \hat{A}, D)\}$
or
Set of SM $\{SM(a_1, B, D), SM(a_2, B, D) \dots, SM(a_n, B, D)\}$

**Fig. 2.** Parallel schematic diagram.

**Algorithm 3:** Calculate Neighborhood Matrix Dynamically

```
Input: The updated information system $\widehat{IS} = (\hat{U}, \hat{A} \cup D, \hat{V}, \hat{f})$, the
       subset of samples $B$,the neighbor radius $\delta$ and the increment
       counter $t$.
Output: Neighborhood matrix $\mathcal{NM}_B^\delta(\hat{U}, \hat{U})$.
1  Initialize $key = hash(B)$;
2  Get $(\mathcal{NM}_B^\delta(U, U), t_{cache})$ from cache via using $key$;
3  if $\mathcal{NM}_B^\delta(U, U)$ not in cache then
4      Calculate neighborhood matrix $\mathcal{NM}_B^\delta(\hat{U}, \hat{U})$ via Definition 3;
5      Save key–value pair $(key, (\mathcal{NM}_B^\delta(\hat{U}, \hat{U}), t))$ in cache;
6  else if $t \neq t_{cache}$ then
7      Initialize $G = \emptyset, H = \emptyset$;
8      for $t_{key} = t_{cache} + 1$ to $t$ do
9          Get $(G_{cache}, H_{cache})$ from cache via using $t_{key}$;
10         $G = G \cup G_{cache}$;
11         $H = H \cup H_{cache}$;
12     end
13     if $G \neq \emptyset$ then
14         Compute local neighborhood matrix $\mathcal{NM}_B^\delta(G, \hat{U})$ via
           Definition 3;
15     end
16     if $H \neq \emptyset$ then
17         Compute local neighborhood matrix $\mathcal{NM}_B^\delta(H, \hat{U})$ via
           Definition 3;
18     end
19     Compute $\mathcal{NM}_B^\delta(\hat{U}, \hat{U})$ with $\mathcal{NM}_B^\delta(U, U)$, $\mathcal{NM}_B^\delta(G, \hat{U})$ and
         $\mathcal{NM}_B^\delta(H, \hat{U})$ via Proposition 3;
20     Save key–value pair $(key, (\mathcal{NM}_B^\delta(\hat{U}, \hat{U}), t))$ in cache;
21 else
22     $\mathcal{NM}_B^\delta(\hat{U}, \hat{U}) = \mathcal{NM}_B^\delta(U, U)$;
23 end
24 return $\mathcal{NM}_B^\delta(\hat{U}, \hat{U})$;
```

## 5. Experimental results and analysis

### 5.1. Experimental preparation

In this section, we selected 12 datasets for our experiments, all of which are listed in Table 4 and are available from the UCI.

During data preprocessing, we applied the sigmoid function to normalize the data to the range $(0, 1)$. Any missing values were replaced with 0, with the 0 value serving as a special marker to differentiate between missing and non-missing entries. The sigmoid function used is expressed as follows, where $\mu$ represents the mean and $\sigma$ denotes the standard deviation:

$$y_i = \frac{1}{1 + e^{-\frac{x_i - \mu}{\sigma}}}.$$

We employed ten-fold cross-validation in all experiments to evaluate different feature subsets. Each dataset was divided into ten equal portions, with seven used as the training set for the classifier model and the remaining three as the test set.

All algorithms discussed in this paper were implemented in Python using the Anaconda Navigator environment. The computations were performed on a computer equipped with an AMD Ryzen 7 5700G CPU (3.80 GHz), 64.0 GB of memory, and running a 64-bit version of Windows 11.

### 5.2. Experimental design

In this subsection, we design an experiment to evaluate the performance and efficiency of our proposed algorithm. Specifically, we compare its execution with 1 process and 8 processes. Additionally, we assess it against 7 feature selection algorithms, 3 of which are based on rough sets. For the PMLCE-S and PMLCE-D algorithms, the neighbor radius $\delta$ is varied from 0.01 to 0.5, with a step size of 0.01.

The detailed information and parameters of the 7 algorithms is as follows:

**Algorithm 4:** PMLCE-D: Parallel Local Neighborhood Rough Set with Composite Entropy for Dynamic Data

```
Input: Original information system $IS = (U, A \cup D, V, f)$, the neighbor
       radius $\delta$, the incremental information system
       $IS' = (U', A' \cup D, V', f')$, last selected attributes $B'$ and
       increment counter $t$.
Output: Selected features $B$.
Initialize $B = \emptyset$;
if it is the first time to train then
    Initialize increment counter $t = 0$;
else
    $t = t + 1$;
end
Merge $IS$ and $IS'$ via Definition 6, and the updated information
  system is $\widehat{IS} = (\widehat{U}, \widehat{A} \cup D, \widehat{V}, \widehat{f})$;
Get $G$ and $H$ via Proposition 3;
Save key–value pair $(t, (G, H))$ in cache;
if $CE(B', D) \neq CE(\widehat{A}, D)$ then        /* Use Algorithm 3 to
  calculate the neighborhood matrix */
    do in parallel
        for $a \in \widehat{A}$ do
            Compute $im = IM(a, \widehat{A}, D)$ via Proposition 2 and
              Algorithm 3;
            if $im > 0$ then
                $B \leftarrow a$;
            end
        end
    end
    while $CE(B, D) \neq CE(\widehat{A}, D)$ do     /* Use Algorithm 3 to
      calculate the neighborhood matrix */
        do in parallel
            for $a \in (\widehat{A} - B)$ do
                Compute $sm = SM(a, B, D)$ via Proposition 2 and
                  Algorithm 3;
            end
        end
        $a_{max} = \arg\max_{a_{max} \in \widehat{A} - B}(sm)$;
        $B \leftarrow a_{max}$;
    end
else
    $B = B'$;
end
do in parallel
    for $b \in B$ do
        if $CE(B - \{b\}, D) = CE(B, D)$ then /* Use Algorithm 3 to
          calculate the neighborhood matrix */
            $B = B - \{b\}$;
        end
    end
end
return $B$;
```

(1) **ICA** [29]: Independent Component Analysis (ICA) is a computational technique employed to segregate a multivariate signal into additive subcomponents. Widely applied in signal processing, image processing, and machine learning, ICA is instrumental for dimensionality reduction and feature selection. In this experiment, all components are utilized.

(2) **Isomap** [30]: Isomap is a nonlinear method for reducing dimensionality, which is used to calculate a low-dimensional embedding of a set of high-dimensional data points. Among several widely used low-dimensional embedding methods, Isomap stands out. For this experiment, we specifically configure $n\_components = 2$.

(3) **PCA** [31]: Principal Component Analysis (PCA) is a widely adopted technique for analyzing extensive datasets characterized by a high number of dimensions or features per observation. Its popularity stems from its ability to improve data interpretability while retaining the maximum amount of information. In the context of this experiment, we specifically choose attributes whose cumulative contribution rate surpasses 0.8.

(4) **Relief** [32]: Relief is an algorithm for feature weighting that assigns varying weights to features based on their correlation with the class. It eliminates features with weights below a specified threshold. In this experiment, we specifically choose components whose cumulative contribution rate exceeds 0.8.

(5) **LFSACIE** [17]: LFSACIE is an attribute reduction algorithm based on composite entropy and local neighborhood rough set. In this experiment, we set the neighborhood radius $\delta = 0.01$ and $\beta = 0.1$.

(6) **LARD** [5]: LARD is an attribute reduction algorithm based on local rough set. The fundamental concept is to ensure that the certainty of the rough set model is at least as much as that of the original model. In this experiment, we set the neighborhood radius $\delta = 0.01$.

(7) **LFSASI** [33]: LFSASI is an attribute reduction algorithm based on self-information and local rough set. In this experiment, the neighborhood radius are $\delta = 0.3$ and $\beta = 0.1$.

To simulate the incremental environment of constantly adding and updating data in reality, we divided the dataset into four parts to simulate data added at four different times. Then, the feature selection results of these four moments were tested separately, and their running time was recorded.

At time $t_0$, we will use the first part of the data to train the model. At $t_1$, before using the second part of the data, we will add the original form of the replaced data into the second data to simulate the situation of data update in reality. This process is repeated at $t_2$. Finally, at $t_3$, we will train models using the original form of the replaced data, which has never been trained in $t_1$ and $t_2$.

After completing feature selection, we measured the classification accuracy using 4 classification algorithms. These 4 algorithms are k-Nearest Neighbor (KNN), Random Forest (RF), Support Vector Classification (SVC), and eXtreme Gradient Boosting (XGBoost, XGB), respectively. The parameters are set as follows: neighbors of KNN is set to 5, the number of trees in RF is set to 100, the degree and regularization parameter of SVC are set to 3 and 1.0, respectively, and the learning rate of XGBoost is set to 0.3.

## 5.3. Experiments of accuracy

In this subsection, we will discuss the performance of the feature extraction algorithm. We have already described the preparation and design of the experiment in Section 5.1 and Section 5.2.

Table 5 provides the $\delta$ values of PMLCE-S and PMLCE-D when the final mean accuracy is highest (accuracy of $t_3$).

Table 6 presents the highest final mean accuracy and its standard deviation for different classifiers after feature selection, with the number of selected features indicated in parentheses. According to the experimental results, we can see that the two algorithms we proposed have shown very good results on data sets except Turkiye and Quality, and have shown significant improvements compared to other feature extraction algorithms based on rough sets, which proves that the algorithms we proposed are effective.

Additionally, in Fig. 3, the $x$-axis represents the feature extraction algorithm, while the $y$-axis represents the dataset. Sub-figures (a), (e), (i), and (m) depict the classification accuracy of KNN, Random Forest, SVC, and XGBoost, respectively, after each feature selection algorithm has performed feature extraction on each dataset at time $t_0$. Similarly, sub-figures (b), (f), (j), and (n) illustrate the classification accuracy of algorithms KNN, Random Forest, SVC, and XGBoost at time $t_1$, following the same process. It is worth noting that PMLCE-S and PMLCE-D may yield different results at time $t_0$ due to different

**Table 4**
Details of datasets.

| | Dataset | Samples | Attributes | Classes | | Dataset | Samples | Attributes | Classes |
|---|---|---|---|---|---|---|---|---|---|
| I | Cancer | 116 | 9 | 2 | VII | Spambase | 4601 | 57 | 2 |
| II | Toxicity | 171 | 1203 | 2 | VIII | Quality | 4898 | 11 | 7 |
| III | Darwin | 174 | 450 | 2 | IX | Turkiye | 5820 | 32 | 3 |
| IV | Wine | 178 | 13 | 3 | X | Grid | 10 000 | 13 | 2 |
| V | Hill | 606 | 100 | 2 | XI | Bean | 13 611 | 16 | 7 |
| VI | Chess | 3196 | 36 | 2 | XII | Telescope | 19 020 | 10 | 2 |

**Table 5**
Delta of PMLCE-S and PMLCE-D.

| | | PMLCE-S | PMLCE-D | | | PMLCE-S | PMLCE-D | | | PMLCE-S | PMLCE-D |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Cancer | KNN | 0.11 | 0.13 | Hill | KNN | 0.29 | 0.02 | Turkiye | KNN | 0.01 | 0.01 |
| | RF | 0.09 | 0.10 | | RF | 0.32 | 0.19 | | RF | 0.03 | 0.02 |
| | SVC | 0.11 | 0.13 | | SVC | 0.31 | 0.29 | | SVC | 0.04 | 0.04 |
| | XGB | 0.18 | 0.18 | | XGB | 0.08 | 0.14 | | XGB | 0.07 | 0.07 |
| Toxicity | KNN | 0.48 | 0.30 | Chess | KNN | 0.01 | 0.01 | Grid | KNN | 0.01 | 0.15 |
| | RF | 0.08 | 0.07 | | RF | 0.38 | 0.01 | | RF | 0.12 | 0.13 |
| | SVC | 0.25 | 0.25 | | SVC | 0.48 | 0.48 | | SVC | 0.01 | 0.15 |
| | XGB | 0.07 | 0.07 | | XGB | 0.43 | 0.43 | | XGB | 0.07 | 0.19 |
| Darwin | KNN | 0.29 | 0.29 | Spambase | KNN | 0.10 | 0.10 | Bean | KNN | 0.04 | 0.03 |
| | RF | 0.41 | 0.34 | | RF | 0.11 | 0.43 | | RF | 0.02 | 0.04 |
| | SVC | 0.40 | 0.40 | | SVC | 0.48 | 0.48 | | SVC | 0.02 | 0.04 |
| | XGB | 0.28 | 0.28 | | XGB | 0.37 | 0.37 | | XGB | 0.03 | 0.05 |
| Wine | KNN | 0.21 | 0.30 | Quality | KNN | 0.09 | 0.09 | Telescope | KNN | 0.01 | 0.03 |
| | RF | 0.30 | 0.36 | | RF | 0.09 | 0.08 | | RF | 0.03 | 0.03 |
| | SVC | 0.38 | 0.40 | | SVC | 0.04 | 0.06 | | SVC | 0.03 | 0.03 |
| | XGB | 0.37 | 0.33 | | XGB | 0.09 | 0.09 | | XGB | 0.03 | 0.03 |

parameter choices. However, as indicated in Table 5, if PMLCE-S and PMLCE-D use the same parameters, their accuracy at time $t_0$ in Fig. 3 will also be identical.

Simultaneously, to assess whether there is a significant difference in classification performance between different algorithms, we employ the Wilcoxon rank sum test to compare the experimental results. With a test threshold set at 0.1, we observe that all the test P-values are smaller than the threshold, as indicated in Table 7. Therefore, we can reject the null hypothesis and conclude that there is a significant difference between PMLCE-S and PMLCE-D compared to other feature selection algorithms.

## 5.4. *Experiments of efficiency*

In this subsection, we will discuss the efficiency of the feature selection algorithm.

We used 1 process for PMLCE-S and 8 processes for PMLCE-D, comparing them with three other rough set-based feature selection algorithms: LFSACIE, LARD, and LFSASI. Since our proposed algorithms are fundamentally different from feature selection methods that do not rely on Rough Set Theory, comparing their running speed with non-rough set algorithms would not accurately reflect the contribution of our work to rough set-based feature selection techniques. Therefore, we selected only the three aforementioned algorithms for comparison and analysis.

At the same time, since parallel computing is not necessary for small data sets, we only analyze the results of the algorithm on data sets with more than 1000 samples, which are Chess, Spambase, Quality, Turkiye, Grid, Bean, and Telescope.

Other details about the experiment have been described in Section 5.1 and Section 5.2.

In Table 8, we present the cumulative running time of each model at four different moments during the dynamic training process. It is evident that PMLCE-S and PMLCE-D are significantly faster than other rough set algorithms. Additionally, when run with the same number of processes, PMLCE-D, being a dynamic algorithm, is much faster than PMLCE-S, which is static. For both PMLCE-S and PMLCE-D, running them in parallel with multiple processes is also much faster than running them in a non-parallel fashion.

In Fig. 4, we illustrate the speed-up ratio of feature selection algorithms, defined as:

$$\text{Speed-up ratio} = \frac{\text{Consuming time of PMLCE-D 1 process}}{\text{Consuming time of other algorithm}}.$$

It is evident that the speed of PMLCE-D with 8 processes has increased by 2 to 5 times compared to PMLCE-S with 1 process. This phenomenon becomes more apparent as the number of samples in the dataset increases.

In conclusion, both PMLCE-S and PMLCE-D are efficient feature selection algorithms. Comparing PMLCE-S with PMLCE-D, the latter can be dynamically trained, resulting in faster performance. By successfully transforming PMLCE-S and PMLCE-D into parallel computing algorithms, we enhance the utilization of computing resources and significantly improve training speed.

## 5.5. *Summary of experiments*

The experiments on accuracy demonstrate that the proposed feature extraction algorithms, PMLCE-S and PMLCE-D, perform exceptionally well on most datasets, except for Turkiye and Quality, showing significant improvements compared to other rough set-based algorithms. The highest mean accuracy and standard deviation for different classifiers after feature selection are presented, and the difference in initial accuracy between PMLCE-S and PMLCE-D is attributed to different parameter choices. However, if the same parameters are selected, their initial accuracy will be identical.

In terms of efficiency, PMLCE-S and PMLCE-D were evaluated using both 1 and 8 processes, and their performance was compared to other rough set-based algorithms on larger datasets. PMLCE-D, being dynamic, was significantly faster than PMLCE-S, especially when parallelized. The speed-up ratio of PMLCE-D with 8 processes showed an increase of 2 to 5 times compared to PMLCE-S with 1 process. Overall, both algorithms proved to be highly efficient, with PMLCE-D demonstrating faster performance and better resource utilization through parallel computing.

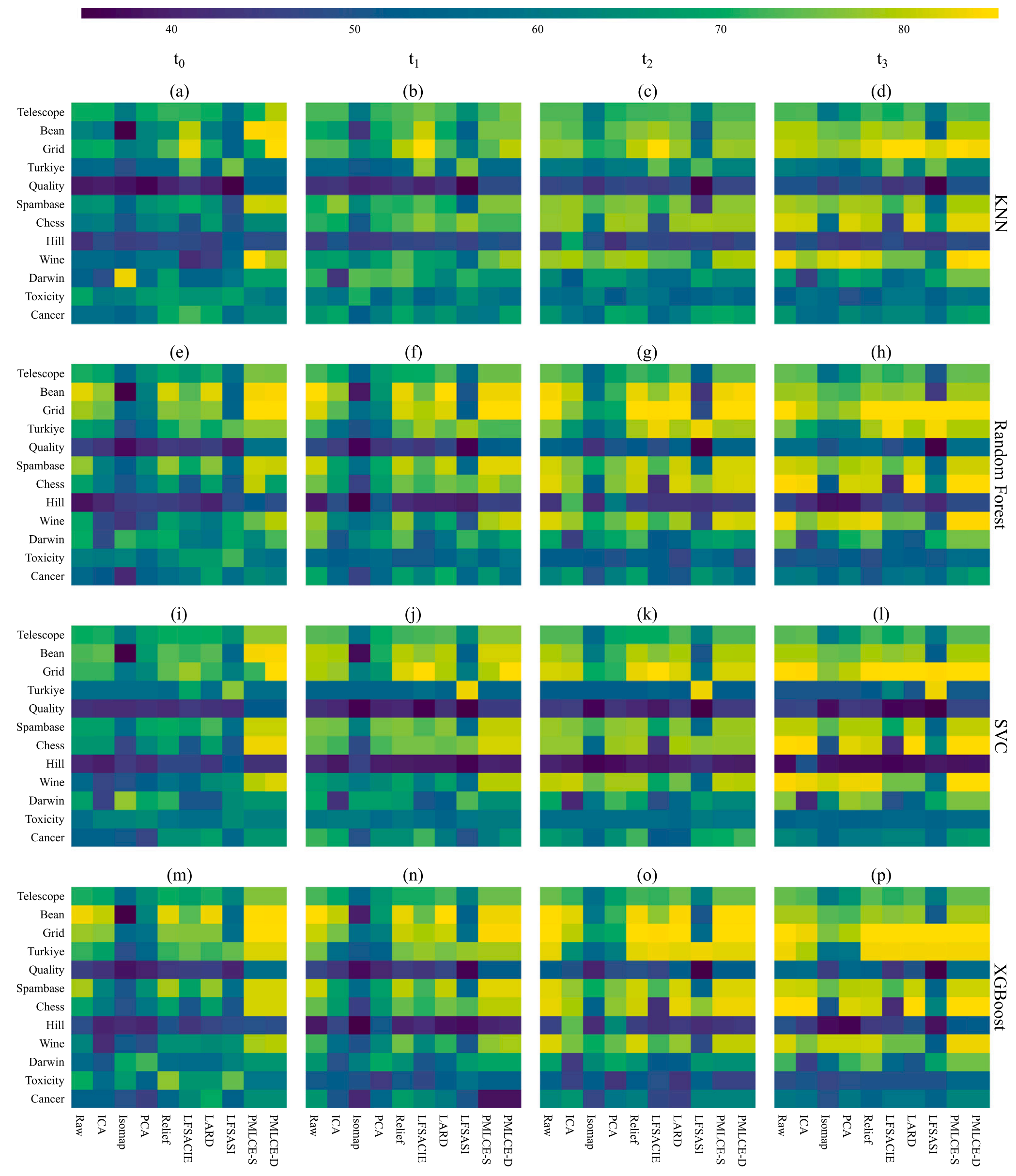

**Fig. 3.** Accuracy (%) of algorithms at each moment.

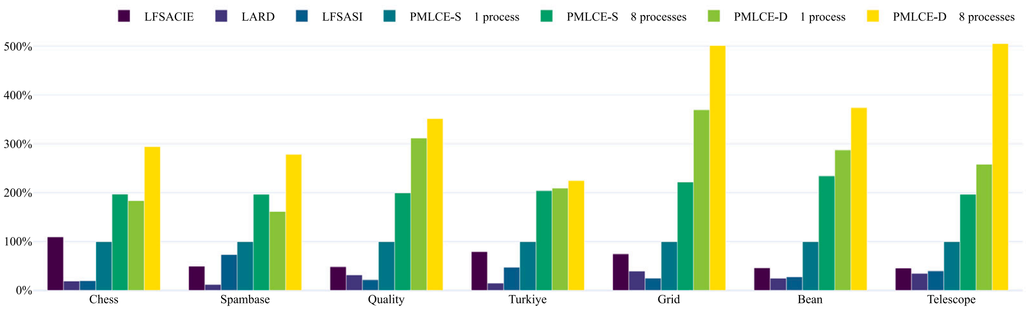

**Fig. 4.** Speed-up ratio (%) of algorithms based on rough set.

Based on the above experiments, we can conclude that both PMLCE-S and PMLCE-D, being parallel algorithms, outperform other algorithms in terms of accuracy and efficiency. This advantage becomes particularly pronounced when dealing with large datasets, showcasing the benefits of parallel algorithms, especially in the context of training substantial data. Additionally, the dynamic nature of PMLCE-D results in significant time savings, thereby realizing an efficient feature selection algorithm.

**Table 6**
Accuracy (mean ± std. %) of algorithms at last moment.

| | | Raw | ICA | Isomap | PCA | Relief | LFSACIE | LARD | LFSASI | PMLCE-S | PMLCE-D |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Cancer | KNN | 75.76 ± 13.29(9) | 65.53 ± 15.51(9) | 64.70 ± 13.90(2) | 73.03 ± 12.61(5) | 69.02 ± 12.37(6) | 72.27 ± 11.89(2) | 72.27 ± 11.89(2) | 65.76 ± 9.77(1) | 77.35 ± 15.94(6) | **79.32 ± 12.63(6)** |
| | RF | 73.33 ± 10.98(9) | 73.26 ± 11.79(9) | 65.53 ± 17.11(2) | 73.11 ± 14.23(5) | 62.20 ± 12.48(6) | 70.91 ± 12.37(2) | 71.89 ± 15.76(2) | 67.42 ± 7.87(1) | 77.65 ± 9.30(6) | **80.98 ± 9.24(6)** |
| | SVC | 75.83 ± 14.39(9) | 74.24 ± 12.08(9) | 66.44 ± 16.80(2) | 73.11 ± 15.56(5) | 73.26 ± 11.79(6) | 74.92 ± 12.17(2) | 74.92 ± 12.17(2) | 67.12 ± 16.36(1) | 79.17 ± 15.32(6) | **80.23 ± 9.67(6)** |
| | XGB | 72.42 ± 9.16(9) | 71.36 ± 14.51(9) | 61.14 ± 16.05(2) | 72.42 ± 7.72(5) | 66.36 ± 10.86(6) | 71.59 ± 8.95(2) | 71.59 ± 8.95(2) | 66.44 ± 5.77(1) | **75.83 ± 9.02(8)** | **75.83 ± 9.02(8)** |
| Toxicity | KNN | 58.59 ± 13.65(1203) | 67.25 ± 2.93(171) | 64.41 ± 16.20(2) | 55.69 ± 15.31(22) | 58.56 ± 13.58(761) | 67.84 ± 5.64(2) | 67.84 ± 5.64(2) | 60.88 ± 9.20(11) | **67.91 ± 9.29(30)** | 67.29 ± 9.49(12) |
| | RF | 61.44 ± 7.68(1203) | 62.55 ± 8.03(171) | 61.99 ± 10.42(2) | 64.41 ± 7.90(22) | 59.77 ± 11.47(761) | 65.52 ± 10.03(2) | 64.31 ± 10.57(2) | 62.58 ± 9.62(11) | **69.64 ± 10.06(5)** | 68.46 ± 10.97(5) |
| | SVC | 65.52 ± 4.01(1203) | 67.25 ± 2.93(171) | 66.67 ± 2.77(2) | 65.52 ± 5.61(22) | 66.11 ± 4.30(761) | 67.25 ± 2.93(2) | 67.25 ± 2.93(2) | 66.11 ± 4.30(11) | **68.43 ± 4.84(11)** | **68.43 ± 4.84(11)** |
| | XGB | 60.29 ± 10.31(1203) | 61.96 ± 10.18(171) | 57.94 ± 10.00(2) | 60.88 ± 9.20(22) | 61.47 ± 11.13(761) | 61.44 ± 14.85(2) | 61.44 ± 14.85(2) | 61.37 ± 9.78(11) | **67.35 ± 8.60(5)** | **67.35 ± 8.60(5)** |
| Darwin | KNN | 75.29 ± 10.33(450) | 50.07 ± 11.24(174) | 71.37 ± 9.48(2) | 80.49 ± 8.23(49) | 77.09 ± 10.79(315) | 63.17 ± 14.73(2) | 67.32 ± 11.10(3) | 78.63 ± 10.17(11) | **87.94 ± 7.80(11)** | **87.94 ± 7.80(11)** |
| | RF | 89.58 ± 5.41(450) | 56.50 ± 15.75(174) | 71.34 ± 9.20(2) | 83.99 ± 8.38(49) | 87.84 ± 9.27(315) | 66.08 ± 12.35(2) | 63.01 ± 14.16(3) | 83.76 ± 11.04(11) | 89.54 ± 9.87(15) | **89.61 ± 8.51(13)** |
| | SVC | 90.78 ± 5.53(450) | 51.18 ± 2.06(174) | 75.75 ± 9.25(2) | **91.90 ± 6.18(49)** | 89.61 ± 7.03(315) | 60.95 ± 8.73(2) | 66.05 ± 5.96(3) | 82.09 ± 9.86(11) | 89.61 ± 6.58(13) | 89.61 ± 6.58(13) |
| | XGB | 85.59 ± 9.62(450) | 56.47 ± 11.75(174) | 69.64 ± 11.30(2) | 84.97 ± 7.42(49) | 89.05 ± 7.36(315) | 67.32 ± 13.54(2) | 62.42 ± 12.48(3) | 81.44 ± 9.92(11) | **89.08 ± 5.61(8)** | **89.08 ± 5.61(8)** |
| Wine | KNN | 96.67 ± 5.97(13) | 90.46 ± 7.43(13) | 95.00 ± 6.11(2) | 96.67 ± 5.37(5) | 95.00 ± 4.86(9) | 87.19 ± 8.27(2) | 87.19 ± 8.27(2) | 69.15 ± 9.46(1) | 97.78 ± 3.88(6) | **99.44 ± 1.76(7)** |
| | RF | 98.33 ± 3.75(13) | 89.90 ± 4.37(13) | 95.00 ± 6.11(2) | 96.67 ± 7.03(5) | 97.78 ± 3.88(9) | 88.86 ± 6.90(2) | 89.41 ± 7.12(2) | 60.69 ± 8.18(1) | **98.89 ± 2.34(7)** | **98.89 ± 3.51(9)** |
| | SVC | 98.33 ± 3.75(13) | 97.22 ± 4.72(13) | 95.56 ± 5.74(2) | 97.22 ± 5.40(5) | 97.78 ± 2.87(9) | 88.86 ± 7.83(2) | 88.86 ± 7.83(2) | 69.25 ± 10.96(1) | **99.44 ± 1.76(10)** | **99.44 ± 1.76(11)** |
| | XGB | 96.08 ± 6.96(13) | 89.97 ± 7.75(13) | 95.00 ± 5.52(2) | 95.00 ± 7.15(5) | 95.56 ± 6.83(9) | 88.89 ± 8.69(2) | 88.89 ± 8.69(2) | 63.56 ± 8.10(1) | **97.78 ± 3.88(10)** | **97.78 ± 3.88(9)** |
| Hill | KNN | 51.63 ± 4.64(100) | **56.13 ± 7.11(100)** | 50.13 ± 7.07(2) | 48.15 ± 7.57(1) | 50.33 ± 4.48(64) | 54.27 ± 7.52(4) | 49.66 ± 5.29(86) | 48.96 ± 7.75(1) | 54.43 ± 5.85(29) | 53.77 ± 4.57(66) |
| | RF | 58.23 ± 7.71(100) | 52.82 ± 4.15(100) | 48.01 ± 10.15(2) | 48.66 ± 6.81(1) | 54.13 ± 5.05(64) | 58.89 ± 7.96(4) | 55.92 ± 6.43(86) | 51.65 ± 5.55(1) | 59.04 ± 7.38(37) | **59.22 ± 7.18(93)** |
| | SVC | 47.53 ± 4.77(100) | **61.23 ± 6.39(100)** | 48.36 ± 3.77(2) | 47.20 ± 4.57(1) | 47.03 ± 5.78(64) | 46.87 ± 4.53(4) | 47.20 ± 5.27(86) | 48.86 ± 4.69(1) | 47.87 ± 4.82(51) | 48.36 ± 3.85(25) |
| | XGB | 60.88 ± 5.03(100) | 55.46 ± 3.59(100) | 48.00 ± 6.37(2) | 46.52 ± 6.43(1) | 55.29 ± 5.18(64) | 56.41 ± 6.75(4) | 61.03 ± 5.29(86) | 48.50 ± 6.62(1) | 62.69 ± 5.41(67) | **63.51 ± 6.38(65)** |
| Chess | KNN | 95.21 ± 1.25(36) | 94.59 ± 1.23(36) | 64.11 ± 1.82(2) | 95.31 ± 1.07(15) | 92.99 ± 1.14(9) | 51.82 ± 1.38(1) | **96.06 ± 1.31(29)** | 75.72 ± 1.62(4) | **96.06 ± 1.31(29)** | **96.06 ± 1.31(29)** |
| | RF | **99.22 ± 0.68(36)** | 98.37 ± 0.64(36) | 62.89 ± 2.59(2) | 96.25 ± 1.17(15) | 95.59 ± 1.64(9) | 52.22 ± 0.10(1) | 99.06 ± 0.57(29) | 75.72 ± 1.62(4) | 99.15 ± 0.68(30) | 99.16 ± 0.55(29) |
| | SVC | 98.81 ± 0.69(36) | 98.65 ± 0.86(36) | 60.76 ± 2.52(2) | 96.71 ± 0.84(15) | 95.43 ± 1.51(9) | 52.22 ± 0.10(1) | 98.78 ± 0.70(29) | 75.69 ± 1.61(4) | **98.87 ± 0.71(34)** | **98.87 ± 0.71(34)** |
| | XGB | 99.41 ± 0.52(36) | 98.94 ± 0.57(36) | 62.42 ± 2.96(2) | 96.50 ± 1.06(15) | 95.49 ± 1.59(9) | 52.22 ± 0.10(1) | 99.44 ± 0.60(29) | 75.66 ± 1.59(4) | **99.50 ± 0.51(31)** | **99.50 ± 0.51(31)** |
| Spambase | KNN | 91.46 ± 1.17(57) | 90.46 ± 1.13(57) | 87.29 ± 0.78(2) | 91.50 ± 0.89(25) | 91.18 ± 1.15(32) | 82.87 ± 1.85(6) | 91.52 ± 1.74(33) | 65.05 ± 2.18(2) | **92.02 ± 1.08(45)** | **92.02 ± 1.08(45)** |
| | RF | 95.52 ± 0.78(57) | 94.87 ± 0.83(57) | 89.24 ± 1.22(2) | 93.39 ± 0.98(25) | 94.94 ± 0.90(32) | 87.57 ± 1.72(6) | 95.09 ± 1.17(33) | 66.25 ± 2.31(2) | 95.50 ± 1.05(45) | **95.54 ± 0.93(55)** |
| | SVC | 94.37 ± 1.05(57) | 94.18 ± 0.81(57) | 87.05 ± 1.31(2) | 93.52 ± 0.91(25) | 93.20 ± 1.01(32) | 82.22 ± 0.87(6) | 93.89 ± 1.13(33) | 66.07 ± 2.80(2) | **94.44 ± 0.93(56)** | **94.44 ± 0.93(56)** |
| | XGB | **95.61 ± 0.90(57)** | 94.98 ± 1.03(57) | 88.15 ± 0.79(2) | 93.76 ± 0.75(25) | 94.57 ± 0.81(32) | 86.53 ± 1.71(6) | 94.98 ± 0.79(33) | 66.22 ± 2.12(2) | **95.61 ± 0.90(56)** | **95.61 ± 0.90(56)** |
| Quality | KNN | 56.14 ± 2.76(11) | **56.23 ± 2.18(11)** | 49.98 ± 2.01(2) | 55.80 ± 1.82(6) | 53.61 ± 2.59(7) | 47.63 ± 2.66(3) | 49.96 ± 2.64(5) | 37.95 ± 2.91(2) | 56.14 ± 2.76(11) | 56.14 ± 2.76(11) |
| | RF | 70.31 ± 1.64(11) | 70.38 ± 1.82(11) | 61.33 ± 1.89(2) | 69.07 ± 1.58(6) | 68.48 ± 1.76(7) | 60.51 ± 2.08(3) | 64.56 ± 1.54(5) | 45.10 ± 1.34(2) | 70.27 ± 2.11(11) | **70.66 ± 2.04(10)** |
| | SVC | 57.13 ± 2.76(11) | **57.66 ± 2.23(11)** | 47.69 ± 2.06(2) | 54.57 ± 1.89(6) | 53.80 ± 2.47(7) | 46.86 ± 1.24(3) | 47.20 ± 1.65(5) | 44.81 ± 0.17(2) | 57.53 ± 2.63(10) | 57.53 ± 2.63(10) |
| | XGB | 68.74 ± 1.88(11) | **69.17 ± 1.43(11)** | 56.04 ± 1.74(2) | 67.27 ± 1.53(6) | 65.52 ± 1.95(7) | 53.88 ± 1.99(3) | 60.94 ± 1.44(5) | 45.12 ± 1.21(2) | 68.74 ± 1.88(11) | 68.74 ± 1.88(11) |
| Turkiye | KNN | 70.45 ± 1.91(32) | 67.94 ± 1.52(32) | 66.24 ± 2.71(2) | 62.27 ± 1.64(3) | 71.92 ± 1.69(20) | **87.25 ± 1.51(2)** | 71.62 ± 1.77(18) | **87.25 ± 1.51(2)** | 72.11 ± 2.30(18) | 72.11 ± 2.30(18) |
| | RF | 90.12 ± 1.15(32) | 82.85 ± 1.33(32) | 72.96 ± 1.70(2) | 74.36 ± 1.12(3) | 92.85 ± 0.93(20) | **98.30 ± 0.57(2)** | 93.32 ± 0.86(18) | **98.30 ± 0.57(2)** | 93.73 ± 0.92(18) | 93.78 ± 0.77(18) |
| | SVC | 62.11 ± 0.27(32) | 62.04 ± 0.38(32) | 61.87 ± 0.05(2) | 61.87 ± 0.05(3) | 62.47 ± 0.44(20) | 74.74 ± 3.04(2) | 61.92 ± 0.17(18) | **97.59 ± 0.70(2)** | 62.34 ± 0.53(19) | 62.34 ± 0.53(19) |
| | XGB | 97.97 ± 0.62(32) | 93.42 ± 0.73(32) | 70.82 ± 1.81(2) | 71.55 ± 1.90(3) | 98.06 ± 0.65(20) | **98.30 ± 0.57(2)** | 98.02 ± 0.65(18) | **98.30 ± 0.57(2)** | 98.06 ± 0.71(30) | 98.06 ± 0.71(30) |
| Grid | KNN | 93.62 ± 0.53(13) | 92.67 ± 0.68(13) | 89.83 ± 0.78(2) | 91.48 ± 0.51(9) | 95.66 ± 0.31(8) | 99.05 ± 0.38(3) | **99.10 ± 0.20(3)** | 95.12 ± 0.78(10) | 98.33 ± 0.30(4) | 97.51 ± 0.38(5) |
| | RF | 99.98 ± 0.04(13) | 95.12 ± 0.64(13) | 89.23 ± 1.03(2) | 92.57 ± 0.91(9) | 99.98 ± 0.04(8) | 99.98 ± 0.04(3) | **99.99 ± 0.03(3)** | 99.98 ± 0.04(10) | **99.99 ± 0.03(8)** | 99.98 ± 0.04(5) |
| | SVC | 98.70 ± 0.27(13) | 98.57 ± 0.13(13) | 90.60 ± 1.14(2) | 94.30 ± 0.55(9) | 99.11 ± 0.23(8) | **99.70 ± 0.12(3)** | 99.69 ± 0.17(3) | 98.92 ± 0.24(10) | 99.54 ± 0.18(4) | 99.41 ± 0.19(5) |
| | XGB | 99.85 ± 0.18(13) | 97.63 ± 0.54(13) | 89.88 ± 0.88(2) | 93.33 ± 0.87(9) | 99.83 ± 0.20(8) | 99.84 ± 0.16(3) | 99.81 ± 0.14(3) | 99.87 ± 0.16(10) | **99.88 ± 0.15(6)** | 99.86 ± 0.12(8) |
| Bean | KNN | 92.24 ± 0.49(16) | 92.25 ± 0.43(16) | 87.05 ± 0.69(2) | 85.89 ± 0.82(2) | 92.13 ± 0.44(10) | 89.71 ± 0.56(4) | 90.91 ± 0.73(5) | 58.20 ± 1.20(2) | 92.28 ± 0.45(14) | **92.37 ± 0.44(11)** |
| | RF | 92.40 ± 0.46(16) | 92.69 ± 0.60(16) | 86.80 ± 0.56(2) | 85.79 ± 0.61(2) | 92.37 ± 0.58(10) | 91.01 ± 0.53(4) | 91.90 ± 0.51(5) | 55.08 ± 1.23(2) | **92.76 ± 0.50(6)** | 92.50 ± 0.46(12) |
| | SVC | 92.99 ± 0.58(16) | 92.98 ± 0.51(16) | 88.41 ± 0.67(2) | 87.74 ± 0.64(2) | 93.03 ± 0.54(10) | 91.18 ± 0.69(4) | 92.39 ± 0.52(5) | 63.35 ± 0.47(2) | **93.12 ± 0.61(6)** | 93.08 ± 0.54(12) |
| | XGB | 92.76 ± 0.49(16) | **92.84 ± 0.56(16)** | 87.18 ± 0.79(2) | 86.32 ± 0.59(2) | 92.60 ± 0.51(10) | 90.74 ± 0.51(4) | 91.78 ± 0.51(5) | 62.66 ± 0.96(2) | 92.79 ± 0.44(12) | 92.76 ± 0.49(16) |
| Telescope | KNN | 84.37 ± 0.62(10) | 84.66 ± 0.99(10) | 68.53 ± 0.97(2) | 76.82 ± 0.57(4) | 83.84 ± 0.81(7) | 81.60 ± 0.77(4) | 84.61 ± 0.71(6) | 67.42 ± 1.11(2) | **85.07 ± 0.66(6)** | 84.37 ± 0.62(10) |
| | RF | 88.20 ± 0.66(10) | 87.67 ± 0.92(10) | 69.68 ± 0.85(2) | 78.68 ± 0.61(4) | 87.49 ± 0.55(7) | 84.58 ± 0.79(4) | 87.00 ± 0.63(6) | 61.62 ± 1.27(2) | 88.21 ± 0.57(10) | **88.35 ± 0.49(10)** |
| | SVC | 86.96 ± 0.76(10) | **87.56 ± 0.84(10)** | 71.76 ± 0.74(2) | 78.44 ± 0.69(4) | 85.91 ± 0.86(7) | 82.91 ± 0.69(4) | 86.35 ± 0.83(6) | 72.01 ± 0.60(2) | 86.96 ± 0.76(10) | 86.96 ± 0.76(10) |
| | XGB | **88.40 ± 0.64(10)** | 87.08 ± 0.61(10) | 71.87 ± 0.86(2) | 78.60 ± 0.80(4) | 87.58 ± 0.71(7) | 84.43 ± 0.60(4) | 86.84 ± 0.55(6) | 72.70 ± 0.66(2) | **88.40 ± 0.64(10)** | **88.40 ± 0.64(10)** |

* The numbers in brackets indicate the number of selected features.

**Table 7**
Wilcoxon rank sum test of PMLCE-S and PMLCE-D.

(a) Wilcoxon Rank Sum Test of PMLCE-S

| | Raw | ICA | Isomap | PCA | Relief | LFSACIE | LARD | LFSASI |
|---|---|---|---|---|---|---|---|---|
| KNN | 0.0001 | 0.0142 | 0.0000 | 0.0001 | 0.0000 | 0.0024 | 0.0050 | 0.0003 |
| RF | 0.0635 | 0.0010 | 0.0000 | 0.0000 | 0.0001 | 0.0012 | 0.0004 | 0.0003 |
| SVC | 0.0190 | 0.0717 | 0.0001 | 0.0006 | 0.0007 | 0.0041 | 0.0008 | 0.0019 |
| XGB | 0.0009 | 0.0062 | 0.0000 | 0.0000 | 0.0001 | 0.0002 | 0.0001 | 0.0005 |

(b) Wilcoxon Rank Sum Test of PMLCE-D

| | Raw | ICA | Isomap | PCA | Relief | LFSACIE | LARD | LFSASI |
|---|---|---|---|---|---|---|---|---|
| KNN | 0.0021 | 0.0253 | 0.0000 | 0.0016 | 0.0000 | 0.0083 | 0.0295 | 0.0004 |
| RF | 0.0093 | 0.0004 | 0.0000 | 0.0001 | 0.0001 | 0.0003 | 0.0003 | 0.0001 |
| SVC | 0.0244 | 0.0551 | 0.0001 | 0.0005 | 0.0007 | 0.0031 | 0.0004 | 0.0019 |
| XGB | 0.0031 | 0.0081 | 0.0000 | 0.0000 | 0.0001 | 0.0002 | 0.0002 | 0.0007 |

## 6. Summary and future research direction

This study introduces advanced parallel feature selection algorithms, founded on matrix formulations and reinforced by the theoretical principles of local neighborhood rough sets and composite entropy. It begins with an in-depth exposition of the foundational concepts of neighborhood rough sets and the complex role of composite entropy. Based on this theoretical groundwork, two parallel feature selection algorithms are presented: one designed for static datasets and another for dynamic datasets, both exhibiting high effectiveness. Empirical testing validates the algorithms' precision, efficiency, and superior operational performance.

Table 8
Cumulative running time (s) of algorithms based on rough set.

| | | $t_0$ | $t_1$ | $t_2$ | $t_3$ | | $t_0$ | $t_1$ | $t_2$ | $t_3$ |
|---|---|---|---|---|---|---|---|---|---|---|
| Chess | LFSACIE | 1.70 | 26.17 | 42.30 | 58.42 | Grid | 12.18 | 51.20 | 140.83 | 236.50 |
| | LARD | 12.66 | 74.02 | 203.62 | 340.37 | | 23.78 | 127.76 | 358.77 | 445.93 |
| | LFSASI | 3.84 | 84.84 | 241.29 | 324.30 | | 14.07 | 70.77 | 197.21 | 697.39 |
| | PMLCE-S 1 process | 10.26 | 25.18 | 48.18 | 63.86 | | 5.83 | 28.86 | 84.17 | 176.59 |
| | PMLCE-S 8 processes | 5.87 | 13.95 | 24.94 | 32.34 | | 2.66 | 13.11 | 37.25 | 79.58 |
| | PMLCE-D 1 process | 6.29 | 14.44 | 27.44 | 34.79 | | 2.20 | 9.98 | 33.42 | 47.72 |
| | PMLCE-D 8 processes | 4.69 | 10.09 | 16.88 | 21.67 | | 2.76 | 8.96 | 22.57 | 35.25 |
| Spambase | LFSACIE | 18.18 | 88.48 | 256.43 | 424.21 | Bean | 26.98 | 153.70 | 425.34 | 695.42 |
| | LARD | 79.00 | 380.88 | 1103.84 | 1757.10 | | 69.28 | 359.78 | 989.40 | 1272.21 |
| | LFSASI | 12.42 | 61.84 | 173.41 | 284.76 | | 50.43 | 252.06 | 704.71 | 1156.32 |
| | PMLCE-S 1 process | 51.90 | 107.74 | 162.29 | 209.30 | | 13.87 | 70.06 | 199.44 | 319.11 |
| | PMLCE-S 8 processes | 30.00 | 58.21 | 84.12 | 106.25 | | 5.92 | 29.72 | 84.86 | 136.05 |
| | PMLCE-D 1 process | 32.16 | 66.73 | 107.94 | 129.76 | | 5.13 | 25.24 | 79.40 | 111.03 |
| | PMLCE-D 8 processes | 26.41 | 46.52 | 63.06 | 75.09 | | 6.30 | 20.52 | 51.91 | 85.25 |
| Quality | LFSACIE | 2.32 | 12.01 | 31.59 | 48.13 | Telescope | 34.40 | 183.33 | 504.61 | 806.27 |
| | LARD | 3.46 | 18.42 | 51.63 | 73.57 | | 52.89 | 264.38 | 720.87 | 1061.50 |
| | LFSASI | 4.71 | 23.28 | 64.80 | 106.21 | | 40.59 | 200.39 | 558.95 | 916.67 |
| | PMLCE-S 1 process | 0.98 | 5.12 | 14.53 | 23.26 | | 16.17 | 83.15 | 231.50 | 366.55 |
| | PMLCE-S 8 processes | 0.62 | 2.68 | 7.32 | 11.66 | | 7.75 | 40.56 | 117.23 | 186.07 |
| | PMLCE-D 1 process | 0.43 | 1.90 | 5.56 | 7.46 | | 6.52 | 32.52 | 102.90 | 142.13 |
| | PMLCE-D 8 processes | 0.63 | 1.91 | 4.48 | 6.62 | | 6.25 | 17.33 | 46.13 | 72.56 |
| Turkiye | LFSACIE | 7.57 | 37.26 | 105.81 | 173.48 | | | | | |
| | LARD | 37.80 | 198.76 | 589.42 | 941.07 | | | | | |
| | LFSASI | 12.57 | 63.08 | 175.38 | 287.94 | | | | | |
| | PMLCE-S 1 process | 8.70 | 36.11 | 94.62 | 137.10 | | | | | |
| | PMLCE-S 8 processes | 5.24 | 18.47 | 46.79 | 67.12 | | | | | |
| | PMLCE-D 1 process | 5.30 | 19.00 | 52.91 | 65.62 | | | | | |
| | PMLCE-D 8 processes | 5.77 | 19.03 | 45.84 | 60.99 | | | | | |

Compared to traditional feature selection algorithms based on rough sets, the algorithms proposed in this paper offer several distinct advantages:

(1) The integration of local neighborhood rough sets and composite entropy significantly enhances the performance of our algorithm. This improvement allows for more effective feature screening, ultimately increasing the predictive power of the overall model.
(2) By employing matrix operations, our algorithm achieves a marked improvement in computational efficiency over conventional rough set-based methods. The addition of parallel computing further enhances this efficiency. Furthermore, the algorithm's adaptability to dynamic data makes it more practical for handling real-world scenarios where the volume of data and features changes continuously.
(3) Traditional rough set-based algorithms struggle with large datasets due to their low computational efficiency. In contrast, our algorithm significantly improves efficiency, enabling the processing of large-scale datasets that traditional algorithms cannot handle. This advancement provides a competitive edge in addressing complex data challenges.

However, our experimental results also highlight certain limitations of the proposed algorithm that require further refinement:

(1) Composite entropy, while useful, still exhibits some uncertainty when measuring information systems. It provides only a single-granularity perspective, lacking the precision of a multi-granularity approach, which results in an incomplete representation of information system uncertainty.
(2) Although our algorithm processes larger datasets than previous rough set-based feature selection algorithms, memory limitations still hinder its ability to handle extremely large datasets efficiently.
(3) The current rough set-based feature selection algorithm is limited to tabular data and is not capable of processing other types of data, such as audio or images, restricting its broader applicability.

To address these limitations, our ongoing efforts and future research directions include:

(1) Investigating a feature selection algorithm that combines composite entropy and rough sets with a multi-granularity approach to improve the measurement of information system uncertainty, thereby enhancing the algorithm's effectiveness.
(2) Exploring strategies to reduce computing resource consumption and enhance the ability to process larger datasets more efficiently. This may involve techniques such as compressing neighborhood matrices to minimize memory usage or utilizing GPUs to accelerate parallel matrix computations.
(3) Investigating the application of rough set-based feature selection algorithms to non-tabular data, such as images and audio, to expand the scope of rough set-based methods.

## CRediT authorship contribution statement

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

## 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.

## Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62376229 and in part by the Natural Science Foundation of Chongqing, China under Grant CSTB2023NSCQLZ
X0027.

## Data availability

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

## References

[1] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (1982) 341–356.
[2] Z. Pawlak, S.K.M. Wong, W. Ziarko, et al., Rough sets: probabilistic versus deterministic approach, Int. J. Man-Mach. Stud. 29 (1) (1988) 81–95.
[3] Y. Yao, S. Wong, P. Lingras, A decision-theoretic rough set model, Methodol. Intell. Syst. 5 (1990) 17–24.
[4] Y. Qian, X. Liang, Q. Wang, J. Liang, B. Liu, A. Skowron, Y. Yao, J. Ma, C. Dang, Local rough set: a solution to rough data analysis in big data, Internat. J. Approx. Reason. 97 (2018) 38–63.
[5] Q. Wang, Y. Qian, X. Liang, Q. Guo, J. Liang, Local neighborhood rough set, Knowl.-Based Syst. 153 (2018) 53–64.
[6] C. Liu, J. Lai, B. Lin, D. Miao, An improved ID3 algorithm based on variable precision neighborhood rough sets, Appl. Intell. 53 (20) (2023) 23641–23654.
[7] A.O. Atagün, H. Kamacı, Strait soft sets and strait rough sets with applications in decision making, Soft Comput. 27 (20) (2023) 14585–14599.
[8] Z. Yuan, H. Chen, T. Li, B. Sang, S. Wang, Outlier detection based on fuzzy rough granules in mixed attribute data, IEEE Trans. Cybern. 52 (8) (2021) 8399–8412.
[9] D. Guo, W. Xu, W. Ding, Y. Yao, X. Wang, W. Pedrycz, Y. Qian, Concept-cognitive learning survey: Mining and fusing knowledge from data, Inf. Fusion 109 (2024) 102426.
[10] D. Guo, W. Xu, Y. Qian, W. Ding, Fuzzy-granular concept-cognitive learning via three-way decision: Performance evaluation on dynamic knowledge discovery, IEEE Trans. Fuzzy Syst. 32 (3) (2024) 1409–1423.
[11] J. Jiang, X. Zhang, Z. Yuan, Multi-label feature selection using self-information in divergence-based fuzzy neighborhood rough sets, Pattern Recognit. (2024) 110684.
[12] T. Yin, H. Chen, Z. Wang, K. Liu, Z. Yuan, S.-J. Horng, T. Li, Feature selection for multilabel classification with missing labels via multi-scale fusion fuzzy uncertainty measures, Pattern Recognit. 154 (2024) 110580.
[13] W. Wei, J. Liang, J. Wang, Y. Qian, Decision-relative discernibility matrices in the sense of entropies, Int. J. Gen. Syst. 42 (7) (2013) 721–738.
[14] L. Sun, L. Wang, W. Ding, Y. Qian, J. Xu, Feature selection using fuzzy neighborhood entropy-based uncertainty measures for fuzzy neighborhood multigranulation rough sets, IEEE Trans. Fuzzy Syst. 29 (1) (2020) 19–33.
[15] C. Wang, Y. Huang, W. Ding, Z. Cao, Attribute reduction with fuzzy rough self-information measures, Inform. Sci. 549 (2021) 68–86.
[16] B. Sang, H. Chen, L. Yang, T. Li, W. Xu, Incremental feature selection using a conditional entropy based on fuzzy dominance neighborhood rough sets, IEEE Trans. Fuzzy Syst. 30 (6) (2021) 1683–1697.
[17] W. Xu, K. Yuan, W. Li, W. Ding, An emerging fuzzy feature selection method using composite entropy-based uncertainty measure and data distribution, IEEE Trans. Emerg. Top. Comput. Intell. 7 (1) (2022) 76–88.
[18] K. Yuan, D. Miao, Y. Yao, H. Zhang, X. Zhao, Feature selection using zentropy-based uncertainty measure, IEEE Trans. Fuzzy Syst. 32 (4) (2024) 2246–2260.
[19] K. Yuan, D. Miao, W. Pedrycz, W. Ding, H. Zhang, Ze-HFS: Zentropy-based uncertainty measure for heterogeneous feature selection and knowledge discovery, IEEE Trans. Knowl. Data Eng. PP (2024) 1–14.
[20] W. Huang, Y. She, X. He, W. Ding, Fuzzy rough sets-based incremental feature selection for hierarchical classification, IEEE Trans. Fuzzy Syst. 31 (10) (2023) 3721–3733.
[21] J. Zhao, Y. Ling, F. Huang, J. Wang, E.W. See-To, Incremental feature selection for dynamic incomplete data using sub-tolerance relations, Pattern Recognit. 148 (2024) 110125.
[22] J. Zhao, D. Wu, J. Wu, W. Ye, F. Huang, J. Wang, E.W. See-To, Consistency approximation: Incremental feature selection based on fuzzy rough set theory, Pattern Recognit. (2024) 110652.
[23] Y. Pan, W. Xu, Q. Ran, An incremental approach to feature selection using the weighted dominance-based neighborhood rough sets, Int. J. Mach. Learn. Cybern. 14 (4) (2023) 1217–1233.
[24] X. Zhang, J. Li, Incremental feature selection approach to interval-valued fuzzy decision information systems based on $\lambda$-fuzzy similarity self-information, Inform. Sci. 625 (2023) 593–619.
[25] W. Xu, Q. Bu, Matrix-based incremental feature selection method using weight-partitioned multigranulation rough set, Inform. Sci. (2024) 121219.
[26] H. Chen, T. Li, Y. Cai, C. Luo, H. Fujita, Parallel attribute reduction in dominance-based neighborhood rough set, Inform. Sci. 373 (2016) 351–368.
[27] F. Nosheen, U. Qamar, M.S. Raza, A parallel rule-based approach to compute rough approximations of dominance based rough set theory, Eng. Appl. Artif. Intell. 115 (2022) 105285.
[28] V.H. Turaga, S. Chebrolu, Parallel computation of probabilistic rough set approximations, in: Proceedings of International Conference on Computational Intelligence: ICCI 2021, Springer, 2022, pp. 431–445.
[29] A. Hyvärinen, E. Oja, Independent component analysis: algorithms and applications, Neural Netw. 13 (4–5) (2000) 411–430.
[30] J.B. Tenenbaum, V.d. Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (5500) (2000) 2319–2323.
[31] M.E. Tipping, C.M. Bishop, Probabilistic principal component analysis, J. R. Stat. Soc. Ser. B Stat. Methodol. 61 (3) (1999) 611–622.
[32] K. Kira, L.A. Rendell, The feature selection problem: Traditional methods and a new algorithm, in: Proceedings of the Tenth National Conference on Artificial Intelligence, 1992, pp. 129–134.
[33] C. Wang, Y. Huang, M. Shao, Q. Hu, D. Chen, Feature selection based on neighborhood self-information, IEEE Trans. Cybern. 50 (9) (2019) 4031–4042.

**Weihua Xu** received the Ph.D. degree in mathematics from School of Sciences, Xi'an Jiaotong University, Xi'an, China, in 2007, and the M.Sc. degree in mathematics from School of Mathematics and Information Sciences, Guangxi University, Nanning, China, in 2004. He is currently a Professor with the College of Artificial Intelligence, Southwest University, Chongqing, China. He has published 5 monographs and more than 200 articles in international journals. His current research interests include granular computing, cognitive computing, and information fusion. Dr. Xu also serves as a Senior Member of Chinese Association for Artificial Intelligence (CAAI). He serves on the Associate editor of International Journal of Machine Learning and Cybernetics and Journal of Intelligent and Fuzzy Systems.

**Weirui Ye** received the B.Sc. degree from the School of Information Management, Shanghai Lixin University of Accounting and Finance, Shanghai, China, in 2022. He is currently working toward the M.Sc. degree in computer science and technology with the School of Artificial Intelligence, Southwest University, Chongqing, China. His research interests cover feature selection, granular computing, and knowledge discovery.