machine learning & knowledge extraction

MDPI

Article

An Improved Differential Evolution Algorithm Using Mahalanobis Distance and Cholesky Decomposition for Wrapper-Based Feature Selection

Angel Casas-Ordaz [1], Diego Oliva [1,*], Marco Pérez-Cisneros [1,*], Guillermo Sosa-Gómez [2,3,*], Itzel Aranguren [1,4] and Arturo Valdivia-G [1,4]

[1] Departamento de Ingeniería Electro-Fotónica, Centro Universitario de Ciencias Exactas e Ingenierías, Universidad de Guadalajara, Guadalajara 44430, Jalisco, Mexico; angel.casas5699@alumnos.udg.mx (A.C.-O.); itzel.aranguren@academicos.udg.mx (I.A.); arturo.valdivia@academicos.udg.mx (A.V.-G.)
[2] Facultad de Ciencias Económicas y Empresariales, Universidad Panamericana, Álvaro del Portillo 49, Zapopan 45010, Jalisco, Mexico
[3] Laboratory for Reputation Management in Education, HSE University Saint-Petersburg, Griboedova Embankment 123, Office 502, 190068 Saint-Petersburg, Russia
[4] Departamento de Ciencias Computacionales, Centro Universitario de Ciencias Exactas e Ingenierías, Universidad de Guadalajara, Guadalajara 44430, Jalisco, Mexico
* Correspondence: diego.oliva@cucei.udg.mx (D.O.); marco.perez@cucei.udg.mx (M.P.-C.); gsosag@up.edu.mx (G.S.-G.)

Abstract

In machine learning, feature selection is a fundamental component that plays a crucial role in improving prediction accuracy and reducing the computational time of classification models. Feature selection involves eliminating irrelevant or redundant features that can negatively affect model learning. This article proposes a wrapper-based method that uses a binary evolutionary algorithm for feature selection. The Differential Evolution algorithm performs feature selection and is combined with Mahalanobis distance and Cholesky decomposition (MaCRO-DE). Using Cholesky decomposition to calculate Mahalanobis distance helps to determine the proximity of feature subsets, which is useful for deciding which features to select or discard. Subsequently, two classifiers (K-nearest neighbors and the support vector machine) are used as evaluators, along with a V-shaped transfer function, to identify the most suitable option for the proposed method. To evaluate the effectiveness of the proposed methodology, the CEC-2017 benchmark and twelve widely used datasets from the UCI repository are employed. Experimental tests are conducted using thirteen prominent metaheuristic algorithms. The results indicate that the proposal effectively achieves its objectives, outperforming the benchmark across the performance indicators used for comparison, including Accuracy, F1-Score, Recall, Precision, Fitness value, and number of selected features.

Keywords: Cholesky decomposition; Differential Evolution; feature selection; Mahalanobis distance; wrapper-based feature selection

Check for updates

Received: 7 August 2026
Revised: 11 September 2026
Accepted: 18 September 2026
Published: 20 September 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Feature selection constitutes a crucial pre-processing step in enhancing datasets for machine learning and data mining applications, since datasets may contain many irrelevant or redundant features, which can compromise the effectiveness of the underlying classification algorithm [1]. The objective of feature selection is to decrease the number of features

by creating subsets that eliminate redundant or irrelevant features while maintaining or improving the classification model's accuracy [2]. The process of identifying subsets is considered to be NP-hard, meaning that for a given set of N features, a complete search can involve $2^n - 1$ possible subsets [3]. Consequently, demand has increased for feature selection methodologies. The most prominent and frequently mentioned are embedded, filter, and wrapper [4]. The filter method does not utilize a learning algorithm for feature classification; instead, it utilizes statistical measures or geometric properties to filter features. The wrapper method integrates learning and search algorithms to evaluate feature subsets and identify those that improve performance and accuracy. Furthermore, embedded approaches incorporate a classification algorithm and select features during classifier training. Among these approaches, wrapper methods are particularly relevant for the combinatorial nature of feature selection, as they explicitly search for feature subsets based on their effect on a learning algorithm's performance. However, the large search space in feature selection makes exhaustive evaluation computationally impractical, motivating the use of optimization techniques.

Since selecting an optimal subset of features can be formulated as a search problem over a large and often complex solution space, metaheuristic algorithms have increasingly been employed to guide this process efficiently [5]. Metaheuristic approaches have been systematically categorized by the search techniques employed, the evaluation criteria considered, and the number of objectives addressed [6], with supervised, unsupervised, and semi-supervised learning techniques adopted to evaluate and analyze the resulting feature subsets. Recent, innovative proposals include [7], which proposes a Differential Evolution (DE) algorithm that replaces traditional random initialization with chaotic sequences based on Lyapunov time and uses an interesting parallelization strategy to find feature subsets for high-dimensional medical classification tasks. Another DE approach, evaluated on twenty-two datasets with feature counts ranging from small to large, is presented in [8]. The proposal incorporates individual grouping strategies and a repository of previous solutions modeled by Gaussian and Cauchy distributions, which enhances exploration of different regions of the search space. In [9], the authors propose a methodology that combines filter and wrapper methods. In the initial stage, the Fisher score and information gain classify features, and adaptive selection is performed based on feature importance. In the next stage, the wrapper method selects features. This methodology was implemented in Particle Swarm Optimization (PSO), Grey Wolf Optimizer (GWO), and the Fish Migration Optimizer (FMO) algorithms, providing promising results on high-dimensional datasets. The work proposed by [10] presents a multi-subgroup adaptive optimization method based on the PSO algorithm, in which each subgroup explores solutions in distinct subspaces of the problem and dynamically adjusts the subgroup by reducing the particle length based on its performance during evolution. The authors evaluated the proposal's efficacy through experiments using a K-nearest-neighbors (KNN) classifier on ten high-dimensional data sets. These experiments were compared with two versions of the classic PSO algorithm, yielding noteworthy results.

A notable implementation of metaheuristic algorithms for feature selection is presented in [11], where the Gravitational Search Algorithm (GSA) is modified to improve performance by balancing exploration and exploitation, incorporating two schemes: a crossover operator is used initially, followed by a mutation operator. The binary version of the algorithm utilizes a V-Shape transfer function. The feature selection wrapper model is implemented using two distinct classifiers, KNN and Decision Tree (DT). The efficacy of the proposed model was evaluated through a comparative analysis of eighteen datasets from the UCI repository, alongside other nature-inspired algorithms; results revealed noteworthy findings in addressing the feature selection problem. The feature selection problem has also

been applied to recent metaheuristics such as the Atom Search Optimization (ASO) algorithm [12]. The ASO algorithm provides a binary selection algorithm for selecting a subset of features through eight S-shaped and V-shaped transfer functions. The authors tested the proposal on twenty-two datasets from the UCI repository and compared it with five state-of-the-art algorithms. The ASO algorithm demonstrated strong competitiveness and superior results. One algorithm widely studied for the feature selection problem is GWO. In [13], a binary version of the GWO is proposed. In the initial phase, the three optimal solutions are binarized, followed by the implementation of crossover. In the subsequent section, a sigmoidal function is employed, and random thresholds are utilized to delineate the new binary position. Comparative assessments are conducted using two conventional algorithms, PSO and Genetic Algorithms (GA), on eighteen distinct data sets from the UCI repository. The findings reveal the efficacy of the binary version of GWO in facilitating feature selection.

Another binary GWO approach is presented in [14]; this article introduces two novel aspects. First, the parameter-calculation equation is modified, and then five transfer functions are proposed to convert continuous values to binary values. The authors evaluate the implementation for feature selection using datasets from the UCI repository. Lastly, [15] presents a hybridization of PSO and GWO. The binary hybridization called BGWOPSO employs a wrapper-based method with a KNN classifier with a Euclidean distance metric to find the optimal features on eighteen datasets from the UCI repository. The results show that BGWOPSO significantly outperformed binary versions of GWO and PSO, among other metaheuristics, in terms of accuracy and computational time. The studies discussed above demonstrate the potential of metaheuristic algorithms for addressing feature selection, with research efforts focusing on aspects such as population initialization, population clustering, and selection strategies based on diverse indicators. Nevertheless, statistical analysis of population behavior as a source of information to guide the search process remains relatively unexplored. Characterizing how individuals are distributed and how their relationships evolve throughout the optimization process may provide valuable insights for identifying informative and complementary features. In this context, analyzing distances between individuals based on each dataset's characteristics offers a potential way to capture population structure, particularly in high-dimensional feature spaces. Therefore, continued investigation of population behavior and the development of techniques that accurately extract and exploit this information represent promising directions for improving metaheuristic-based feature selection and, ultimately, obtaining more effective and informative feature subsets.

This work investigates whether incorporating statistical population behavior can enhance metaheuristic search capabilities in feature selection tasks. DE algorithm serves as the foundation for this study, as its population-based search mechanism and mutation operator naturally accommodate relational data among individuals. To exploit this advantage, we introduce MaCRO-DE, a novel variant that incorporates a Mahalanobis distance-based modification into the DE mutation operator. Unlike conventional distance metrics, Mahalanobis distance accounts for variance and inter-dimensional correlation, providing a richer characterization of individual relationships within high-dimensional search spaces. To preserve computational efficiency, Cholesky decomposition is employed to manage the associated covariance structure. The main contributions of this work are summarized as follows:

- DE algorithm has been modified with the strategic objective of enhancing its exploration-exploitation balance by implementing statistical techniques that account for the population's probability distribution.

- The Mahalanobis distance and Cholesky decomposition are strategically implemented to select subsets of features, and these methods have demonstrated competitive results compared to other statistical distances.
- The MaCRO-DE algorithm has been shown to produce competitive results in both single-objective benchmark test functions and feature selection datasets.

The subsequent sections of this article will adhere to the following structure. Section 2 provides background information. Section 3 depicts the proposal. The binary version is presented in Section 4. The results are in Section 5, and the conclusions are in Section 6.

2. Background

This section aims to provide a general description of the preliminary stages to facilitate comprehensive understanding of the proposal's implementation in subsequent stages. In this investigation, we thoroughly examine the DE algorithm's description and the implementation of distance-based measures in evolutionary algorithms.

2.1. Differential Evolution

Differential Evolution (DE) [16] is a widely studied algorithm in evolutionary computation. It is well known that DE has a general structure comprising initialization, mutation, crossover, and selection. The following brief description of the algorithm aims to clarify its structure:

2.1.1. Initialization

In this phase, a series of potential solutions is randomly generated within specified limits. Each solution is represented as a vector x_i^g, which is defined as follows $x_i^g = \{x_{i,1}^g, x_{i,2}^g \ldots, x_{i,d}^g\}$, $i = 1, \ldots, Np$. In this model, the population size is denoted by Np, the initial condition of the generation is set to $g = 0$, and the problem's dimensionality is denoted by d. Equation (1) defines the generation of individuals.

$$x_{i,d}^0 = x_{min,d} + rand(0,1) \cdot (x_{max,d} - x_{min,d}) \tag{1}$$

In this context, the notation $rand(0,1)$ represents a random variable that is uniformly distributed within the interval $[0,1]$. The lower and upper bounds for each search space dimension are represented by x_{min} and x_{max}, respectively.

2.1.2. Mutation

The DE algorithm employs a random perturbation mechanism in which the elements of each candidate solution are recombined to generate a modified version of each individual, thereby producing the so-called mutant vector v_i^g. The most common mutation strategy employed is $DE/rand/1$ (see Equation (2)), which is specified in the following:

$$v_i^{g+1} = x_{r3}^g + F(x_{r1}^g - x_{r2}^g) \tag{2}$$

where the population vectors x_{r1}, x_{r2}, and x_{r3} are randomly selected, with the understanding that $r1$, $r2$, and $r3$ are not equal to the index of the target vector i. The scaling factor, F, is a fixed value within the range of $[0,2]$.

2.1.3. Crossover

The crossover process enhances the population's diversity, and the constant p_{cr} performs a controlling role within the interval $[0,1]$. A trial vector, designated as u_i^g, is

generated in accordance with the formulation presented in Equation (3). This is achieved by combining the target vector, represented as $x^g_{i,d}$, with the mutant vector, denoted as $v^g_{i,d}$.

$$u^{g+1}_i = u^g_d \begin{cases} v^g_{i,d}, & \text{if } d = d_{rand} \text{ or } rand(0,1) \leq p_{cr} \\ x^g_{i,d}, & \text{otherwise} \end{cases} \tag{3}$$

As illustrated, d_{rand} is a randomly chosen index $\in 1, 2, \ldots, d$ which ensures that u^g_i includes at least one parameter of the mutant vector v^g_i. Meanwhile, $rand(0,1)$ denotes a uniform random variable in the range of $[0,1]$.

2.1.4. Selection

This selection process enables the DE to assess the viability of the target or test solution in subsequent generations by evaluating its objective function values, as shown in Equation (4). The vector that exhibits the most favorable values is retained within the population of the subsequent generation. This operation is performed as follows:

$$x^{g+1}_i = \begin{cases} u^g_{i,d}, & \text{if } f(u^{g+1}_i) \leq f(x^g_i), \\ x^g_{i,d}, & \text{otherwise} \end{cases} \tag{4}$$

subsequent to the new population being constituted, the mutation, crossover, and selection processes are replicated iteratively until the optimal solution is determined or the predefined termination criterion is achieved.

2.2. Distances Between Probability Distributions

The purpose of this section is to provide the reader with a comprehensive overview of the implementations of distances utilized in the most prevalent probability distributions. The objective of this analysis is to underscore the rationale for selecting the Mahalanobis distance as the preferred implementation option in the proposal. Distance measurement is fundamental to machine learning and statistical processes, particularly in cases such as the multivariate Gaussian distribution [17]. It is present in almost any field that uses statistical methods for data analysis, such as clustering, density estimation, and image recognition. In order to establish the significance of the distance utilized in this section, it is necessary to first consider the most prominent probability density function in recent years. The normal distribution is widely regarded as a foundational paradigm within the scientific community, particularly given its prevalence in the physical sciences, where it underlies numerous empirically observed variables. This distribution plays an crucial role in various fields, including parametric distribution in clustering, density estimation, and classification [18]. Accordingly, the starting point is the univariate normal distribution. It is imperative to consider a random variable x that follows a normal distribution, defined by a mean μ and a variance σ^2, as shown in Equation (5). The probability density function of x is expressed as follows:

$$\mathcal{N}(x|u, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \tag{5}$$

in this context, the term $(x-\mu)^2$ quantifies the distance of a value x from the mean μ of the distribution. Consequently, the probability density decreases exponentially with increasing distance from the mean. The maximum value of the density is achieved at the mean value $x = \mu$, given as $\mathcal{N}(u) = -\frac{1}{2\pi\sigma^2}$, which is inversely proportional to the standard deviation (σ) of the distribution. By extending the present case to a distribution of greater dimensionality, specifically d-dimensional, and employing the random variable vector x, it can be stated that x possesses a multivariate normal distribution with parameters μ,

the mean, and Σ, the covariance matrix, as shown in Equation (6). The following joint multivariate probability density function substantiates this assertion:

$$\mathcal{N}(x|u,\Sigma) = \frac{1}{\sqrt{(2\pi)^2|\Sigma|}} e^{\left(-\frac{1}{2\sigma^2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)} \tag{6}$$

where μ is a d-dimensional mean vector, Σ is a $d \times d$ covariance matrix, $|\Sigma|$ denotes the determinant of Σ, and Σ^{-1} is the inverse of the covariance matrix. The Mahalanobis distance is the quadratic term in the exponent of the density of the distribution. The Mahalanobis distance was first defined in 1936 by the Indian statistician Prasanta Chandra Mahalanobis [19]. In his pioneering work, Mahalanobis established the Mahalanobis distance as a statistical measure that accounts for both the correlations among variables and the multivariate dispersion of the data. The fundamental idea is to provide a generalized method for measuring distance in a multidimensional space that scales with the data distribution. The underlying formula for calculating the Mahalanobis distance is shown in Equation (7).

$$\triangle^2 = (x-\mu)^\top \Sigma^{-1}(x-\mu) \tag{7}$$

Formally, the generalized Mahalanobis distance between two points x and x' can be expressed as:

$$d_M(x,x') = \sqrt{(x_i - x_i')^\top \Sigma^{-1}(x_i - x_i')} \tag{8}$$

which corresponds to the same structure as the exponent term in the multivariate Gaussian density when x' is replaced by the mean vector μ. This relationship highlights that the Euclidean distance is a particular case of the Mahalanobis distance, valid when $\Sigma = I$ and the data dimensions are statistically independent. Consequently, the Euclidean distance method does not consider the information about the correlation between attributes, while the Mahalanobis distance method explicitly incorporates this information. In the above Equation (7), the quadratic term $(x-\mu)^\top \Sigma^{-1}(x-\mu)$ represents the measure of the distance of a multivariate observation x from the mean μ, weighted by the inverse of the covariance matrix Σ. This mathematical formula normalizes each component based on its variance and considers correlations among features. When the covariance matrix Σ is assumed to be the identity matrix I, it can be deduced that all features are uncorrelated and possess an equivalent variance. Consequently, the Mahalanobis distance is reduced to the Euclidean distance. Although the Mahalanobis distance constitutes the foundation of the proposed approach, other distance measures are also considered to evaluate the influence of this choice on the algorithm's behavior. Euclidean, Manhattan, Chebyshev, and Minkowski distances are considered for the experimental conditions. Detailed descriptions of these metrics can be found in Supplementary Note S1.

3. Improved Differential Evolution with Mahalanobis Distance and Cholesky Decomposition

This article proposes an enhancement to the Improved Differential Evolution with Mahalanobis Distance algorithm. The initial version of this algorithm used the Mahalanobis distance to constrain the number of individuals required to execute the DE algorithm's mutation. To achieve this enhancement, two significant updates were implemented in the algorithm. The first approach used Cholesky decomposition, while the second selected a range of individuals for mutation based on an exploration-exploitation ratio. Subsequently, once the distance calculation has been adjusted, this value is used to configure the algorithm's internal parameters (scaling factor and crossover rate). The selection of the DE algorithm is motivated by the structure of its mutation operator, which directly utilizes individuals from the population to generate new candidate solutions. This attribute

provides a natural mechanism for incorporating the Mahalanobis distance and Cholesky decomposition by ensuring that individuals are selected not only randomly but also in a controlled manner, with a statistical relationship, thereby contributing to the balance between exploration and exploitation. Therefore, this section details the implementation of the improvements described above. Initially, the direct implementation of the enhanced Cholesky decomposition is employed, with its usage linked to the Mahalanobis distance. Subsequently, the entire mechanism is defined as a self-adaptive system of control parameters. This development led to a novel algorithm, designated Improved Differential Evolution with Mahalanobis Distance and Cholesky Decomposition (MaCRO-DE). The primary distinctions between the aforementioned proposal and the new proposal under consideration are as follows:

- The calculation of the Mahalanobis distance is enhanced through the implementation of the Cholesky decomposition, leading to a more precise determination of the relationship between the optimal solution and each individual within the DE algorithm population.
- The probability of choosing individuals from the population that will interact with the mutation operator is switchable between nearby and distant individuals.
- The calculation of the Mahalanobis distance with Cholesky decomposition is computationally less expensive. The probability of selecting distant and nearby particles contributes to the algorithm's diversity.

In light of these considerations, the implementation of the Mahalanobis distance process is initiated through a systematic approach, starting with Equations (7) and (8). To accomplish this objective, it is imperative to calculate the mean, variance, and covariance. Subsequently, the modification begins with a Cholesky decomposition to calculate the covariance. According to the established definition of Cholesky decomposition [20], the application is contingent upon the condition that A belongs to the set $\mathbb{C}^{N\times N}$ of positive definite Hermitian matrices. The Cholesky decomposition process entails the division of A into a lower triangular matrix and its conjugate transpose $A = LL^*$. In light of these considerations, the implementation is realized through the use of an upper triangular matrix, denoted by $R = L^*$, as $A = R^*R$. The elements of the matrix R are specified (Equations (9) and (10)) in the following manner. First, for the diagonal elements, the calculation is as follows:

$$r_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} r_{ki}^2} \tag{9}$$

and for the upper-triangular elements $(i < j)$:

$$r_{ij} = \frac{1}{r_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj} \right) \tag{10}$$

since the older values of a are not required for computing newer elements, they can be replaced by r. As a result, the algorithm can be performed in place using the same memory for matrices A and R. The Cholesky decomposition is characterized by a computational complexity class that is polynomial in the input size, specifically of order $\mathcal{O}(n^2)$. This decomposition requires approximately $\frac{1}{6}n^3$ operations to execute. The numerical stability of matrix inversion based on this decomposition is guaranteed under the condition that the matrices are well-conditioned. Additionally, a numerical safety mechanism has been introduced into the proposal to ensure the applicability of the Cholesky decomposition when the covariance matrix has insufficient rank or is ill-conditioned. First, the covariance matrix is symmetry-adjusted and regularized as $\Sigma_\varepsilon = (\Sigma + \Sigma^T)/2 + \varepsilon I$, where $\varepsilon = 10^{-6}$. If the

Cholesky decompositions cannot be obtained due to numerical ill-conditioning, the Moore-Penrose pseudoinverse is used instead. This safeguard, specific to the implementation, prevents singular or nearly singular covariance matrices from disrupting the optimization process. To continue with the implementation of the Mahalanobis distance calculation, it is necessary to replace x with x_{best}. In the original algorithm [21], this ensures that the distance calculation is based on the best solution at each iteration, creating a unified calculation that can be used later. The implementation of the MaCRO-DE using Cholesky decomposition is indicated by reduced variation in the calculations, with estimates of the closest and furthest individuals exhibiting greater stability (see Equation (11)). This calculation determines the previous selection range, and it is done using a cumulative distribution function of the chi-square distribution:

$$\tau = F_{\chi^2}^{-1}(q; v) \tag{11}$$

where q denotes the cumulative probability level and v represents the number of degrees of freedom, equivalent to the dimensionality of the problem. Given that the grouping criterion is based on the Mahalanobis distance, it is expressed on the same scale as the chi-squared quantile. Individuals that satisfy $\triangle^2 \leq \tau$ are assigned to the near group, while those that satisfy $\triangle^2 > \tau$ are assigned to the far group. The probability level q should be interpreted as a cumulative probability threshold of the multidimensional chi-squared distribution, not as a direct equivalent of the one-standard-deviation interval of a univariate normal distribution. For initial configurations, a reference value of $q = 0.68$ was considered. However, the final value used was $q = 0.50$, selected based on the sensitivity analysis described in Supplementary Note S8. Calculating the Mahalanobis distance and verifying that the random selection of the three solutions required for mutation falls within the confidence interval around the best solution are essential prerequisites for the DE algorithm's iterative process. However, in the event that the probability exceeds the predetermined threshold percentage, the process is subject to modification. It is evident that the individuals will gravitate toward the most distant ones. Establishing a balance criterion is imperative for selecting individuals to be used by the mutation operator. As the Mahalanobis distance decreases, the confidence interval will include only the closest ones. As the iterations progress, the encapsulation will become smaller. This is done to accentuate the algorithm's direction of convergence. The novelty of this approach is that, if the algorithm does not identify individuals close to the optimal solution, it will search beyond the selection threshold to expand its exploration. In any case, to ensure the validity of the DE mutation operation, if the group selected based on Mahalanobis contains fewer than three individuals, the entire population is used as the mutation set. In other words, individuals are selected at random. This safeguard guarantees that at least three distinct individuals will be chosen during the mutation process. This is intended to prevent the algorithm from stagnating. It has been demonstrated that both the EDA [22] and CMA-ES [23] algorithms employ variants of these probability-based techniques, yielding favorable results. The employment of this advanced technique provides a novel perspective on data distribution and correlation. The pseudocode for this algorithm is delineated in Algorithm 1. Finally, by leveraging the distance measure derived from the reduced Cholesky decomposition, an adaptive adjustment is implemented for the crossover values and the scaling factor. This adjustment is performed at each iterative cycle. The objective of this adjustment is to treat the distance as an adaptive control parameter. This objective is realized through the implementation of the following formula:

$$\begin{aligned} F &= \text{rand} \cdot (0.60 + (1 - d_M)), \\ p_{cr} &= 0.1 + 0.25\,(1 - d_M) \end{aligned} \tag{12}$$

In such cases, the Mahalanobis distance (d_M) emerges as the pivotal factor in assessing the adequacy of the parameters. This adaptive control is inversely proportional to the population distance, promoting exploration at low levels and exploitation at high levels. The initial control parameters are defined based on the configuration obtained from the sensitivity analysis detailed in Supplementary Material Note S8.

Algorithm 1 Differential evolution with Mahalanobis distance and Cholesky decomposition

Parameter settings: $beta_{min} = 0.10$, $beta_{max} = 0.60$, $p_{cr} = 0.10$, $q = 0.5$
Population Initialization
$x_i^g = \{x_{i,1}^g, x_{i,2}^g \ldots, x_{i,d}^g\}$, $i = 1, \ldots, Np$
Calculate μ and Σ of entire population
Compute the Mahalanobis Distance with respect to x_{best} ▷ Equation (8)
Calculate **the Cholesky decomposition** instead of Σ^{-1} ▷ Equation (9)/(10)
Mahalanobis-based grouping and selection:
Compute $\tau \leftarrow F_{\chi^2}^{-1}(q; v)$ ▷ Equation (11)
$close_individuals \leftarrow \{ p_i \in population \mid \triangle_i^2 \leq \tau \}$
$far_individuals \leftarrow \{ p_i \in population \mid \triangle_i^2 > \tau \}$
if $|close_individuals| \geq 3$ **then**
    Randomly select 3 distinct indices from $close_individuals$
    $x_1, x_2, x_3 \leftarrow$ selected close individuals
else if $|far_individuals| \geq 3$ **then**
    Randomly select 3 distinct indices from $far_individuals$
    $x_1, x_2, x_3 \leftarrow$ selected far individuals
else
    Randomly select 3 different individuals from the complete population
end if
Proceed to Mutation with the 3 selected individuals:
$v_i^{g+1} = x_{r3}^g + F(x_{r1}^g - x_{r2}^g)$
Crossover
$u_i^{g+1} = u_d^g \begin{cases} v_{i,d}^g, & \text{if } d = d_{rand} \text{ or } rand(0,1) \leq p_{cr} \\ x_{i,d}^g, & \text{otherwise} \end{cases}$
Selection
$x_i^{g+1} = \begin{cases} u_{i,d}^g, & \text{if } f(u_i^{g+1}) \leq f(x_i^g), \\ x_{i,d}^g, & \text{otherwise,} \end{cases}$
Adaptive Parameter Control of F and p_{cr} ▷ Equation (12)

In order to facilitate the understanding of the proposal's implementations, a block diagram has been developed that illustrates the internal process within the DE algorithm. The implementations in question are illustrated in Figure 1. As illustrated, the fundamental implementation arises immediately subsequent to the population initialization calculation, wherein the Mahalanobis distance is determined through Cholesky decomposition. The grouping of individuals into groups of close and far candidates according to the best solution so far serves to guide the mutation process. Subsequent to the crossover process, the solutions are evaluated using the wrapper-based objective function to finally select the best subsets of features. Based on the distance calculated in the current iteration, the scaling factor and the crossover range are modified.

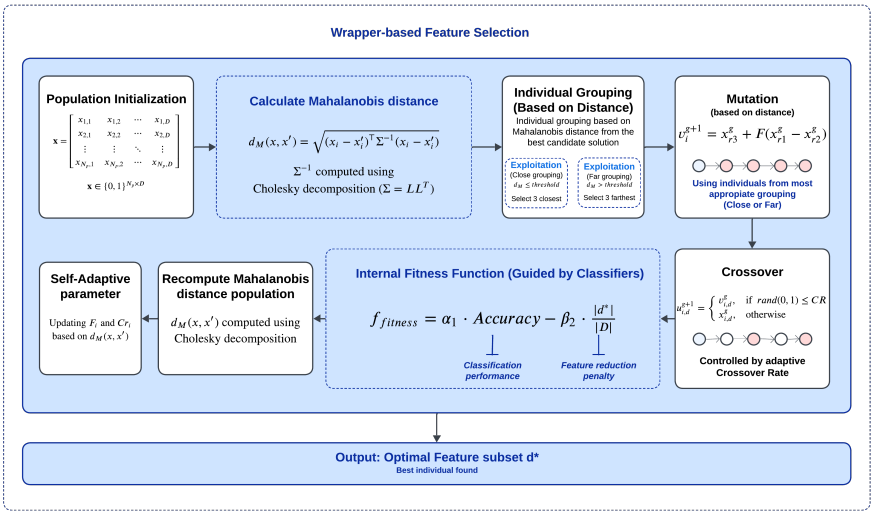

Figure 1. Particular framework of the proposed MaCRO-DE Algorithm. Note: d^* denotes the selected feature subset, as defined in Section 4.2.

4. The Proposed Binary Approach for Feature Selection

This section details the implementation of the proposal's binary version of the feature selection algorithm. As previously stated, the objective of feature selection is to identify and retain the most pertinent and significant features from the original dataset, while discarding those that are least relevant or redundant. In such instances, it is evident that a binary representation of the solutions to a problem in the continuous domain must be converted. In this scenario, binary optimization evaluates the viability of solutions and determines their subsequent use. Consequently, for each dimension, the solution will assume a value of 1, indicating selection of the feature, or 0, denoting non-selection. Prior studies have explored the application of transfer functions, underscoring their efficacy in converting continuous to binary optimization algorithms [13].

4.1. Transfer Function for Binarization

In the context of feature selection, numerous methodologies have been proposed to transform continuous optimization algorithms into binary form. This procedure is frequently referred to as "binarization". Among the suite of transfer functions, some have gained prominence for their simplicity and effectiveness in balancing exploration and exploitation during binarization. These functions map the continuous outputs of metaheuristic operators to probabilistic binary values, thereby enabling continuous optimization algorithms to be applied to problems such as feature selection. Specifically, this study analyzes eight transfer functions that are widely used and studied in the literature for the binarization process. These transfer functions are classified into two distinct categories: four V-shaped functions and four S-shaped functions. Their mathematical formulation and graphical behavior are described in detail in the Supplementary Note S2. Subsequently, experimental tests are conducted to evaluate which function offers the best performance for the framework of this improvement.

4.2. Assessment of the Fitness Function for the Feature Selection Problem

In accordance with the preceding remarks, a primary objective of the feature selection problem is to identify the minimum set of features. Conversely, it is also the aim to maximize the accuracy of the classifier or minimize the classification error. Consequently, this problem is regarded as a bi-objective problem by its inherent nature. It is imperative to consider a fitness function that incorporates these two aspects to effectively address the problem. The fitness function that combines both factors is shown below.

$$Fitness = \alpha_1 \cdot (1 - Accuracy) + \beta_2 \cdot \frac{|d^*|}{|D|} \quad (13)$$

In this context, *Accuracy* denotes the accuracy achieved by the classifier. In this instance, the K-Nearest Neighbor (KNN) and Support Vector Machine (SVM) classifiers are employed. A detailed description and configuration of these classifiers can be found in Supplementary Note S3. The number of features selected within the entire space $|D|$ of total features is denoted by $|d^*|$. The values of α and β are constrained to the interval $[0, 1]$. For general purposes, initial values were selected in accordance with previous literature that has refuted them [13]. However, for the purposes of this research, a sensitivity analysis was conducted to determine these values for the implementation of the proposal. The results of this analysis can be found in Supplementary Note S11.

4.3. Validation Protocol for Wrapper-Based Feature Selection

In order to prevent loss of information during the process of feature selection, each dataset was initially separated into training and test partitions using an 80/20 division. A split was performed with a fixed random state whenever the class distribution enabled it. The external test partition was excluded from the feature selection process. The feature selection process was executed exclusively on the training partition, which was further subdivided into training and validation subsets within the optimization process to assess the performance of candidate feature subsets. Consequently, the metaheuristic search and the calculation of the objective function were executed without access to the external test data. Subsequent to optimization, the selected feature subset was evaluated on the previously unseen test partition, from which accuracy, F1-score, precision, and recall were obtained. For a more detailed explanation, relevant information about this process has been added to Supplementary Note S4.

4.4. General Framework of the Proposed Wrapper-Based Feature Selection Methodology

The methodological framework comprises a series of essential phases for conducting the entire feature selection process using the wrapper-based method and machine learning-based classification. The proposed methodology incorporates the MaCRO-DE algorithm as a process to enhance the overall methodology. This approach considers improvements implemented in the Mahalanobis distance and Cholesky decomposition. These improvements benefit the generation of subset features prior to classification. This general framework is illustrated in Figure 2. The diagram offers a comprehensive representation of the methodological framework, delineating its sequential phases. The sequence of operations begins with the collection, preprocessing, and extraction of features from the data. These fundamental processes are necessary and serve as a prerequisite for the subsequent optimization phase, which employs a wrapper-based method. Subsequently, the implementation of the proposed MaCRO-DE algorithm begins. This algorithm is responsible for searching for the optimal subset of features while incorporating the implemented improvements. In light of the foundational significance of this process, the preceding section provided a comprehensive overview and elaborated on this phase (see Figure 1). Subsequent to obtaining the optimal subset, the classification stage is initiated using the classifiers considered in this methodology, specifically SVM and KNN. This is due to the fact that, based on the relevant literature, these classifiers have achieved optimal performance in a series of problems to which they have been applied. These processes yield the results necessary to calculate the performance metrics (i.e., Accuracy, F1-score, Precision, and Recall), as well as the fitness values of the objective function and the feature subset. This process is repeated multiple times to ensure the statistical validity of the results. Consequently, a range of statistical analyses and graphs are generated, including convergence curves, box plots, heat maps, and

violin plots. The objective of this study is to compare the proposed MaCRO-DE mechanism with other evolutionary algorithm methodologies and to assess its effectiveness.

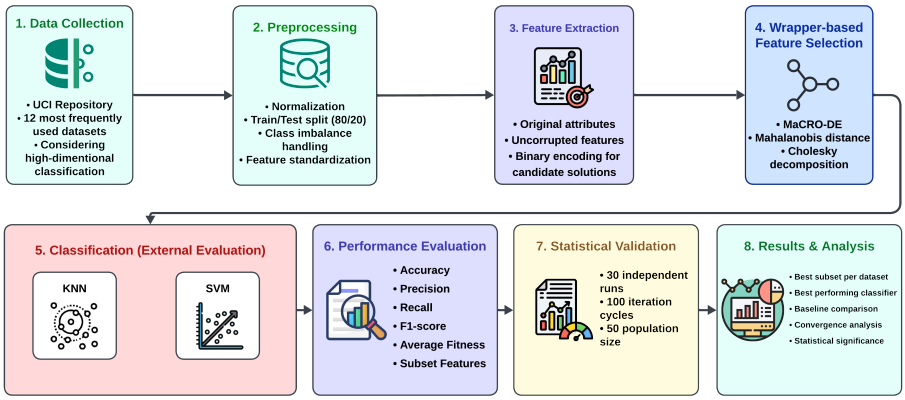

Figure 2. General framework of the proposed MaCRO-DEAlgorithm for wrapper-based feature selection.

5. Experimental Design

This section presents the experimental results obtained by implementing the proposed methodology and algorithms for comparison purposes. The section is divided into two parts: the first part is a comparative analysis of the CEC 2017 benchmark suite, which was used to demonstrate the performance of the algorithms on test functions for global optimization problems. The second part is dedicated to the implementation of feature selection. For this purpose, various datasets were selected for analysis. These datasets are used to compare the proposed algorithm with various metaheuristic algorithms to validate its performance. An ablation study, a parameter sensitivity analysis, and a final weight variable sensitivity analysis of the objective function were conducted. These analyses can be found in the supplementary material, in the respective sections.

5.1. Parameter Settings for Experimental Tests

This section delineates the experimental parameters employed to validate the efficacy and performance of the proposal. The configuration of these parameters was implemented to guarantee a fair comparison among the metaheuristic algorithms utilized. The evaluation was conducted using the CEC-2017 benchmark suite and focused on the performance of 30-dimensional problems. The population size was set at 50 individuals, 2000 epochs, and each function and algorithm was evaluated 30 times. The statistical outcomes were summarized using the best, worst, mean, and standard deviation values obtained from the independent runs. The experiments were executed on a GPU-based configuration on a Linux/Ubuntu workstation equipped with an NVIDIA GeForce RTX 3060 GPU with 12 GB RAM. The comparison is based on a total of thirteen algorithms, including both classic and contemporary ones: the Battle Royale Optimization (BRO) [24], the Dung Beetle Optimizer (DBO) [25], the Differential Evolution (DE) [16], the Dwarf Mongoose Optimization Algorithm (DMOA) [26], the Grey Wolf Optimizer (GWO) [27], the Harris Hawks Optimization (HHO) [28], the Moth-Flame Optimization (MFO) [29], the Moss Growth Optimization (MGO) [30], the Particle Swarm Optimization (PSO) [31], the Success-History-based Adaptive Differential Evolution (SHADE) [32], the Whale Optimization Algorithm (WOA) [33], Adaptive Differential Evolution with Optional External Archive (JADE) [34] and Ficks Law Algorithm (FLA) [35]. The proposed algorithm employed a final parameter configuration that was obtained from a sensitivity analysis. The algorithms utilized for the comparative analysis were constrained by the parameters delineated in their original formulations. To facilitate the replicability of the experiment, the parameters are outlined in Table 1.

Table 1. Parameter settings used in the CEC 2017 experiments for all algorithms.

Category	Parameter	Value
General (all algorithms)	Independent runs	30
	Population size	50
	Dimensions	30
	Maximum epochs	2000
Algorithm-specific settings		
MaCRO-DE	Minimum scaling parameter $beta_{min}$	0.10
	Maximum scaling parameter $beta_{max}$	0.60
	Crossover probability p_{cr}	0.10
	Mahalanobis probability level q	0.50
BRO	Threshold	3
	Remaining parameters	Default
DBO	Behavioral strategies	Original
	Remaining parameters	Default
DE	Scaling factor F	0.8
	Crossover rate Cr	0.2
DMO	Number of babysitters	3
	Peep parameter	2
	Remaining parameters	Default
GWO	Control parameter a	$2 \rightarrow 0$
HHO	Escaping energy E	$2E_0(1 - t/T)$
MFO	Number of flames	$N \rightarrow 1$
MGO	Search control parameters	Adaptive
PSO	Inertia weight w	0.4
	Cognitive coefficient c_1	2.05
	Social coefficient c_2	2.05
SHADE	Initial memory value M_F	0.5
	Initial memory value M_{Cr}	0.5
WOA	Control parameter a	$2 \rightarrow 0$
	Spiral constant b	1
JADE	Parameter configuration	Default
FLA	Parameter configuration	Default

5.2. Comparative Results of the CEC-2017 Benchmark

A comprehensive comparative evaluation was conducted using the CEC-2017 benchmark suite to evaluate the optimization capability of the proposed MaCRO-DE algorithm across different search landscapes. The experiments included functions F1 through F29 and compared MaCRO-DE with a diverse set of well-established metaheuristic optimizers under the same computational conditions. The evaluation primarily focuses on the average fitness obtained across independent runs. For this purpose, the values shown in bold indicate the best result obtained for the method in each case using the given measure. The supplementary material provides additional statistics, such as best, worst, and standard deviation values, for a more detailed analysis of robustness and stability. This experiment aims to determine if the mechanisms incorporated into MaCRO-DE improve the balance between exploration and exploitation in functions with varying levels of complexity. The comparative results are summarized in Table 2, and the corresponding convergence behavior is illustrated in Figures 3 and 4. These results provide an overview of MaCRO-DE's and competing algorithms' relative performance across the CEC-2017 test set.

Table 2. Mean fitness values obtained on the CEC-2017 benchmark functions.

Function	MaCRO-DE	BRO	DBO	DE	DMOA	GWO	HHO	MFO	MGO	PSO	SHADE	WOA	JADE	FLA
F1	1.14×10^{2}	2.87×10^{10}	1.95×10^{6}	1.97×10^{10}	2.91×10^{6}	2.59×10^{8}	2.04×10^{6}	3.25×10^{8}	2.25×10^{3}	4.36×10^{6}	$\mathbf{1.00 \times 10^{2}}$	5.72×10^{8}	$\mathbf{1.00 \times 10^{2}}$	1.92×10^{7}
F2	$\mathbf{2.00 \times 10^{2}}$	5.63×10^{4}	2.76×10^{2}	6.26×10^{4}	5.97×10^{3}	1.92×10^{3}	2.20×10^{2}	3.43×10^{3}	2.00×10^{2}	2.00×10^{2}	$\mathbf{2.00 \times 10^{2}}$	1.06×10^{4}	$\mathbf{2.00 \times 10^{2}}$	2.94×10^{2}
F3	3.32×10^{2}	5.31×10^{3}	4.41×10^{2}	3.39×10^{3}	3.32×10^{2}	4.03×10^{2}	4.07×10^{2}	4.52×10^{2}	3.44×10^{2}	3.79×10^{2}	$\mathbf{3.25 \times 10^{2}}$	5.10×10^{2}	$\mathbf{3.29 \times 10^{2}}$	3.67×10^{2}
F4	6.23×10^{2}	3.70×10^{4}	8.13×10^{2}	2.67×10^{4}	6.38×10^{2}	1.17×10^{3}	1.27×10^{3}	4.64×10^{3}	5.50×10^{2}	1.14×10^{3}	$\mathbf{4.30 \times 10^{2}}$	8.36×10^{3}	$\mathbf{4.87 \times 10^{2}}$	6.68×10^{2}
F5	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$	$\mathbf{5.00 \times 10^{2}}$
F6	8.61×10^{2}	1.31×10^{6}	1.52×10^{3}	7.20×10^{5}	1.03×10^{3}	1.73×10^{4}	1.48×10^{3}	3.22×10^{4}	8.86×10^{2}	2.01×10^{3}	$\mathbf{6.60 \times 10^{2}}$	1.18×10^{5}	$\mathbf{7.28 \times 10^{2}}$	1.68×10^{3}
F7	1.08×10^{3}	1.28×10^{3}	1.08×10^{3}	9.44×10^{2}	1.04×10^{3}	8.92×10^{2}	1.17×10^{3}	9.90×10^{2}	$\mathbf{8.31 \times 10^{2}}$	1.02×10^{3}	$\mathbf{7.72 \times 10^{2}}$	1.46×10^{3}	9.03×10^{2}	9.32×10^{2}
F8	$\mathbf{8.00 \times 10^{2}}$	8.21×10^{2}	8.09×10^{2}	8.12×10^{2}	8.00×10^{2}	8.02×10^{2}	8.11×10^{2}	8.06×10^{2}	8.03×10^{2}	8.08×10^{2}	8.00×10^{2}	8.18×10^{2}	8.00×10^{2}	8.08×10^{2}
F9	9.00×10^{3}	7.56×10^{3}	4.94×10^{3}	4.63×10^{3}	8.72×10^{3}	7.96×10^{3}	5.42×10^{3}	4.86×10^{3}	$\mathbf{2.37 \times 10^{3}}$	3.86×10^{3}	$\mathbf{1.17 \times 10^{3}}$	6.38×10^{3}	7.79×10^{3}	2.69×10^{3}
F10	$\mathbf{1.22 \times 10^{3}}$	8.10×10^{5}	1.45×10^{5}	2.58×10^{6}	8.84×10^{4}	7.74×10^{4}	7.90×10^{4}	2.68×10^{4}	1.65×10^{4}	1.93×10^{4}	$\mathbf{1.23 \times 10^{3}}$	1.80×10^{5}	1.64×10^{3}	2.90×10^{4}
F11	1.44×10^{5}	2.05×10^{9}	1.18×10^{7}	2.09×10^{9}	6.02×10^{6}	1.50×10^{7}	1.71×10^{7}	3.48×10^{5}	1.90×10^{4}	1.42×10^{4}	$\mathbf{3.86 \times 10^{3}}$	2.39×10^{6}	$\mathbf{4.40 \times 10^{3}}$	1.12×10^{6}
F12	$\mathbf{2.31 \times 10^{3}}$	1.70×10^{9}	2.22×10^{6}	2.06×10^{9}	6.62×10^{4}	1.66×10^{6}	1.29×10^{6}	1.92×10^{5}	4.14×10^{3}	1.23×10^{4}	$\mathbf{1.73 \times 10^{3}}$	7.48×10^{5}	3.96×10^{3}	2.37×10^{5}
F13	$\mathbf{2.04 \times 10^{3}}$	1.14×10^{6}	3.53×10^{5}	3.89×10^{6}	3.81×10^{5}	1.88×10^{5}	1.00×10^{6}	2.46×10^{5}	7.99×10^{4}	9.17×10^{4}	$\mathbf{1.59 \times 10^{3}}$	4.71×10^{5}	7.13×10^{3}	7.31×10^{4}
F14	5.40×10^{3}	4.79×10^{8}	5.79×10^{5}	3.80×10^{8}	7.60×10^{4}	1.98×10^{5}	4.67×10^{5}	3.53×10^{4}	1.57×10^{4}	1.72×10^{4}	$\mathbf{2.32 \times 10^{3}}$	7.84×10^{6}	$\mathbf{2.30 \times 10^{3}}$	1.08×10^{5}
F15	$\mathbf{7.32 \times 10^{2}}$	3.02×10^{7}	3.78×10^{5}	1.05×10^{8}	2.62×10^{4}	7.80×10^{5}	1.26×10^{5}	2.54×10^{4}	1.97×10^{3}	2.11×10^{3}	1.18×10^{3}	3.58×10^{5}	1.23×10^{3}	4.31×10^{3}
F16	$\mathbf{1.62 \times 10^{3}}$	4.29×10^{8}	3.57×10^{4}	1.52×10^{9}	1.11×10^{5}	3.82×10^{4}	1.47×10^{4}	7.68×10^{3}	2.56×10^{3}	2.73×10^{3}	$\mathbf{1.78 \times 10^{3}}$	5.75×10^{4}	2.01×10^{3}	3.31×10^{3}
F17	$\mathbf{2.53 \times 10^{3}}$	1.21×10^{6}	1.54×10^{5}	2.01×10^{6}	2.24×10^{5}	9.13×10^{4}	1.13×10^{5}	5.17×10^{4}	4.70×10^{4}	4.10×10^{4}	$\mathbf{1.93 \times 10^{3}}$	7.32×10^{4}	3.11×10^{3}	5.19×10^{4}
F18	1.85×10^{3}	4.56×10^{10}	1.42×10^{5}	1.50×10^{12}	1.93×10^{4}	1.61×10^{7}	1.45×10^{5}	4.99×10^{4}	7.25×10^{3}	1.89×10^{4}	$\mathbf{1.81 \times 10^{3}}$	4.14×10^{6}	$\mathbf{1.84 \times 10^{3}}$	1.93×10^{5}
F19	$\mathbf{1.58 \times 10^{3}}$	5.67×10^{3}	2.52×10^{3}	3.58×10^{3}	$\mathbf{1.59 \times 10^{3}}$	1.97×10^{3}	6.05×10^{3}	2.99×10^{3}	1.83×10^{3}	2.14×10^{3}	1.64×10^{3}	9.71×10^{3}	1.63×10^{3}	1.95×10^{3}
F20	2.31×10^{3}	1.30×10^{4}	2.37×10^{3}	1.68×10^{4}	2.38×10^{3}	2.93×10^{3}	2.94×10^{3}	5.56×10^{3}	2.26×10^{3}	2.72×10^{3}	$\mathbf{2.19 \times 10^{3}}$	9.26×10^{3}	$\mathbf{2.24 \times 10^{3}}$	2.39×10^{3}
F21	1.52×10^{4}	1.52×10^{4}	1.49×10^{4}	1.52×10^{4}	1.50×10^{4}	1.52×10^{4}	1.46×10^{4}	1.50×10^{4}	7.58×10^{3}	1.25×10^{4}	2.05×10^{3}	1.40×10^{4}	$\mathbf{1.37 \times 10^{3}}$	$\mathbf{1.97 \times 10^{3}}$
F22	2.43×10^{3}	3.24×10^{4}	2.67×10^{3}	2.68×10^{4}	2.53×10^{3}	3.93×10^{3}	2.70×10^{3}	5.63×10^{3}	2.45×10^{3}	2.74×10^{3}	$\mathbf{2.42 \times 10^{3}}$	8.14×10^{3}	$\mathbf{2.43 \times 10^{3}}$	2.88×10^{3}
F23	2.54×10^{3}	1.98×10^{4}	2.61×10^{3}	1.57×10^{4}	2.58×10^{3}	3.60×10^{3}	2.73×10^{3}	3.86×10^{3}	2.51×10^{3}	2.64×10^{3}	$\mathbf{2.50 \times 10^{3}}$	6.19×10^{3}	$\mathbf{2.50 \times 10^{3}}$	2.85×10^{3}
F24	2.82×10^{3}	4.42×10^{3}	2.87×10^{3}	3.93×10^{3}	2.82×10^{3}	2.87×10^{3}	2.92×10^{3}	2.92×10^{3}	2.83×10^{3}	2.85×10^{3}	$\mathbf{2.82 \times 10^{3}}$	3.01×10^{3}	$\mathbf{2.82 \times 10^{3}}$	2.83×10^{3}
F25	$\mathbf{3.33 \times 10^{3}}$	5.58×10^{3}	3.50×10^{3}	4.36×10^{3}	3.33×10^{3}	3.36×10^{3}	3.68×10^{3}	3.44×10^{3}	3.37×10^{3}	3.42×10^{3}	3.33×10^{3}	4.68×10^{3}	3.33×10^{3}	3.35×10^{3}
F26	3.11×10^{3}	3.56×10^{3}	3.20×10^{3}	3.26×10^{3}	$\mathbf{3.10 \times 10^{3}}$	3.10×10^{3}	3.63×10^{3}	3.21×10^{3}	3.14×10^{3}	3.17×10^{3}	$\mathbf{3.09 \times 10^{3}}$	3.66×10^{3}	3.10×10^{3}	3.12×10^{3}
F27	$\mathbf{2.70 \times 10^{3}}$	4.59×10^{3}	3.18×10^{3}	3.83×10^{3}	2.79×10^{3}	3.11×10^{3}	3.10×10^{3}	3.14×10^{3}	2.73×10^{3}	2.90×10^{3}	$\mathbf{2.71 \times 10^{3}}$	3.33×10^{3}	2.71×10^{3}	2.92×10^{3}
F28	$\mathbf{5.60 \times 10^{3}}$	6.69×10^{9}	2.20×10^{8}	6.20×10^{9}	5.17×10^{6}	1.89×10^{6}	3.64×10^{5}	6.79×10^{4}	9.60×10^{3}	1.87×10^{7}	$\mathbf{5.36 \times 10^{3}}$	1.79×10^{8}	5.88×10^{3}	7.12×10^{4}
F29	9.68×10^{3}	1.26×10^{10}	6.19×10^{6}	7.37×10^{9}	8.95×10^{4}	1.65×10^{7}	6.92×10^{6}	9.43×10^{4}	1.93×10^{4}	1.59×10^{6}	$\mathbf{6.43 \times 10^{3}}$	1.22×10^{7}	$\mathbf{9.36 \times 10^{3}}$	1.17×10^{6}

Note: Bold values highlight the best-performing results; lower values are better.

The complete CEC-2017 benchmark results are summarized in Table 2. This table lists the mean performance scores obtained by MaCRO-DE and competing metaheuristic algorithms for functions F1–F29. While no single method stands out across the entire test suite, MaCRO-DE achieves highly competitive mean values on several functions, including F2, F5, F8, F10, F15, F16, F19, F24, F25, and F27. Furthermore, MaCRO-DE remains close to the best-performing method on other functions, demonstrating its ability to perform competitively across the different characteristics of the CEC-2017 search landscapes. These results provide further evidence of the robustness and stability of the compared algorithms. While SHADE and JADE achieve higher scores on some functions, MaCRO-DE exhibits robust convergence and low variability in several cases. This supports the effectiveness of the proposed search mechanisms and demonstrates that their relative performance depends on the characteristics of each benchmark function.

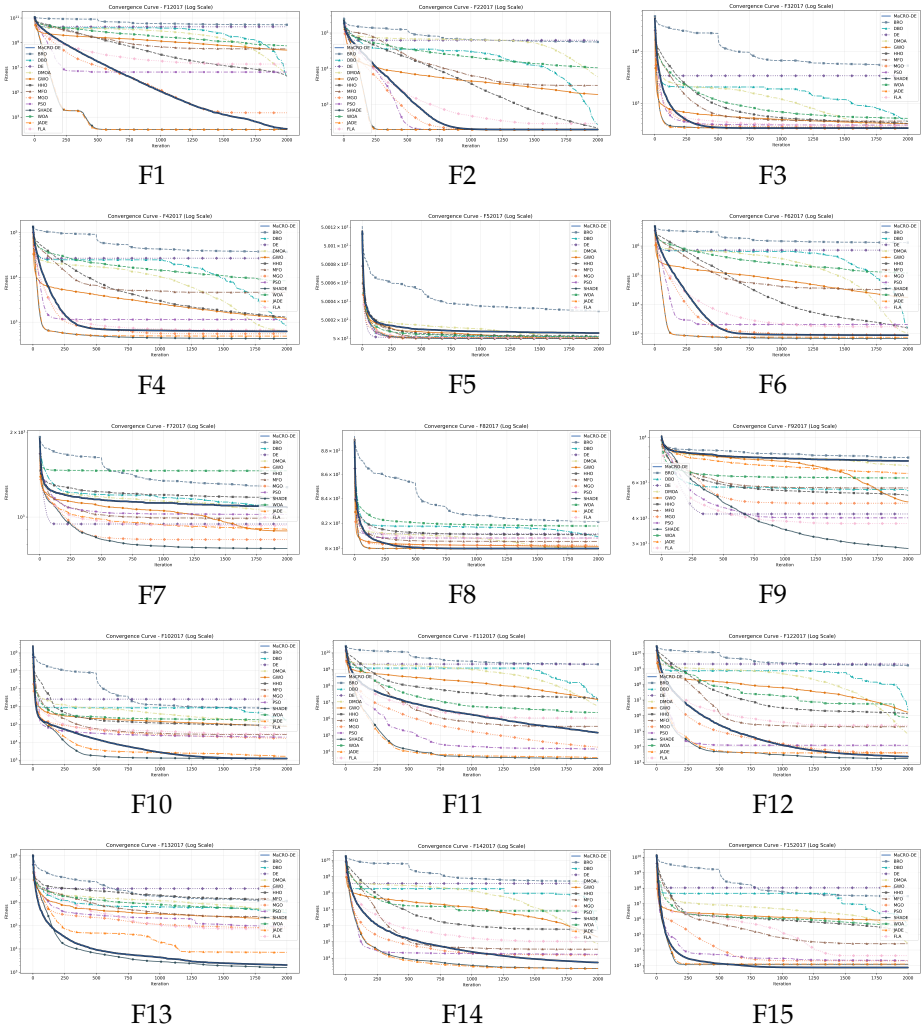

Figure 3. Convergence curves in CEC-2017 benchmark functions (F1–F15).

5.3. Friedman Rank Test

To complement the numerical comparison of the CEC-2017 benchmark suite, a statistical analysis was conducted using the Friedman rank-sum test [36,37]. This nonparametric test is commonly used to determine whether there are statistically significant differences among various optimization algorithms evaluated on the same set of problems. Since metaheuristic algorithms are stochastic in nature, the use of a rank-based statistical analysis provides additional evidence beyond the direct comparison of fitness values.

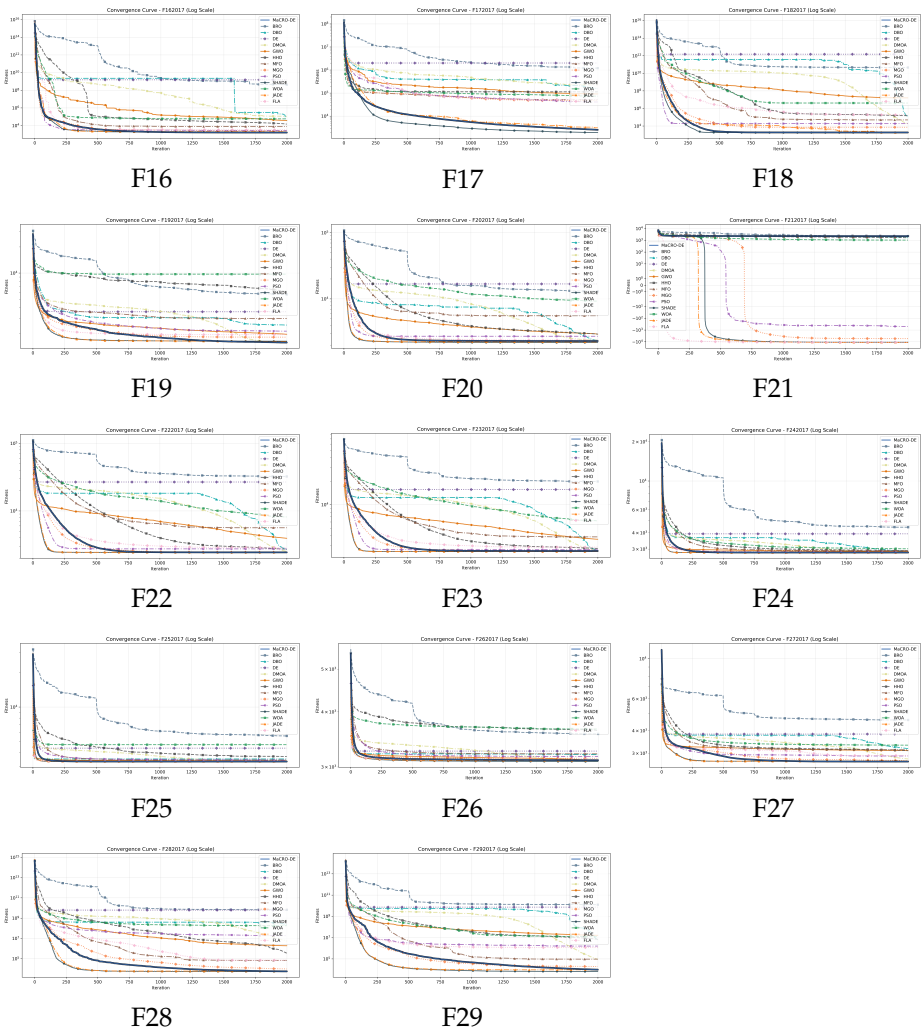

Figure 4. Convergence curves in CEC-2017 benchmark functions (F16–F29).

The Friedman test was performed by considering the average final fitness obtained by each algorithm across the 29 functions in the CEC-2017 test suite. Table 3 presents the corresponding mean ranks and final positions in the ranking. MaCRO-DE obtained the best mean rank, followed by JADE, while SHADE ranked third among the fifteen evaluated algorithms. The Friedman test yielded a p-value of 8.01×10^{-53}, considerably lower than the significance level of 0.05. Therefore, the null hypothesis that the algorithms' performance is equivalent is rejected, confirming the existence of statistically significant differences among the evaluated methods.

5.4. Experimental Results for Feature Selection Problems

This section presents the comparative results obtained using the binary version of the proposed algorithm for the feature selection problem. The experimental analysis focuses on the final configuration of the method and how it compares to the selected metaheuristic algorithms on the evaluated benchmark datasets. The transfer function analysis used to determine the binarization strategy is presented in the supplementary materials (Supplementary Note S10). Additionally, a sensitivity study of the objective function weight parameters is included in Supplementary Note S11. Thus, this section focuses on the experimental parameters, evaluation criteria, and comparative performance of the algorithm for wrapper-based feature selection.

Table 3. Friedman test results for the CEC-2017 benchmark.

Algorithm	Friedman Rank	Final Rank
MaCRO-DE	$\mathbf{1.66 \times 10^{0}}$	**1**
SHADE	3.00×10^{0}	2
JADE	4.28×10^{0}	3
MGO	4.45×10^{0}	4
PSO	7.14×10^{0}	5
FLA	7.31×10^{0}	6
DMOA	7.52×10^{0}	7
MFO	9.10×10^{0}	8
GWO	9.59×10^{0}	9
DBO	9.79×10^{0}	10
HHO	1.04×10^{1}	11
WOA	1.22×10^{1}	12
DE	1.33×10^{1}	13
BRO	1.43×10^{1}	14
p-value	8.01×10^{-53}	

Note: Bold values indicate the algorithm with the lowest Friedman rank and its corresponding final rank.

5.4.1. Setting Parameters and Dataset for Feature Selection

In order to evaluate the performance of the proposed algorithm for wrapper-based feature selection, twelve benchmark datasets that are commonly used in the literature were selected from the UCI repository [38]. These datasets exhibit different characteristics in terms of dimensionality, number of instances, and number of classes, providing heterogeneous classification scenarios for evaluation. Table 4 summarizes the main properties of the datasets, including the number of instances, attributes, and target classes. The same population size and number of independent runs were used for all comparative experiments to ensure equivalent experimental conditions among the evaluated algorithms. Each feature selection experiment was run for 150 iterations and repeated 30 times to reduce the effect of stochastic variability. The experiments were conducted on the same computational platform used for the CEC-2017 benchmark tests. Feature selection was performed exclusively on the training data in accordance with the validation protocol described above. The external test partition remained hidden during the optimization process. The final feature subset obtained in each run was subsequently evaluated on the corresponding test partition. The objective function weighting parameters, α_1 and β_2, were selected based on the sensitivity analysis presented in the Supplementary Materials.

Table 4. Detailed description of the dataset used for feature selection.

Dataset	Instances	Attributes	Classes
Breast Cancer Wisc. (Original)	699	9	2
Breast Cancer Wisc. (Diagnostic)	569	30	2
Glass	214	9	6
HeartEW	270	13	2
Ionosphere	351	34	2
Lymphography	148	18	4
Sonar	208	60	2
SpectEW	267	22	2
Tic-tac-toe	958	9	2
Wine	178	13	3
WaveformEW	5000	40	3
Zoo	101	16	7

5.4.2. Evaluation Criteria

In this section, the performance of the proposal's feature selection is evaluated. To accomplish this objective, it is necessary to establish evaluation criteria aligned with the specific problem to be addressed. In this study, four distinct performance metrics were employed to assess the performance of two machine learning models: Accuracy, F1-Score, Recall, and Precision, as presented in Equations (14)–(17), respectively. To calculate these metrics, four distinct concepts were considered: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). Subsequently, Accuracy is calculated as the proportion of true positive predictions relative to the total number of correct predictions. This metric assesses the algorithm's ability to predict positive cases. The F1-Score is defined as the harmonic mean of precision and recall; it combines both metrics into a single measure of their balance. Precision is defined as the proportion of correct predictions relative to the total number of cases in the evaluated set. Finally, Recall demonstrates the proportion of true positives and false negatives, which correspond to the actual number of positive cases.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{15}$$

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

Consequently, two performance indicators are derived: fitness value and number of feature subsets. Due to the experimental nature of the testing, in which experiments are conducted multiple times, these indicators are calculated as averages in each case.

$$Accuracy_{Avg} = \frac{1}{M} \sum_{j=1}^{M} \left(\frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(C_{ij} = L_i) \right) \tag{18}$$

Secondly, the average fitness value is considered. As previously indicated, this value is determined by the fitness function and used in the algorithm's optimization process.

$$Fitness_{Avg} = \frac{1}{M} \sum_{j=1}^{M} Fit_i^* \tag{19}$$

Finally, the average of the selected subset of features is calculated. This approach enables the representation of the algorithm's performance as the number of features selected to address the feature selection problem.

$$Features_{subset} = \frac{1}{m} \sum_{i=1}^{M} \frac{d_i^*}{D} \tag{20}$$

5.4.3. Comparison of the Performance of MaCRO-DE Wrapper-Based Feature Selection

This section evaluates the performance of the proposed MaCRO-DE algorithm for wrapper-based feature selection. The analysis focuses on classification performance, feature reduction, computational cost, and the overall balance achieved by the metaheuristic algorithms on the considered datasets. First, graphical results are examined to provide an overview of comparative behavior. Then, a numerical analysis of average accuracy and average number of selected features is performed. Results obtained from twelve

feature selection datasets demonstrate that no single metaheuristic consistently dominates all classification problems, reflecting the datasets' heterogeneous features. The graphical analysis in Figure 5 confirms that KNN and SVM can achieve competitive classification performance with subsets generated by the evaluated optimization methods. Similarly, the F1-score heatmap in Figure 6 illustrates that the relative performance of the algorithms varies across datasets. Additionally, Figure 7 reveals clear differences in the trade-off between dimensionality reduction and computational effort.

The mean accuracy results presented in Table 5 further support this behavior. While MaCRO-DE does not achieve the highest mean accuracy on all datasets, its performance remains competitive across the entire benchmark set. In general, its performance is close to that of the best-performing method. This suggests that the proposed search mechanism can identify compact subsets while retaining a significant amount of predictive information from the original feature space. The feature selection results in Table 6 also show that the algorithm systematically reduces the dimensionality of the evaluated datasets. Specifically, it achieves the lowest average number of selected features on the Zoo dataset and remains close to the most compact solutions on several other problems. Therefore, the performance of the proposal should not be interpreted solely in terms of achieving the highest classification score or the smallest subset, but rather in terms of balancing predictive performance, dimensionality reduction, and computational effort.

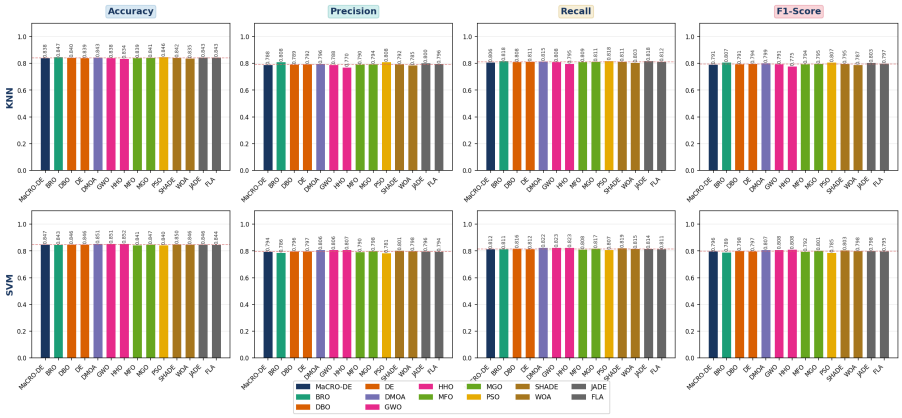

Figure 5. Classification performance obtained using KNN and SVM across all evaluated datasets and performance metrics.

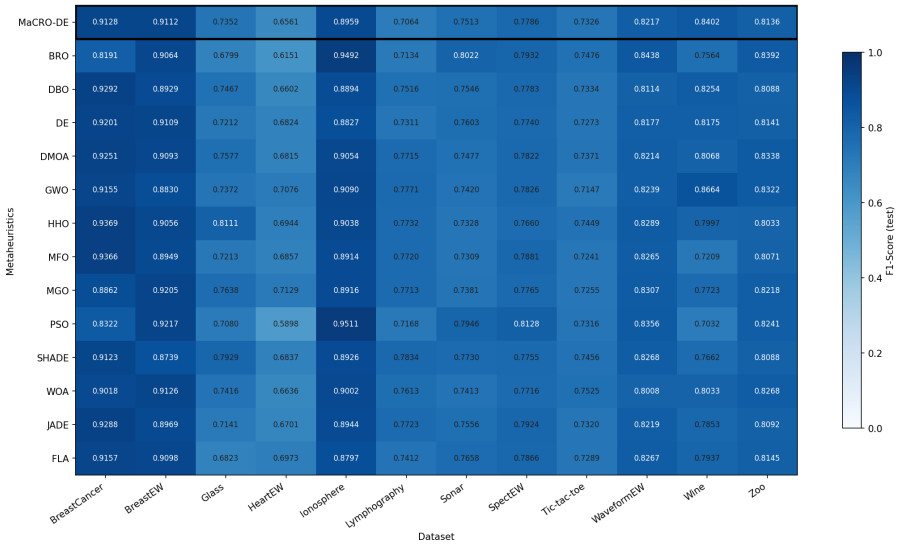

Figure 6. F1-score performance of the evaluated metaheuristic algorithms across all feature-selection datasets.

Table 5. Mean classification Accuracy (%) obtained by the evaluated algorithms across the feature-selection datasets.

Dataset	MaCRO-DE	BRO	DBO	DE	DMOA	GWO	HHO	MFO	MGO	PSO	SHADE	WOA	JADE	FLA
BreastCancer	92.6310	87.8333	92.6071	92.2619	92.5357	92.4524	**93.2738**	92.3333	89.8929	90.0000	92.7738	92.0714	92.6786	91.4881
BreastEW	91.4474	92.0760	89.8830	89.6491	91.3012	90.4240	91.1842	88.4795	90.9942	**92.5439**	88.8889	90.5117	89.6784	91.0526
Glass	89.4583	90.5985	91.6577	91.0699	**93.5803**	92.4437	90.7581	90.3380	91.7063	92.6411	92.9124	91.2313	93.0683	90.5237
HeartEW	66.5741	65.6173	66.7901	66.6975	67.2222	**68.7963**	67.5000	68.1790	67.4691	64.6296	67.4074	66.7284	67.5309	67.6543
Ionosphere	89.7887	**91.8545**	89.9531	90.0000	90.1174	90.5399	91.0094	89.5305	90.3286	91.7840	89.3662	89.6244	89.6244	89.2958
Lymphography	79.4315	75.5463	76.5130	78.9259	78.7111	78.0167	78.8833	78.9815	79.6759	75.5296	**80.2019**	78.5537	79.8574	77.4852
Sonar	74.5635	**83.1349**	77.7381	76.4683	74.2063	75.6746	74.9206	74.7222	76.0714	82.4206	77.1032	74.8016	75.4365	77.3413
SpectEW	79.6914	80.1235	80.3086	79.5679	80.2160	80.0926	79.8765	80.1852	80.0309	**80.9877**	79.7840	79.7531	80.4630	80.2778
Tic-tac-toe	73.7066	74.9132	73.8108	74.1059	**74.9392**	71.1372	73.9497	74.1927	73.8889	73.6285	73.9410	74.2101	73.7413	74.1667
Wine	91.4455	88.1662	89.6914	89.7917	91.2680	**91.8801**	89.0998	88.2433	90.0154	84.1127	89.7659	88.7809	88.7834	89.8251
WaveformEW	86.3716	**87.5430**	86.6256	86.6604	86.8031	86.6891	86.8199	87.1145	87.4582	87.5178	86.9083	85.7472	86.8850	86.8368
Zoo	96.2132	**97.0068**	96.1187	95.8428	96.0128	95.3250	94.1950	95.5480	95.3893	96.2169	95.7710	96.2736	96.1073	96.0242

Note: Bold values indicate the best results; higher accuracy is better.

Table 6. Mean number of selected features obtained by the evaluated algorithms across the feature-selection datasets.

Dataset	MaCRO-DE	BRO	DBO	DE	DMOA	GWO	HHO	MFO	MGO	PSO	SHADE	WOA	JADE	FLA
BreastCancer	2.2500	3.0500	2.4167	**1.9667**	2.2667	2.2833	2.1667	2.4833	2.5500	3.5667	2.2333	2.4833	2.3167	2.7333
BreastEW	2.8333	7.6333	2.9833	2.7167	3.1167	3.0167	2.7333	**2.6167**	2.7167	12.5833	2.9000	2.9167	3.0500	3.3000
Glass	1.5333	2.3000	1.7333	1.6667	1.4500	1.7000	1.4833	1.4333	1.5000	3.2333	1.7333	**1.3667**	1.8000	1.6333
HeartEW	2.1667	4.1833	2.3000	2.0833	2.1333	2.4333	**1.8833**	2.0667	2.0667	5.3167	2.1667	2.4167	2.4500	2.2667
Ionosphere	3.3000	11.7500	3.8500	3.1000	3.4667	3.5333	3.6167	**3.0667**	3.1667	14.4000	3.1833	3.7833	3.5333	4.0667
Lymphography	4.0333	8.0667	5.5333	**4.0000**	4.9667	4.7500	5.2833	5.3667	6.6167	9.2000	4.8000	5.9667	4.6167	6.5333
Sonar	6.1167	27.2667	7.2333	**5.1333**	5.5500	5.9167	6.6333	6.1667	9.3000	29.7000	7.6667	6.6500	7.2667	10.1000
SpectEW	4.0000	9.5500	4.1667	**3.9833**	**3.9833**	4.2000	4.2000	4.1167	4.4667	11.2333	4.4000	4.4000	4.3333	4.7667
Tic-tac-toe	4.7333	5.7833	5.1333	4.8333	5.2333	**4.3833**	5.3167	5.6000	5.9333	5.7667	5.2667	5.6333	5.0667	5.9333
Wine	3.5333	5.3333	3.9333	3.0167	3.4833	3.3833	**2.9833**	3.3667	3.9333	6.2167	3.7167	3.6333	3.2667	4.0500
WaveformEW	7.8667	20.1500	9.8000	**7.7500**	7.8333	8.3167	10.0833	8.7333	13.2167	23.0000	9.1167	8.8833	8.9167	11.5333
Zoo	**3.9333**	6.9000	4.4333	4.1667	4.2667	4.3500	4.4500	4.4167	4.5667	7.7500	4.1333	5.0500	4.1833	4.9500

Note: Bold values indicate the fewest selected features; lower values are better.

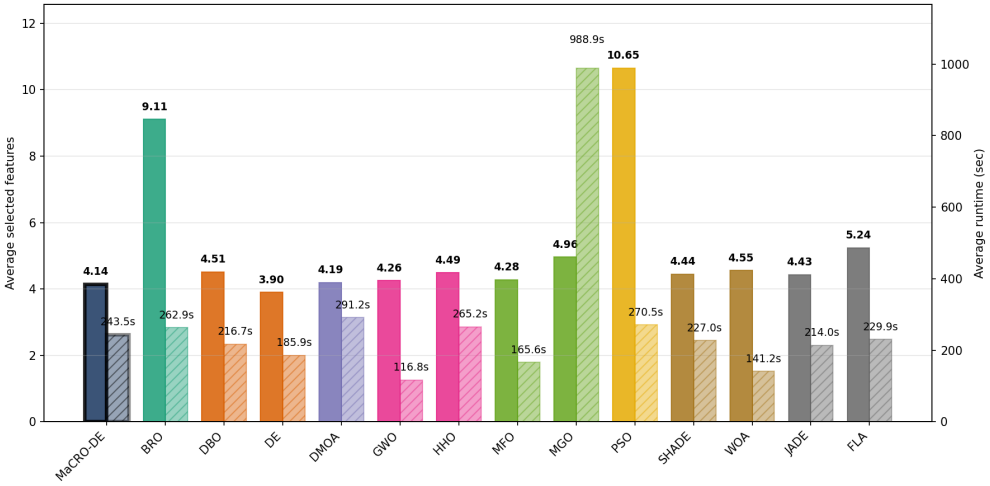

Figure 7. Global trade-off between the average number of selected features and computational runtime for the evaluated metaheuristic algorithms.

Overall, the experimental results indicate that MaCRO-DE provides a competitive wrapper-based feature selection strategy for datasets with different dimensionalities, numbers of instances, and class structures. The results also demonstrate that improvements in classification performance and feature reduction depend on the dataset. Consequently, no single optimization method shows consistent superiority across all problems. Nevertheless, the consistent performance of the algorithm across all evaluated metrics supports the applicability of the proposed Mahalanobis distance and Cholesky decomposition mechanisms for guiding the search toward informative, reduced feature subsets. Additional detailed graphical analyses and sensitivity results are provided in the supplementary materials (Supplementary Note S12).

6. Conclusions

In this study, a binary version of MaCRO-DE was used to solve the feature selection problem. The algorithm incorporates Cholesky decomposition to enhance the calculation of the Mahalanobis distance, which is employed to cluster individuals based on their proximity to the optimal solution and to select appropriate solutions for the mutation operator. This mechanism enables the algorithm to consider both the closest and the farthest individuals, thereby facilitating a balance between exploration and exploitation. The proposed algorithm was evaluated using the CEC 2017 benchmark functions, considering the optimal mean performance, the median, and the standard deviation. The findings indicated that the approach exhibited competitive performance and stable convergence in comparison to both classical and recent metaheuristic algorithms. Furthermore, the binary version of the algorithm was evaluated on twelve UCI datasets using KNN and SVM classifiers. The comparison encompassed a range of metrics, including Accuracy, Precision, Recall, the F1-Score, the number of selected features, and computational runtime. In summary, MaCRO-DE demonstrated consistent and competitive performance across all datasets under evaluation. Although this approach did not consistently achieve top performance, it exhibited a favorable balance between classification performance and feature reduction, while selecting relatively compact subsets. Future work will concentrate on high-dimensional and imbalanced datasets, additional classifiers, and multi-objective extensions of the proposed method.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/make8090291/s1, Supplementary Note S1: An Examination

on Different Distance Measures; Supplementary Note S2: Analysis of Various Transfer Functions; Supplementary Note S3: Overview of the Classification Models; Supplementary Note S4: Detailed Validation Protocol for Wrapper-Based Feature Selection; Supplementary Note S5: Description of the CEC-2017 Benchmark Suite; Supplementary Note S6: Extended Comparative Results of the CEC-2017 Benchmark Using Different Distance Measures; Supplementary Note S7: Ablation Study of MaCRO-DE Algorithm; Supplementary Note S8: Parameter Sensitivity Analysis; Supplementary Note S9: Extended Comparative Results of the Full CEC-2017 Benchmark; Supplementary Note S10: Evaluation of Transfer Functions for Binarization; Supplementary Note S11: Parameter Sensitivity Analysis of α_1 and β_2; Supplementary Note S12: Extended Feature Selection Results.

Author Contributions: Investigation, Methodology, Software, Writting—original draft, A.C.-O.; Supervision, Validation, Writing—original draft, Investigation, Conceptualization, D.O.; Supervision, Project administration, Funding acquisition, Writing—original draft, M.P.-C.; Supervision, Project administration, Funding acquisition, Writing—original draft, G.S.-G.; Data Curation, Validation, Formal analysis, Writing—original draft, I.A.; Data Curation, Validation, Formal analysis, Writing—original draft, A.V.-G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets analyzed in this study are publicly available in the UCI Machine Learning Repository: https://archive.ics.uci.edu (accessed on 1 September 2026). The source code used to implement the proposed methodology and conduct the experiments is publicly available in the following GitHub repository: https://github.com/angelcasasordaz/Adaptive_Mahalanobis-Cholesky_Differential_Evolution_MaCRO_DE.git (accessed on 1 September 2026). The MaCRO-DE implementation used in the reported experiments is provided in the file `macro_de_optimizer.py`.

Conflicts of Interest: The authors have no competing interests to declare relevant to this article's content. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject or materials discussed in this manuscript.

References

1. Zaki, M.J.; Meira, W., Jr.; Meira, W. *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*; Cambridge University Press: Cambridge, UK, 2020.
2. Blum, A.L.; Langley, P. Selection of relevant features and examples in machine learning. *Artif. Intell.* **1997**, *97*, 245–271. [CrossRef]
3. Liu, H.; Yu, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 491–502. [CrossRef]
4. Guyon, I.; Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res.* **2003**, *3*, 1157–1182. [CrossRef]
5. Rostami, M.; Berahmand, K.; Nasiri, E.; Forouzandeh, S. Review of swarm intelligence-based feature selection methods. *Eng. Appl. Artif. Intell.* **2021**, *100*, 104210. [CrossRef]
6. Song, X.; Zhang, Y.; Zhang, W.; He, C.; Hu, Y.; Wang, J.; Gong, D. Evolutionary computation for feature selection in classification: A comprehensive survey of solutions, applications and challenges. *Swarm Evol. Comput.* **2024**, *90*, 101661. [CrossRef]
7. Vivek, Y.; Vadlamani, S.K.; Ravi, V.; Krishna, P.R. Improved differential evolution based feature selection through chaos, quantum, and lasso logistic regression. *Quantum Mach. Intell.* **2026**, *8*, 13. [CrossRef]
8. Gupta, S.; Sahu, D. Diversified clustering and archive pattern-based differential evolution for feature selection problems. *Clust. Comput.* **2026**, *29*, 21. [CrossRef]
9. Hu, P.; Zhu, J. A filter-wrapper model for high-dimensional feature selection based on evolutionary computation. *Appl. Intell.* **2025**, *55*, 581. [CrossRef]
10. Tran, B.; Xue, B.; Zhang, M. Adaptive multi-subswarm optimisation for feature selection on high-dimensional classification. In *Proceedings of the Genetic and Evolutionary Computation Conference*; ACM: New York, NY, USA, 2019; pp. 481–489.
11. Taradeh, M.; Mafarja, M.; Heidari, A.A.; Faris, H.; Aljarah, I.; Mirjalili, S.; Fujita, H. An evolutionary gravitational search-based feature selection. *Inf. Sci.* **2019**, *497*, 219–239. [CrossRef]

12. Too, J.; Rahim Abdullah, A. Binary atom search optimization approaches for feature selection. *Connect. Sci.* **2020**, *32*, 406–430. [CrossRef]
13. Emary, E.; Zawbaa, H.M.; Hassanien, A.E. Binary grey wolf optimization approaches for feature selection. *Neurocomputing* **2016**, *172*, 371–381. [CrossRef]
14. Hu, P.; Pan, J.S.; Chu, S.C. Improved binary grey wolf optimizer and its application for feature selection. *Knowl.-Based Syst.* **2020**, *195*, 105746. [CrossRef]
15. Al-Tashi, Q.; Kadir, S.J.A.; Rais, H.M.; Mirjalili, S.; Alhussian, H. Binary optimization using hybrid grey wolf optimization for feature selection. *IEEE Access* **2019**, *7*, 39496–39508. [CrossRef]
16. Storn, R.; Price, K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
17. Kelbert, M. Survey of distances between the most popular distributions. *Analytics* **2023**, *2*, 225–245. [CrossRef]
18. Mohammed, J.Z.; Wagner, M. *Data Mining and Analysis: Fundamental Concepts and Algorithms*; Cambridge University: Cambridge, UK, 2014.
19. Mahalanobis, P.C. On the generalized distance in statistics. *Sankhyā Indian J. Stat. Ser. A (2008-)* **2018**, *80*, S1–S7.
20. Krishnamoorthy, A.; Menon, D. Matrix inversion using Cholesky decomposition. In *2013 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), Poznan, Poland, 26–28 September 2013*; IEEE: Piscataway, NJ, USA, 2013; pp. 70–72.
21. Casas-Ordaz, A.; Navarro-Velázquez, M.A.; Valdivia, A.; Oliva, D.; Ramos-Frutos, J. Improved Differential Evolution with Mahalanobis Distance: IMPDE. In *2025 IEEE Congress on Evolutionary Computation (CEC), Hangzhou, China, 8–12 June 2025*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–8.
22. Mühlenbein, H.; Paass, G. From recombination of genes to the estimation of distributions I. Binary parameters. In *International Conference on Parallel Problem Solving from Nature*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 178–187.
23. Hansen, N. The CMA evolution strategy: A comparing review. In *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 75–102.
24. Rahkar Farshi, T. Battle royale optimization algorithm. *Neural Comput. Appl.* **2021**, *33*, 1139–1157. [CrossRef]
25. Xue, J.; Shen, B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *J. Supercomput.* **2023**, *79*, 7305–7336. [CrossRef]
26. Agushaka, J.O.; Ezugwu, A.E.; Abualigah, L. Dwarf mongoose optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **2022**, *391*, 114570. [CrossRef]
27. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
28. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [CrossRef]
29. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [CrossRef]
30. Zheng, B.; Chen, Y.; Wang, C.; Heidari, A.A.; Liu, L.; Chen, H. The moss growth optimization (MGO): Concepts and performance. *J. Comput. Des. Eng.* **2024**, *11*, 184–221. [CrossRef]
31. Kennedy, J.; Eberhart, R. Particle swarm optimization. In *Proceedings of ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995*; IEEE: Piscataway, NJ, USA, 1995; Volume 4, pp. 1942–1948.
32. Tanabe, R.; Fukunaga, A. Success-history based parameter adaptation for differential evolution. In *2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013*; IEEE: Piscataway, NJ, USA, 2013; pp. 71–78.
33. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
34. Zhang, J.; Sanderson, A.C. JADE: Adaptive differential evolution with optional external archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [CrossRef]
35. Hashim, F.A.; Mostafa, R.R.; Hussien, A.G.; Mirjalili, S.; Sallam, K.M. Fick's Law Algorithm: A physical law-based algorithm for numerical optimization. *Knowl.-Based Syst.* **2023**, *260*, 110146. [CrossRef]
36. Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *J. Am. Stat. Assoc.* **1937**, *32*, 675–701. [CrossRef]
37. Friedman, M. A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Stat.* **1940**, *11*, 86–92. [CrossRef]
38. Kelly, M.; Longjohn, R.; Nottingham, K. The UCI Machine Learning Repository. 2023. Available online: https://archive.ics.uci.edu (accessed on 12 June 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.