People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
Mohamed El Bachir El Ibrahimi University of Bordj Bou Arreridj
Faculty of Mathematics
Department of Mathematics

THESIS
In order to obtain the Doctorate degree in LMD (3^{rd} cycle)
Branch : Applied Mathematics
Option : Operational Research

THEME

Using Multi-objective Meta-heuristics for Data Mining

By: Soumaia KAHLOUL

Publicly defended on: 05/01/2025

In front of the jury composed of:

Dr. Fodil Belhadj	**University of B.B.A**	**President**
Pr. Djaafar Zouache	**University of B.B.A**	**Supervisor**
Dr. Boualem Brahmi	**University of B.B.A**	**Co-Supervisor**
Dr. Djamila Mohdeb	**University of B.B.A**	**Examiner**
Pr. Hichem Debbi	**University of M'sila**	**Examiner**
Dr. Abdelaziz Lakhfif	**University of Setif**	**Examiner**

2024/2025

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed El Bachir El Ibrahimi de Bordj Bou Arréridj
Faculté des Mathématiques
Département de Mathématiques

THÈSE
Pour l'obtention du diplôme de Doctorat en LMD (3ème cycle)
Branche : Mathématiques
Option : Recherche Opérationnelle

THÈME

Utilisation des Méta-heuristiques Multi-objectifs pour la Fouille de Données

---

Présentée par : Soumaia KAHLOUL

Soutenue publiquement le : 05/01/2025

Devant le jury composé de:

Dr. Fodil Belhadj	Université de B.B.A	Président
Pr. Djaafar Zouache	Université de B.B.A	Directeur de thèse
Dr. Boualem Brahmi	Université de B.B.A	Co-directeur de thèse
Dr. Djamila Mohdeb	Université de B.B.A	Examinateur
Pr. Hichem Debbi	Université de M'sila	Examinateur
Dr. Abdelaziz Lakhfif	Université de Sétif	Examinateur

2024/2025

Acknowledgments

First and foremost, I am grateful to Allah for granting me the strength to undertake this work and to advance further.

With deep gratitude, I would like to thank and express my appreciation to my supervisor, Pr. Djaafar Zouache, for his profound understanding and wise guidance in research. His unwavering support, insightful advice, patience, and encouragement have been invaluable throughout this journey. I would also like to thank Dr. Boualam Brahmi, whose invaluable mentorship and expertise were instrumental in the completion of this thesis.

I dedicate this work to the angel of my life, my mother, Mebrouka Fites, for her assistance and presence in my life, her encouragement, and her prayers throughout my studies. I also dedicate it to my father, Ferhet Kahloul; may God protect them both.

This work is also dedicated to my little family, especially my dear husband, Kamel Zergane, whose belief in my abilities has been the driving force behind my success. To my children, Zakaria, Iskandar, and Omnia, for their patience, encouragement, and help, whose unwavering love and support have made everything possible.

Finally, I dedicate this work to my brothers, sisters, and all my friends who have shared both good and challenging times with me. To all those who love Soumaia and those whom Soumaia loves.

Contents

List of Figures

List of Tables

List of Algorithms

General Introduction

In the age of information and big data, the ability to extract valuable knowledge from vast data volumes has become crucial for innovation and decision-making. This capability requires advanced algorithms and computational methods to identify significant patterns in complex datasets [96].

Knowledge discovery begins with data preprocessing, including cleaning, integration, transformation, and dimensionality reduction through feature selection. Feature selection is essential for building efficient models by selecting relevant features from a larger set, improving model performance while reducing dimensionality and minimizing overfitting [128].

Data mining uses various methodologies, including statistics and machine learning algorithms, to extract valuable knowledge from large datasets. This process can be viewed as an optimization problem, encompassing both discrete optimization (e.g., selecting features) and continuous optimization (e.g., optimizing model parameters). Classification tasks rely on machine learning algorithms, including supervised approaches like Decision Trees, Random Forests, Support Vector Machines (SVM), and K-Nearest Neighbors (K-NN), as well as unsupervised techniques like K-means Clustering [56, 167].

Feature selection enhances classification models' performance but presents a complex challenge, requiring balance between multiple objectives such as maximizing accuracy while minimizing selected features. This multifaceted nature aligns with multi-objective optimization, which provides a framework for finding Pareto optimal solutions. Metaheuristic optimization techniques, particularly nature-inspired algorithms, play a crucial role in efficiently navigating vast solution spaces to address these challenges [41, 32].

In real-world applications, large data volumes complicate data mining tasks. Metaheuristic optimization techniques are crucial for efficiently navigating solution spaces to address these challenges. These algorithms, inspired by natural phenomena, provide effective strategies for exploring large search spaces to find high-quality solutions [12].

The combinatorial nature of feature selection, where features are either selected or

not, adds complexity to the optimization process. Many such problems are NP-complete, requiring specialized approaches that can handle discrete search spaces while optimizing multiple objectives. Multi-objective optimization techniques like Pareto optimization, weighted-sum approach, and ϵ-constraint method are essential for balancing trade-offs between conflicting objectives.

Feature Selection has become a crucial technique at the convergence of machine learning and metaheuristics, providing an effective strategy for selecting optimal subsets of features from high-dimensional datasets. By leveraging the strengths of machine learning alongside the adaptability and efficiency of metaheuristics, feature selection holds significant promise for enhancing data mining and model development. It acts as a link between machine learning and metaheuristic algorithms. The challenge of feature selection is framed as a combinatorial optimization problem, characterized by its discrete nature where features are either included or excluded. This complexity requires tailored approaches capable of navigating discrete search spaces while optimizing multiple objectives simultaneously.

Feature selection is crucial for machine learning model effectiveness, as feature quality directly affects performance. Given the complexity of high-dimensional datasets, metaheuristic optimization becomes essential for efficient solution space exploration. The core challenge involves finding optimal solutions while balancing conflicting objectives: maximizing accuracy while minimizing dimensionality. The Pareto optimization framework effectively addresses these trade-offs in multi-objective optimization, enabling the identification of optimal feature subsets that enhance machine learning outcomes [49].

Despite advances in feature selection and optimization, significant challenges remain in multi-objective feature selection. Current single-objective approaches fail to adequately address the simultaneous optimization of accuracy and dimensionality reduction. In high-dimensional datasets, where the search space grows exponentially, traditional algorithms face difficulties in effective navigation, leading to overfitting and poor generalization. These limitations highlight the need for advanced techniques capable of handling multi-objective optimization in high-dimensional, combinatorial settings. Future research should focus on developing specialized metaheuristics that balance computational efficiency with solution quality in complex feature selection tasks [3].

This thesis investigates the role of metaheuristics in improving data mining tasks, with a specific emphasis on feature selection for supervised classification. The objective is to develop and assess a novel multi-objective feature selection framework that employs combinatorial optimization techniques and nature-inspired metaheuristic algorithms within a wrapper method approach. This research advances the fields of optimization, machine learning, and data mining by enhancing feature selection through population-based metaheuristic algorithms. These algorithms facilitate the simultaneous exploration of various regions in the feature space, thereby improving the discovery of global optima and ad-

dressing local optima challenges.

This research contributes to the fields of optimization, machine learning, and data mining through two phases:

1. Development of multi-objective versions of the Henry Gas Solubility Optimization (HGSO) algorithm to solve complex optimization problems:
    - First version employs a grid mechanism and roulette wheel for selection and deletion strategies;
    - Second version uses crowding distance with random selection to maintain population diversity.
2. Application of the superior MOHGSO algorithm to feature selection problems:
    - Investigation of eight transfer functions within the proposed approach;
    - Implementation in both single-objective and multi-objective optimization contexts;
    - Application to COVID-19 diagnostic models using X-ray image datasets.

Phase 1: develops a novel multi-objective approach combining optimization techniques with nature-inspired metaheuristics. Two versions of the Multi-Objective Henry Gas Solubility Optimization (MOHGSO) algorithm are designed to address multi-objective complexities, extending single-objective HGSO capabilities through Pareto optimality, multi-archives, and multi-leaders.

MOHGSO algorithms were evaluated using established benchmarks (ZDT-series, DTLZ-series, and CEC2009) with two and three objectives, plus four real-world engineering design problems. Performance assessment focused on convergence (reaching optimal solutions) and diversity (exploring effective solutions across objective space). Results demonstrated MOHGSO's excellence in both metrics for complex multi-objective optimization problems.

Phase 2: applies MOHGSO to discrete search spaces for supervised classification feature selection. Eight transfer functions were evaluated in both single-objective (balancing feature reduction and classification accuracy) and multi-objective contexts (minimizing features while maximizing accuracy) using K-Nearest Neighbor classification.

This approach integrates deep learning with feature selection via the Multi-Objective Henry Gas Solubility Optimization (MOHGSO) algorithm to improve COVID-19 detection from X-ray images. Deep learning aids in feature extraction, while MOHGSO optimizes feature selection, balancing classification accuracy and feature count. The method was tested using K-Nearest Neighbors (KNN) and Support Vector Machines (SVM), both effective in medical imaging. Performance metrics, including accuracy, sensitivity, specificity, and efficiency, showed that the approach enhances classification

performance. KNN performed well with selected features, but SVM outperformed it in accuracy and generalization on unseen data. This highlights the importance of classifier choice in the feature selection process, as different classifiers respond differently to features selected by MOHGSO. Overall, the study showcases the potential of combining deep learning, machine learning, and advanced feature selection for effective COVID-19 detection.

The thesis is structured into two main parts across six chapters:

- Part One: Establishes foundational concepts in feature selection and optimization.
- Part Two: Details contributions through continuous and discrete optimization approaches tailored for feature selection.

Chapters cover an overview of research areas, proposed algorithms, experimental evaluations against benchmark functions, and applications in supervised classification tasks using K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) classifiers.

Chapter 1 provides an overview of optimization problems, multi-objective optimization principles, and nature-inspired metaheuristic algorithms.

Chapter 2 examines machine learning and data mining challenges, emphasizing feature selection's role in improving classification accuracy.

Chapter 3 introduces our proposed MOHGSO algorithm, detailing two new multi-objective versions and presenting experimental evaluations using established benchmark functions.

Chapter 4 presents eight binary versions of the HGSO algorithm for discrete feature selection problems, analyzing their effectiveness using the K-Nearest Neighbors classifier.

Chapter 5 demonstrates the application of MOHGSO to feature selection for supervised classification tasks, detailing its adaptation for discrete optimization problems.

Chapter 6 explores the HGSO algorithm's application in COVID-19 classification tasks, addressing both single-objective and multi-objective optimization scenarios.

The thesis concludes by summarizing the main contributions and suggesting future research directions in the field of multi-objective feature selection and optimization.

The following academic papers have been produced as a result of this research:

- Kahloul. S, and al. "A multi-external archive-guided henry gas solubility optimization algorithm for solving multi-objective optimization problems." Engineering Applications of Artificial Intelligence 109 (2022): 104588. Impact Factor: 7.5
- Kahloul.S, and al. (01-02 December 2021). Multi-Objective Henry Gas Solubility Optimization Algorithm for solving MOPs. Fourth Conference on Informatics and Applied Mathematics IAM'21.

- Kahloul.S. (December 18-19, 2022). Three novel algorithms for complex MOPs: Performance comparison study. The second National Conference on Pure and Applied Mathematics.
- Kahloul.S, and al. " An improved binary Henry Gas Solubility Optimization Algorithm for feature selection in classification. "Multimedia Tools and Applications, Impact factor: 3.0
- Kahloul.S, and al. " S-shaped versus V-shaped transfer functions for binary Multiobjective Henry Gas Solubility Optimizer in Feature Selection Problem".

CHAPTER 1

Multi-objective Optimization

1.1 Introduction

This chapter provides a comprehensive overview of Multi-Objective Programming and Optimization, exploring the fundamental principles and techniques involved. We will present various solution methods and established metaheuristic approaches for addressing Multi-objective Optimization Problems. Finally, we will justify our focus on Multi-objective Metaheuristics, which are effective in finding high-quality solutions for complex optimization problems central to this thesis.

1.2 Multi-Objective Optimization Problems

In mathematics, optimization problems involve finding solutions that yield the best outcomes, such as maximizing profit or minimizing travel time. These problems are prevalent in engineering, computer science, and economics.

Understanding the structure and characteristics of an optimization problem is essential for effective formulation and solution. Key elements include the objective function, decision variables, and constraints, which enable systematic application of solution algorithms to discover optimal or near-optimal solutions. Each optimization problem is characterized by (see Figure 1.1) [1]::

- The objective function to be optimized (minimized or maximized), representing the goal of the optimization process;
- Decision variables that affect the objective function's value and can be controlled to achieve an optimal solution;

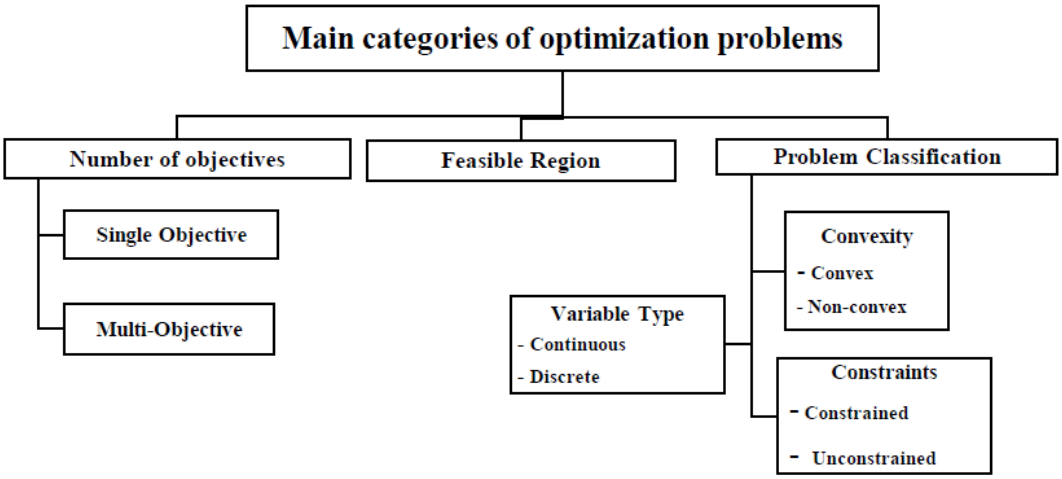


Figure 1.1: Navigating the Landscape of Optimization Problems.

- Constraints defining the feasible region of valid solutions, formulated as inequality or equality equations that bound the search space;
- The feasible region, comprising all possible solutions for decision variables satisfying the constraints;
- The optimal solution, which is the feasible solution resulting in the minimum or maximum objective function while satisfying all constraints.

Problems can be classified as linear/nonlinear, discrete/continuous, constrained/unconstrained, etc. The mathematical formulation represents objectives, decisions, and constraints for real-world optimization problems as follows:

$$\begin{cases} opt \quad f(x) \\ s.c \\ \qquad g_i(x) \leq 0, \quad i = 1, 2, ..., m \\ \qquad h_i(x) = 0, \quad i = 1, 2, ..., p \end{cases} \tag{1.2.1}$$

where,$f(x)$ is the objective function to be optimized, $x \in D$ is the vector of decision variables, and D is the search space. g_i and h_i represent the inequality and equality constraints, respectively. The specifics of f, g, and h determine the problem's nature and solution method.

Solving an optimization problem involves searching for one or more optimal solutions according to the objective function. Methods include graphical methods (for two decision variables), the simplex algorithm (for Linear Programming), and its variants like revised simplex method and interior point methods. The efficiency of these methods can be influenced by the number of variables and constraints.

Remarque 1. *In Linear Programming (LP), the feasible region is convex with optimal solutions at extreme points. In Integer Linear Programming (ILP), feasible regions may not be convex due to integer constraints.*

Integer Linear Programming (ILP) problems can be challenging; however, resolution methods can find optimal solutions efficiently within defined time intervals. Techniques like Cut methods (e.g., Dantzig Cut) iteratively add linear constraints to tighten feasible regions. Branch and Bound methods systematically divide problems into smaller subproblems by fixing variables to 0 or 1.

Multi-objective programming, or multi-objective optimization, involves simultaneously optimizing multiple conflicting criteria. Unlike single-objective optimization focused on one optimal solution, multi-objective programming seeks a set of compromise solutions reflecting real-world scenarios where balancing multiple objectives is crucial.

Definition 1 (**Multi-objective programming**). *Let x be a set of solutions (actions), and let $f_m, m = 1, \cdots, M$, be a coherent family of real functions on x called objective functions or criteria. A multi-objective mathematical programming problem refers to a decision problem aimed at simultaneously optimizing the M objectives (where $M > 1$ is an integer)[77].*

Formally, a MOP with M objectives can be mathematically defined as follow:

$$\begin{cases} min \quad F(x) = (f_1(x), f_2(x), ..., f_M)^T \\ s.c \\ \qquad g_i(x) \leq 0, \quad i = 1, 2, ..., m \\ \qquad h_i(x) = 0, \quad i = 1, 2, ..., p \end{cases} \tag{1.2.2}$$

were, $x \in D$ denotes the decision variable vector, D represents the search space, and m and p indicate the number of inequality and equality constraints, respectively.

Understanding decision variable characteristics (integer, continuous, binary) aids in formulating multi-objective problems effectively.

Types of Decision Variables:

- Integer Values: Refers to ILP useful in scheduling or budgeting.
- Mixed Variables: Involves real and integer values suitable for production planning.
- Binary Values: Special case of ILP for binary choices common in machine learning.
- Continuous Values: Refers to LP problems used in resource allocation.

This understanding ensures appropriate formulations for multi-objective problems leading to meaningful results.

1.3 Multi-Objective Combinatorial Optimization

Combinatorial optimization addresses complex problems involving discrete variables with specific values:

$$x = \begin{cases} 1, \text{if the event's realization} \\ 0, \text{otherwise} \end{cases} \tag{1.3.1}$$

The goal is to optimize an objective function $f(x)$ assigning values to each solution while adhering to constraints that define feasible solutions. These problems often model as binary integer linear programming with binary decision variables [130].

Key Points:

- Combinatorial optimization involves finding optimal solutions from discrete sets.
- Objective functions represent metrics like cost or profit.
- Constraints define feasible solution spaces.
- Many classical problems belong to combinatorial optimization.

To solve combinatorial optimization problems effectively:
1. Define feasible solution space.
2. Establish objective function for evaluating quality.
3. Choose appropriate optimization method.

Modeling impacts solution quality; linear programming can model constraints effectively. Multi-objective modeling provides robust representation with trade-offs when faced with conflicting objectives.

A multi-objective combinatorial optimization problem seeks binary decision variables satisfying constraints while optimizing conflicting objectives.

1.4 Solution of Multi-objective Optimization Problems

In single-objective optimization problems, there typically exists one global optimum representing the best value for that objective. In contrast, multi-objective optimization often lacks a single "best" solution due to conflicting objectives. Scalar approaches combine multiple objectives into a single scalar function but have limitations in handling non-convex Pareto fronts.

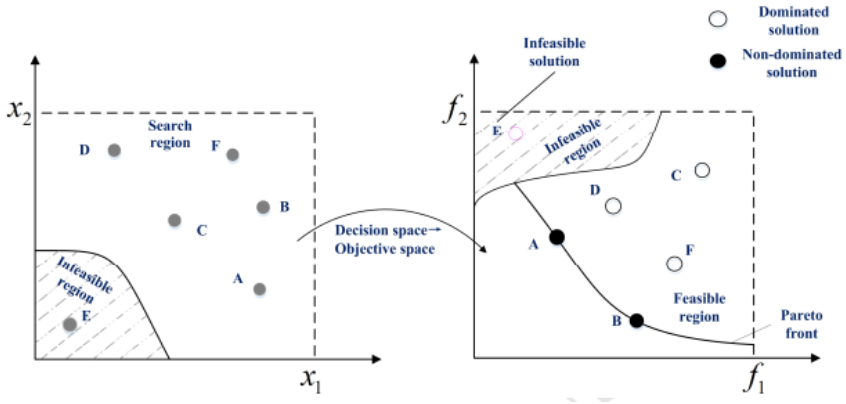


Figure 1.2: Decision space and objective space of a MOP.

Pareto-optimal solutions represent compromises between objectives forming a "Pareto front." The concept originates from economist Vilfredo Pareto's [106] work on optimality in multi-objective contexts where no single solution can minimize one objective without increasing another.

Definition 2 (**Pareto Dominance**). *A solution x_1 is considered to dominate another solution x_2 (denoted as $x_1 \prec x_2$) if and only if:*

$$\forall i \in \{1, 2, \ldots, M\},\ f_i(x_1) \leq f_i(x_2)\ \textit{and}\ \exists j \in \{1, 2, \ldots, M\}\ \textit{such that}\ f_j(x_1) < f_j(x_2).$$

Note that dominance is not a symmetric relation.

Definition 3 (**Pareto optimality**). *A solution $x \in D$ is deemed Pareto optimal if there is no other solution $x' \in D$ such that x' dominates x:*

$$\nexists\, x' \in D\ \textit{such that}\ x' \prec x.$$

Definition 4 (**Pareto optimal set**). *The set comprising all Pareto optimal solutions is known as the Pareto set, denoted by P^*, and is defined as:*

$$P^* = \{x \in D \mid \nexists\, x' \in D\ \textit{such that}\ x' \prec x\}.$$

For any given Pareto optimal solution, the set of corresponding objective function values in the objective space is termed the Pareto front (PF):

$$PF = \{F(x) \mid x \in P^*\}.$$

Similarly to single-objective optimization problems, multi-objective problems can be formulated with or without constraints. In the absence of constraints, the search for optimal solutions explores the entire decision space, as there are no limitations on the feasible solutions. However, if constraints are present, the search will be restricted to the feasible region defined by those constraints, limiting the exploration of the decision space (see Figure 1.2).

par Multi-objective problem-solving methods can be classified into three families based on when the decision-maker intervenes (see Figure 1.3)[46]:

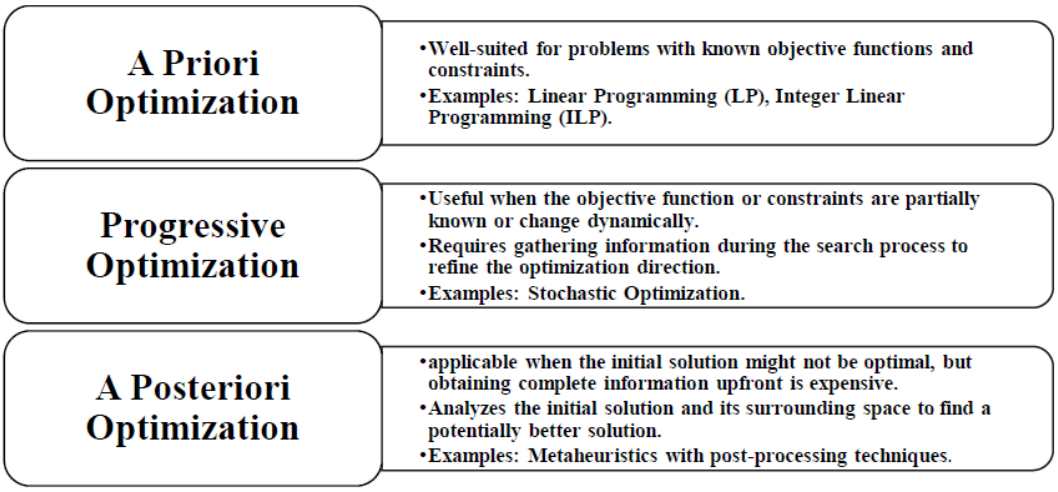


Figure 1.3: Selecting Optimization Methods Based on Information Availability.

A priori optimization methods, require the decision-maker to express their preferences regarding objectives before the optimization process begins. The optimization algorithm then seeks solutions that maximize alignment with these preferences. Techniques in this category include: i) scalarization, which combines multiple objectives into a single objective function using weights reflecting the relative importance of each objective; ii) utility functions, which numerically represent the decision-maker's preferences to guide the search towards aligned solutions.

Progressive optimization methods, refer to dynamic or interactive approaches where the decision-maker interacts with the optimization process during execution. The algorithm generates solutions iteratively, allowing for feedback from the decision-maker throughout the optimization process.

A posteriori optimization method, generate a set of Pareto-optimal solutions first, from which the decision-maker selects their preferred solution(s) based on their preferences (for more details see [99]).

The best approach for solving a specific multi-objective optimization problem depends on factors such as the number of objectives, problem complexity, computational resources, and decision-maker preferences. Each approach has its advantages and limitations.

Firstly, consider the decision-maker's availability; if they cannot be involved throughout the process, methods like a priori (setting preferences beforehand) or a posteriori (choosing from a final set of solutions) may be more practical. Secondly, the clarity of the decision-maker's preferences plays a role. If they have well-defined priorities, a priori methods can be efficient. However, for less clear preferences, progressive methods that involve interaction can help refine priorities during the optimization process.

Finally, the complexity of the problem itself is important. For very complex problems,

a posteriori methods might be more manageable. By first generating a set of good solutions (Pareto-optimal solutions), they allow for focused decision-making, enabling the selection of the best option based on the specific trade-offs involved.

1.4.1 Complexity Theory

Complexity theory is a branch of computer science that classifies and analyzes the difficulty of computational problems, including optimization problems. It provides methodologies for assessing the resource requirements of algorithms designed to address these problems, focusing on two key factors: i) time complexity, which measures the computational time required by an algorithm to solve a problem instance of a given size, and ii) space complexity, which indicates how much memory an algorithm requires for the same instance.

The complexity of an algorithm refers to the resources it needs to run, measured in terms of time (how long it takes) and space (how much memory it uses). We typically use "Big O" notation to describe complexity, illustrating how resource requirements grow as the input size increases. For example, sorting larger lists of numbers will vary in time and memory usage depending on the algorithm employed. Algorithms are classified as constant, linear, quadratic, etc., with polynomial-time algorithms generally considered efficient. In contrast, exponential algorithms become impractical for large problems, while super-polynomial algorithms grow faster than any polynomial function [112].

Problem complexity extends beyond specific algorithms to examine the inherent difficulty of problems themselves. For instance, solving a maze may be inherently complex regardless of strategy. Problem complexity assesses the fundamental time and space requirements needed to solve the most challenging instances of a problem, even by theoretical computers like Turing machines. Problems solvable in polynomial time are deemed solvable, whereas those requiring exponential or super-polynomial time become impractical for moderately sized inputs [112].

Complexity classes categorize problems based on their difficulty concerning computational resources (time and space):

- **P Class**: Problems solvable by a deterministic Turing machine in polynomial time, indicating efficient solutions.
- **NP Class**: Problems for which solutions can be verified in polynomial time by a deterministic Turing machine, though finding those solutions may be challenging.
- **NP-Complete**: A special subset of NP problems that are considered the hardest; any NP problem can be efficiently reduced to an NP-complete problem.
- **NP-Hard**: Problems broader than NP-Complete; solving any NP problem can be efficiently reduced to solving an NP-hard problem, but not all NP-hard problems are in NP.

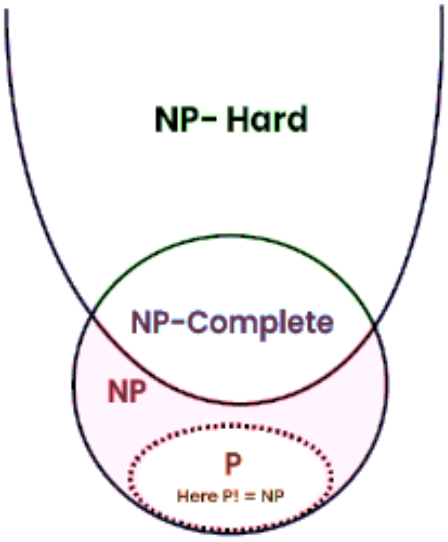


Figure 1.4: Problem complexity space.

To solve multi-objective optimization problems involving multiple conflicting objectives simultaneously, various methods are employed, categorized broadly into two main types [116]:

1. **Transformation Methods:** These traditional methods simplify multi-objective problems into single-objective ones that can be solved using existing optimization techniques. Key approaches include:

   - Aggregation methods that combine multiple objectives into a single objective function using weights reflecting their importance.
   - Target vector methods that aim to find solutions closest to specified target values for each objective.
   - Epsilon-constraint methods that optimize one objective while constraining others within specified bounds.

   These transformation methods enable the application of classical single-objective optimization techniques.

2. **Multi-objective Optimization Methods:** Specifically designed to handle multi-objective nature without simplifying or aggregating objectives. They aim to find a diverse set of Pareto-optimal solutions (Pareto front), representing trade-offs between conflicting objectives. Notable methods include:

   - Pareto optimization algorithms that search for non-dominated solutions in the objective space (e.g., NSGA-II, SPEA2).

- Evolutionary algorithms inspired by natural selection that iteratively improve candidate solutions towards Pareto-optimality.

The approaches for solving multi-objective optimization problems are grouped into exact and metaheuristic methods. Exact methods aim to find one or a subset of Pareto-optimal solutions representing optimal trade-offs among conflicting objectives through both scalar (aggregation-based) and non-scalar (target vector, epsilon-constraint) approaches.

1.4.2 Exact Methods

Exact methods are methods that aim to find one or a subset of Pareto-optimal solutions that represent the best trade-offs among the conflicting objectives, they often both scalar (aggregation-based) of multiple objectives into a single objective function, which can then be optimized using traditional optimization techniques, and non-scalar (target vector, epsilon-constraint, direct) approaches.

1. Scalar approaches based on aggregation methods: These methods transform multiple objectives into a single scalar objective function by combining the objectives using techniques such as weighted sum and weighted Tchebycheff, goal programming, or other techniques [129] and [134].

2. Target Vector Methods: These methods involve defining a target vector that represents the desired characteristics of the solution and searching for solutions that approximate this target vector, for example: The goal programming method, which minimizes the sum of weighted deviations from goal values specified for each objective. Goal attainment method, which minimizes the weighted sum of deviations from specified target values for each objective [166].

3. Epsilon-Constraint Method: In this method, one objective is optimized subject to constraints on the other objectives, which are expressed as inequalities with epsilon values [10].

4. Lexicographic Ordering Method: This method ranks the objectives by importance proposed by the decision-maker and optimizes them sequentially [47].

5. Min-Max Method: This method aims to optimize the worst-case performance across all objectives, by minimizing the maximum deviation from the ideal solution for each objective [81].

6. Direct Methods: These methods directly find the Pareto optimal set i.e. identify non-dominated solutions in the objective space without scalarizing or transforming the objectives [43].

7. Two-Phase Method: This method generates first a set of non-dominated solutions (Pareto optimal set) and then refines these solutions within those regions to improve their quality [136].

8. Klein and Hannan Method: This method generates the Pareto frontier by solving a series of single-objective optimization problems with modified objective functions, it is an extension of the epsilon-constraint method [72].

9. Sylva and Crema Method: This method involves transforming the multi-objective problem into a single-objective problem by using scalarization and then solving it using traditional optimization techniques. It is a variant of the epsilon-constraint method.[134, 135]

Multi-objective optimization problems are often difficult and belong to the class of NP-hard problems, which means finding the exact Pareto set can be computationally expensive, especially for large problems. As the problem size increases, exact methods that aim to find the global optimum for each objective become impractical or even impossible due to the exponential growth in computation time. These limitations of exact methods for complex problems encourage the use of metaheuristics. Metaheuristic algorithms aim to find good approximations of the Pareto set in a reasonable time. In this case, the goal is to identify a set of solutions that are good enough rather than finding the absolute optimal solution for each objective [116, 29].

1.5 Metaheuristics

1.5.1 Mode of operation

Metaheuristics are approximate optimization methods designed to overcome the limitations of basic heuristics (which is providing a solution corresponding only to a local optimum of the objective function) by using various strategies and exploring a broader search space and avoiding local optima, thus enabling the determination of global optima. They provide general frameworks with specific parameter adjustments that can be adapted to a wide class of optimization problems, often used when classical methods become inefficient (computationally expensive or impractical) or when the problem is complex and lacks exact solutions. While not guaranteeing the absolute global optimum, metaheuristics aim to find good or near-optimal solutions, especially for solving real-world optimization problems with multiple and conflicting objectives [129].

Metaheuristic algorithms aim to solve a given optimization problem by finding a mathematical object that minimizes or maximizes objective functions. The set of all possible solutions forms the search space, which is bounded with or without additional constraints. In multi-objective problems, the goal is to find a set of Pareto-optimal solutions, and the successive iterations allow for moving from less efficient to more efficient solutions, which

leads to improved solution quality. In general, metaheuristics incorporate the five essential concepts [46]:

1. Neighborhood defines the set of solutions reachable from a current solution through specific operations or transformations. The neighborhood concept is crucial in algorithms such as local search, where solutions are iteratively improved by moving from one solution to another within the neighborhood until the stopping criterion is met or an optimal solution is found.

2. Exploration aims to diversify the search space, avoiding premature convergence to local optima. Various strategies can be employed to promote exploration including randomly selecting solutions that allow the algorithm to explore different regions of the search space without bias, and using diverse operators (eg: mutation, crossover...) allowing for generating new candidate solutions from existing ones.

3. Exploitation focuses on intensifying promising areas of the search space identified during the exploration phase, leading to the improvement of solutions and complementing exploration.

4. Memory mechanisms allow the algorithm to "remember" good solutions or search regions encountered during the optimization process by storing the best solutions found so far. Maintaining an archive of the best solutions enables the algorithm to retain valuable information that can guide future iterations towards better solutions.

5. Learning mechanisms enable the algorithm to adapt its behavior, i.e., adjust dynamically its parameters, and strategies based on the information gathered during the optimization process, leading to better solutions in less time.

1.5.2 operating principles

Metaheuristics are optimization methods that can be applied to various optimization problems without requiring significant changes to the underlying algorithm. They are often iterative (iterative improvement) and stochastic, meaning they start with an initial solution and iteratively search for the better solution until a predetermined stopping criterion is satisfied, many metaheuristics involve randomization in their search process to avoid local optima and explore different areas of the search space to gradually improve their solutions over successive iterations [46, 29].

Metaheuristics, such as genetic algorithms, simulated annealing, and particle swarm optimization, explore the solution space to identify promising regions and converge towards an optimal solution by sampling the objective function using specific mechanisms of evaluating and updating solutions that vary depending on the chosen metaheuristic algorithm to find an optimal or near-optimal solution. They do not guarantee to find the

global optimum but aim to find a good solution (good enough) within a reasonable amount of time, especially when exact solutions are computationally expensive or impossible.

In multi-objective optimization, where there are multiple conflicting objectives to be optimized simultaneously, metaheuristics aim to identify a collection of solutions that represent the trade-offs between these objectives. The set of solutions is known as the Pareto-optimal set in the decision space, which represents the variables that can be controlled, and the corresponding set of objective values is called the Pareto front. Metaheuristics can approximate this Pareto front efficiently. The compromise surface is the image of the Pareto set in the objective space, representing the trade-offs between conflicting objectives, offering decision-makers insights into the range of possible solutions.

1.5.3 History of major metaheuristics

Metaheuristic algorithms work by iteratively searching for good (often near-optimal) solutions within a large search space to tackle optimization problems, especially those that are difficult to solve exactly. There are two main categories for approaching this search: single-solution-based and population-based (see Figure 1.5).

Single-based metaheuristics, also known as local search algorithms, focus on improving a single candidate solution over time. These methods generally work by iteratively modifying and enhancing a single candidate solution, using local search strategies to explore the solution space. The goal is to find an optimal or near-optimal solution by making incremental improvements.

Population-based metaheuristics work with a group of candidate solutions, known as a population. These methods involve interactions among the candidates to explore the solution space effectively. They are generally more robust and can find better solutions by leveraging the diversity within the population.

In this section, we will present the foundations of the main metaheuristics for solving multi-objective optimization problems, that are inspired by natural systems, drawing inspiration from various fields such as physics, evolutionary biology, and ethology. We first present the principles of representative methods including simulated annealing, genetic algorithms, and tabu research. Then, popular multi-objective metaheuristic algorithms that have been developed to tackle real-world problems involving trade-offs between multiple competing objectives will be presented.

Simulated Annealing is inspired by the behavior of slowly cooling metals, where gradual reduction in temperature allows atoms to reach a state of minimum energy, known as the ground state. In the simulated annealing algorithm, this translates to gradually reducing the "temperature" parameter to accept potentially worse solutions early on, allowing for the exploration of diverse options, and eventually converging towards an optimal or near-optimal solution. This concept is applied to an optimization algorithm by using a gradually decreasing "temperature" parameter to control the exploration vs. exploitation trade-off [135, 134]. Simulated Annealing has been adapted for multi-objective optimiza-

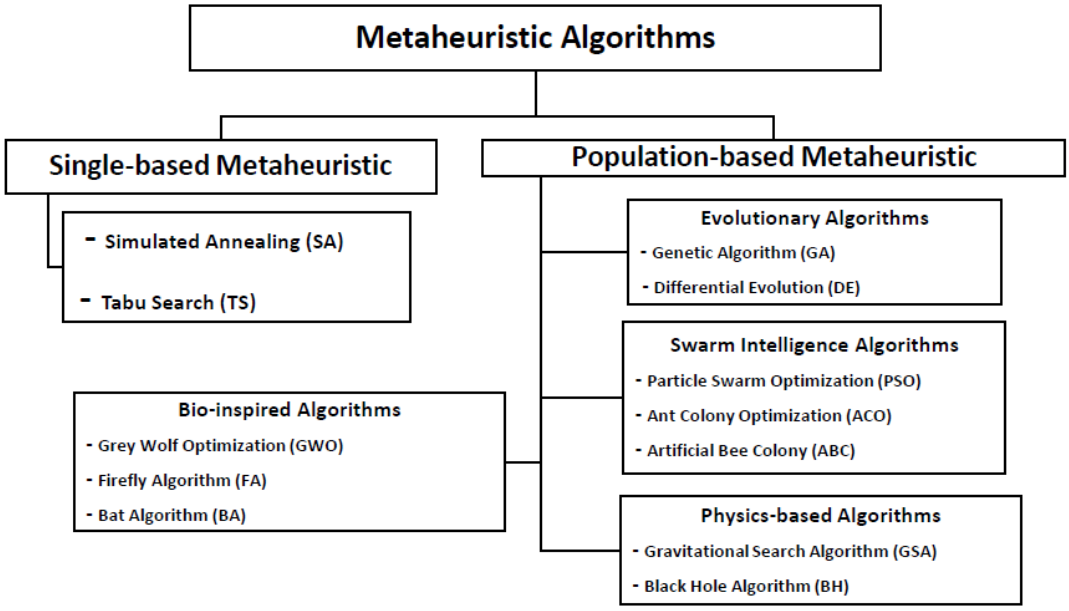


Figure 1.5: Taxonomy of metaheuristic optimization algorithms.

tion by incorporating temperature control and acceptance criteria to explore the Pareto front [71].

Tabu search is a metaheuristic optimization algorithm that does not directly mimic a specific natural phenomenon, it is inspired by the concept of overcoming taboos to explore a solution space more effectively, guiding the search process to avoid local optimality by employing strategies for diversification and intensification such as adaptive memory (learning from experience) to explore new areas, dynamically adjust its search strategy that terminates after a certain number of iterations or when satisfy certain criteria, leading to a more robust search process. Its ability to effectively navigate complex solution spaces and find near-optimal solutions has made it an effective choice for solving many combinatorial optimization problems. Tabu search (known for its efficiency in single-objective optimization) has been adapted for multi-objective optimization by incorporating tabu strategies and diversification mechanisms to explore the Pareto front [61].

Genetic algorithms draw inspiration from biological evolution and natural selection. These algorithms operate on a population of potential solutions, each evaluated by a fitness function that quantifies its quality relative to the optimization goal. The process mimics evolutionary principles, where fitter individuals have a higher chance of passing their traits to subsequent generations. The algorithm employs genetic operators such as mutation and crossover to create diversity and combine favorable features. Selection mechanisms then determine which solutions progress to the next generation, typically favoring those with higher fitness scores. Through repeated application of these processes, the population evolves over multiple generations, gradually converging towards optimal or near-optimal

solutions for the given problem [35].

To address multi-objective optimization challenges, genetic algorithms have been adapted using various strategies. These include incorporating Pareto dominance concepts, developing specialized fitness assignment methods, and implementing selection techniques that maintain diversity across the Pareto front. These modifications enable genetic algorithms to effectively navigate the complex trade-offs inherent in multi-objective problems, producing a set of solutions that represent different compromises among competing objectives, making them a powerful tool in solving complex optimization problems across various fields [125].

In recent years, the emergence of multi-objective metaheuristics has been significant, offering powerful techniques for tackling real-world problems across various domains including engineering, finance, logistics, and telecommunications where multiple conflicting objectives need to be considered simultaneously. These metaheuristics aim to find a set of compromise solutions, known as the Pareto front, without the need to convert the multi-objective problem into a single objective at the risk of losing its relevance [78, 164]. These metaheuristic algorithms, among others, have demonstrated great success in optimizing multi-objective problems including Differential evolution algorithms that have been extended to handle multiple objectives by integrating dominance-based selection and diversity maintenance mechanisms.

Following early efforts to solve multi-objective problems, numerous Multi-Objective Evolutionary Algorithms (MOEAs) emerged, including NSGA-II [33], SPEA2 [165], and MOEA/D [154]. NSGA-II employs a fast non-dominated sorting scheme to rank solutions and a crowding comparison operator to maintain diversity. MOEA/D decomposes multi-objective problems into single-objective subproblems, optimizing them simultaneously while considering the relationships between neighboring subproblems based on their weight vectors.

In fact, NSGA-II remains a widely used and powerful tool for solving complex multi-objective optimization problems in various fields like engineering design, resource allocation, and machine learning. NSGA-II, which stands for Non-dominated Sorting Genetic Algorithm II, is a popular algorithm used for solving multi-objective optimization problems. In these problems, there's no single "best" solution, but rather a set of trade-offs between multiple objectives. NSGA-II aims to find a well-distributed set of solutions that represent these trade-offs effectively [33]. Here's a breakdown of how NSGA-II works 1:

1. Population Initialization: It starts with a population of individuals (candidate solutions) encoded with chromosomes (representing the solution's properties). Each individual is evaluated based on its objective functions.

2. Non-dominated Sorting: Individuals are ranked based on their dominance relationship. A solution dominates another if it performs better on at least one objective without being worse on any others. This sorting process creates multiple fronts, with the first front containing the non-dominated solutions (best trade-offs).

Algorithm 1: NSGA-II Algorithm

Population size N, maximum generations T
Set of non-dominated solutions P_f
Initialize population P_0 with random individuals
for $t = 1$ **to** T **do**
    Evaluate fitness of all individuals in P_t
    F = FastNonDominatedSort(P_t)
    for $i = 1$ **to** $|F|$ **do**
        $I_{crowding}$ = CalculateCrowdingDistance(F_i)
    end
    Q_t = CombinePopulations(P_t, offspring population)
    P_{t+1} = Selection(Q_t, N)
end
$P_f = F_1$
Return P_f

3. Crowding Distance Calculation: Within each front, a crowding distance is assigned to each individual. This distance represents the density of solutions surrounding it in the objective space. A larger crowding distance indicates a more isolated solution on the Pareto front (set of optimal solutions).

4. Selection: A new population is created by selecting individuals based on both their rank (fronts) and crowding distance. Individuals from lower-ranked fronts are always preferred. If the fronts are exhausted, crowding distance is used to choose the least crowded solutions from the current front. This ensures diversity within the population and avoids premature convergence to a local optimum.

5. Crossover and Mutation: Genetic operators like crossover and mutation are applied to the selected individuals to generate offspring with new combinations of genetic material. This promotes the exploration of the search space.

6. Repeat: Steps 2-5 are repeated for a predefined number of generations, gradually evolving the population towards a diverse set of Pareto-optimal solutions.

NSGA-II stands out as a powerful algorithm for tackling multi-objective optimization problems. Thanks to its crowding distance calculation, It finds diverse, well-distributed solutions across the Pareto front. This ensures a comprehensive exploration of the trade-offs between conflicting objectives. Furthermore, NSGA-II incorporates elitism, cleverly preserving the best solutions from each generation to guide the search towards optimal outcomes. However, it's not without limitations. Sorting and calculating crowding distances can become computationally expensive for large populations. Additionally, fine-tuning parameters like population size and mutation rates is crucial for optimal performance.

Despite these considerations, NSGA-II remains a dominant force in multi-objective optimization due to its effectiveness and ability to handle complex challenges.

In fact, there are other natural phenomena that have inspired the development of diverse metaheuristic algorithms which are called nature-inspired metaheuristics by using principles from nature and mimicking natural processes of adaptation, exploration, and exploitation, to efficiently solve complex optimization problems. For example: Ant Colony Optimization (ACO) is inspired by the behavior of ants in searching for food and mimics its communication to find the shortest path that guides to better food sources. The Ant Colony Optimization algorithm simulates this behavior, where ants move through a problem space (explore the search space), leaving a trail that influences the search paths of other ants and leads them towards promising solutions. Particle swarm optimization (PSO) is inspired by the social behavior of birds, and simulates the movement of particles (solutions) in a search space, influenced by their previous positions and the best positions found by their neighbors, leading the swarm towards promising areas of the search space. These examples demonstrate how metaheuristic algorithms have become the forefront of the current research thanks to the use of a population of solutions to efficiently solve complex optimization problems.

The Multi-Objective Particle Swarm Optimization (MOPSO) stands out as a prominent PSO-based algorithm for multi-objective optimization. It utilizes an external repository to store optimal configurations and guide the search, along with an adaptive grid archiving strategy to promote swarm diversity [27]. Here's how MOPSO works 2:

1. Initialization: Similar to PSO (Particle Swarm Optimization), MOPSO starts with a swarm of particles. Each particle represents a potential solution in the search space and has a position vector and a velocity vector. The positions encode the candidate solutions, and the velocities determine how the particles move in the search space.

2. Fitness Evaluation: Each particle's fitness is evaluated based on all the objective functions involved. This evaluation determines how well a solution performs on each objective.

3. Pareto Dominance: MOPSO utilizes the concept of Pareto dominance to identify promising solutions. A solution dominates another if it performs better on at least one objective without being worse on any others.

4. Personal Best *pbest* and Global Best *gbest*: MOPSO maintains two important concepts:

    - Personal Best: Each particle keeps track of its own best position encountered so far based on dominance. This guides the particle's search towards its own "best" trade-off.
    - Global Best: A mechanism is employed to identify a leader *gbest* from the entire swarm based on dominance. This leader can be the single best solution

Algorithm 2: Multi-Objective Particle Swarm Optimization (MOPSO)

Initialize particle position $x_i = (x_{i1}, \cdots, x_{id})$
Initialize velocity $v_i = (v_{i1}, \cdots, v_{id})$ for all particles
Initialize external archive A (empty)
for $i = 1$ **to** ps **do**
    Evaluate fitness vector $f(x_i)$
    Set $pbest_i = x_i$
    if *Non-dominated by any solution in* A **then**
        Add x_i to archive A
    end
end
Find a leader from A and set $gbest = x_{leader}$
for $t = 1$ **to** t_{max} **do**
    for $i = 1$ **to** ps **do**
        Update velocity v_i
        Update position x_i
        Evaluate fitness vector $f(x_i)$
        if *Non-dominated by any solution in* $A \cup \{x_i\}$ **then**
            Set $pbest_i = x_i$
            Add x_i to archive A (update or add)
        end
        Update leader and $gbest$ based on dominance criteria
    end
    Update archive A (remove dominated solutions)
end
Set S = non-dominated solutions in A
Return S

found so far or a representative solution from a collection of non-dominated solutions depending on the specific MOPSO variant. The ***gbest*** guides the swarm's exploration towards promising regions of the search space.

5. Velocity and Position Update: Similar to PSO, MOPSO updates the velocity and position of each particle based on its pbest, gbest, and its current state. The update equations incorporate inertia, cognitive (based on pbest), and social (based on gbest) components, influencing the particle's movement towards potentially better solutions.

6. Archive Management: Some MOPSO variants maintain an external archive to store non-dominated solutions found during the search. This archive helps preserve diversity and prevents the loss of good solutions.

7. Iteration: The process of fitness evaluation, dominance check, pbest and gbest update, and velocity/position update continues for a predefined number of iterations.

MOPSO emerges as a powerful optimizer for problems with conflicting objectives. It tackles multi-objective landscapes effectively, guiding a swarm of particles toward a diverse set of well-balanced solutions. This is achieved through the concepts of personal best ***pbest*** and global best ***gbest***, where each particle learns from its own best experience and a leader identified from the entire swarm. Additionally, MOPSO's core principles borrow from the well-established PSO algorithm, making it conceptually accessible. However, MOPSO is not without limitations. The choice of parameters and dominance criteria can significantly impact its performance. Furthermore, for problems with many objectives, dominance checks, and archive management can become computationally expensive. Finally, some variants might struggle to converge to the optimal trade-offs in complex scenarios. Despite these considerations, MOPSO remains a robust tool for tackling multi-objective challenges across various fields like engineering design, resource allocation, and machine learning.

Building on the success of NSGA-II and MOPSO, researchers have developed various multi-objective metaheuristics. These include adaptations of nature-inspired algorithms such as: Multi-Objective Grey Wolf optimization (MOGWO)[92], Multi-Objective Slime Mould Algorithm Based on Elitist Non-Dominated Sorting(MOSMA)[105], Multi-objective Salp Swarm Algorithm (MSSA)[87], Multi-Objective Firefly Algorithm (MOFA)[149], Multi-Objective Ant Colony Optimization [117]„ Multi-Objective Flower Pollination Algorithm (MOFPA)[148], Multi-Objective Cat Swarm Optimization (MOCSO)[103], Multi-Objective Spotted Hyena Optimizer (MOSHO)[37], Multi-Objective Seagull Optimization Algorithm(MOSOA)[40], Multi-Objective Artificial Bee Colony algorithm [59], , Multi-Objective Gradient-Based Optimizer for real-world structural optimization problems(MOGBO)[104],

Additionally, hybrid approaches like PSO combined with Sine Cosine Algorithm and Levy flight have been proposed [20]. Recent developments also include the Gradient-Based Optimizer for structural optimization and variations on existing algorithms, such

as the Non-Dominated Sorting Grey Wolf Optimizer (NS-GWO)[67]. New evolutionary multi-objective seagull optimization algorithm for global optimization(EMoSOA) [39] and Multi-Objective Ant Lion Optimizer (MOALO) [88].

Reviewing these works reveals a common approach: extending single-objective algorithms to handle multiple objectives by incorporating various techniques. For instance, MOCSO adapts the standard Cat Swarm Optimization (CSO [23]) for multi-objective problems by integrating Pareto ranking [103]. The NS-GWO algorithm enhances Grey Wolf Optimization with an external archive, crowding distance mechanism, and Pareto dominance-based selection strategy [67].

MSSA extends the Salp Swarm Algorithm by adding an external archive to store non-dominated Pareto solutions. It uses a ranking system based on neighboring solutions and a roulette wheel for leader selection to maintain archive diversity [87]. Similarly, MOGWO expands the single-objective GWO [91] by implementing a fixed-size external archive with an adaptive grid mechanism to preserve diversity and manage archive size.

EMoSOA presents an evolutionary multi-objective version of the Seagull Optimization Algorithm (SOA [38]), incorporating a dynamic archive concept with genetic operators like crossover and mutation. It employs grid mechanisms and roulette-wheel methods for leader selection [39]. MOSOA [40] also adapts SOA for multi-objective optimization, integrating an external archive, grid mechanism, and roulette wheel-based leader selection.

MOGBO introduces an elitist non-dominated sorting mechanism with a crowding distance diversity operator to enhance Pareto front coverage, particularly for truss-bar design problems [104]. The GPAWOA algorithm integrates an external archive into the Whale Optimization Algorithm (WOA) to improve convergence while using the crowding distance concept for solution diversity [4].

These adaptations demonstrate the versatility of single-objective algorithms in addressing multi-objective problems through strategic modifications and additional mechanisms.

1.6 Conclusion

In this chapter, we presented generalities about optimization, with a focus on multi-objective problems. We discussed the main approaches to addressing these problems, highlighting the complexity involved in resolving them using classical techniques. To handle this complexity, we introduced the concept of formulating the multi-objective problem as a multi-objective optimization problem. We examined various methods for solving multi-objective optimization problems, including exact methods, metaheuristics, non-Pareto, and Pareto algorithms. Each method has its strengths and applicability depending on the problem at hand. While exact methods and metaheuristics have their advantages and limitations, metaheuristic algorithms, particularly Pareto-based ones, are often preferred for their ability to explore complex, non-convex, or multi-modal solution

spaces efficiently. They provide a diverse set of well-balanced solutions by explicitly considering Pareto dominance and optimizing for Pareto optimality, especially when dealing with complex Pareto fronts or conflicting objectives.

This discussion underscores the motivation behind our focus on developing a multi-objective metaheuristic algorithm to address continuous optimization problems. Specifically, we propose the Multi-Objective Henry Gas Solubility Optimization (MOHGSO) algorithm, which will be presented in detail in the contributions section.

CHAPTER 2

Multi-Objective Feature selection

2.1 Introduction

Multi-objective feature selection has emerged as a pivotal technique at the intersection of machine learning and metaheuristics, that leverages the strengths of machine learning and metaheuristic optimization to address the challenges of feature selection in high-dimensional datasets. By combining these approaches, multi-objective feature selection can significantly enhance model development and efficiency. This section aims to explore existing feature selection techniques, their ability and effectiveness particularly in classification tasks across various domains, including bioinformatics, image processing, and text classification, where effective feature selection is crucial for building accurate models.

2.2 Types of Machine Learning

In recent years, the exponential growth of data has posed significant challenges in various fields, including machine learning which involves developing algorithms that can learn from data by analyzing and processing sample data which leads to making predictions or decisions based on new or unseen data, and also data mining applications that focus on discovering patterns and relationships from large datasets and involve supervised learning (such as classification) and unsupervised learning (such as clustering) when high dimensional data are used as input in learning and knowledge-based systems.

The process of a learning algorithm has a two-phase approach consisting of a learning phase and a deployment phase. The algorithm is trained on a labeled training dataset to generate a prediction model during learning. The trained model from the learning phase is then used to make predictions or decisions on new or unseen data during deployment. The

quality and accuracy of the decisions made during deployment rely on the performance of the model generated during the learning phase. Moreover, the division of the input data into a training set (where the model is trained) and a testing set (evaluating the model's performance) is a common practice in machine learning, the evaluation of the model's performance on the held-out testing set ensures the model can effectively generalize to new data [5, 95].

In the literature, there are several types of machine learning (see Figure 2.1), including supervised, unsupervised, semi-supervised, and reinforcement learning, which will be presented in the following sections.

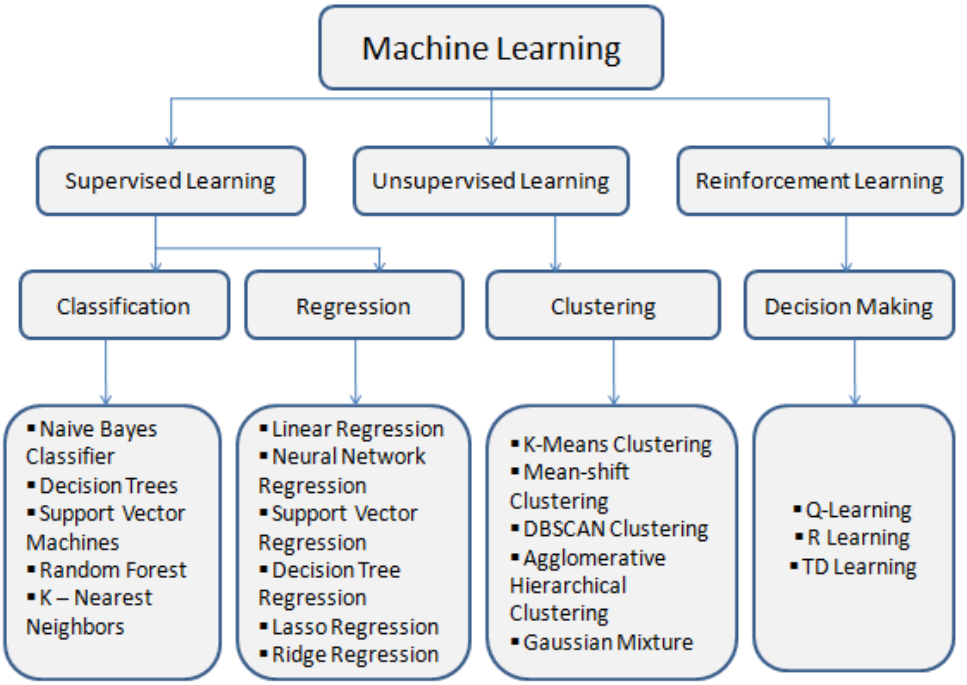


Figure 2.1: Types of Machine Learning.

2.2.1 Supervised learning

Supervised learning is a powerful technique in machine learning that works by learning the algorithm from labeled data, where each input example is paired with a corresponding target label. In another meaning, in supervised learning, the training data consists of inputs which are features or independent variables, and also outputs which are the corresponding labels or target variables. The goal is to build a learning model that can accurately predict the output label (target variable) for new or unseen input data [41]. Supervised learning can be used for various tasks, including:

- **Classification:** Predicting a category or class label for new data (e.g., spam email detection, image recognition) based on its features, where the output is a discrete category or class label.
- **Regression:** Predicting a continuous value for new data (e.g., weather forecasting, stock price prediction) based on its features, regression aims to estimate a numerical quantity.

Supervised learning is a fundamental concept in machine learning, it is widely used in various applications such as image recognition, natural language processing, spam detection, credit risk assessment, and medical diagnosis. Supervised learning algorithms include Linear Regression, Logistic Regression, Decision Trees, Support Vector Machines, and K-Nearest Neighbors algorithms. To name a few, two popular supervised learning algorithms will be presented in the following subsections.

2.2.1.1 K-nearest neighbor (KNN) algorithm

K-Nearest Neighbors (K-NN) is a popular algorithm widely used in supervised learning for classification and regression tasks. Its adoption is motivated by the ease of implementation, simplicity, and low computational cost. k-NN algorithm is a non-parametric method, i.e. it makes no assumptions about the distribution of the underlying data [24].

For the classification case, the algorithm stores all the data points from the training set with their associated class labels. For a given new data point, the algorithm finds the k nearest neighbors by calculating the distance between that point and all other points in the training dataset, usually Euclidean distance is used, but other metrics like Manhattan distance or cosine similarity can also be used [22]. K-NN assigns the most frequent class label among the K neighbors to the new data point. For regression case, the predicted continuous value for the new data point is the average (or weighted average) of the target values of its k nearest neighbors (see Figure 2.2).

The choice of the value of K, the number of neighbors to consider, is crucial for K-NN's performance. It is a hyperparameter that needs to be chosen before applying the algorithm, which is not learned from the data during the training process, selecting an appropriate value is essential for better classification results. Indeed, setting a high value for k means that the prediction for a new instance might be influenced by more distant neighbors (the irrelevant data points), and the model may overlook local patterns in the data, leading to underfitting and poor performance. On the other hand, a low value of K can lead to different predictions for nearby instances, resulting in a more complex decision boundary. While this can capture local patterns better, it may also lead to overfitting, where the model becomes too sensitive to noise or outliers in the training data. Despite its simplicity, k-NN is a powerful classifier that has been successfully applied in several areas for example in forestry [51], healthcare [80], finance [21], and image and video recognition[98].

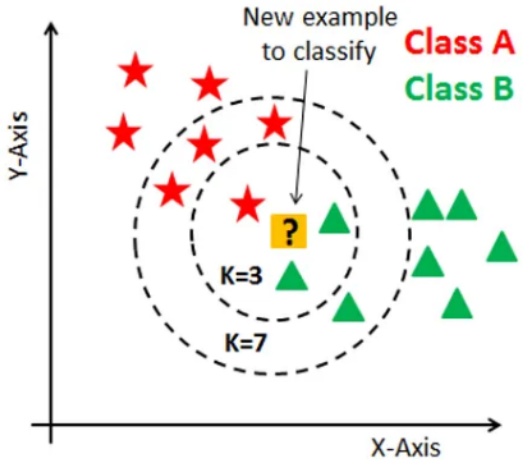


Figure 2.2: K-Nearest Neighbor classification (K-NN) algorithm.

The major advantage of the k-nearest neighbor algorithm is its simplicity of understanding, ease of implementation, ability to work well with various data types, and effectiveness in the classification and regression tasks. On the other hand, the computationally expensive for large datasets, storing all the training data, and sensitivity to the choice of the value of k are the main disadvantages of the k-nearest neighbor algorithm.

2.2.1.2 Support Vector Machine (SVM)

Support Vector Machine algorithms (SVMs) are a type of supervised learning algorithm, they are versatile and usually used for classification tasks, SVMs can handle both linearly separable and non-linearly separable data. The goal is to find the best separation (the optimal hyperplane) that separates different classes of data points by maximizing the distance between the decision boundary and the most critical data points, the support vectors which are the nearest data points that define the margins (see Figure 2.3). SVMs are powerful tools for machine learning applications, they are able to avoid overfitting even with complex and high-dimensional data [30]; [13].

SVMs are applied to linearly and non-linearly separable classification problems through the use of different kernel functions, which allow SVM to find a hyperplane in the higher-dimensional space that can separate the classes linearly.

The linear kernel is used for linearly separable data and corresponds to performing SVM in the original input space. Non-linear kernels like polynomial or radial basis function (RBF) kernels are used for non-linearly separable data, where a more complex boundary is needed.

The general formula for the separating hyperplane in SVM with a kernel function is:

$$h(x) = \sum_{k=1}^{n} a_k y_k K(x, x_k) + b$$

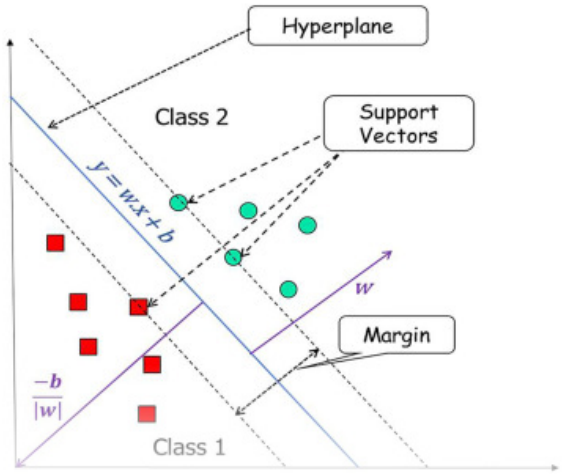


Figure 2.3: Support Vector Machine (SVM) algorithm

where $h(x)$ represents the decision function that classifies a new data point x, this function can be expressed using the kernel trick:

$$h(x) = w^T \phi(x) + b$$

$\phi(x)$ represents the mapping of the data point x from the original input space to the higher-dimensional feature space, and w is the weight vector.
a_k are the Lagrange multipliers obtained during the optimization process, y_k are the class labels of the support vectors, and $K(x, x_k)$ is the kernel function that computes the inner product of the transformed data points in the higher-dimensional space.

Different types of kernel functions can be used in SVM, such as:[114]

- Linear Kernel: $K(x, x_k) = x^T x_k$, it is used for linearly separable data;
- Polynomial Kernel: $K(x, x_k) = (x^T x_k + c)^d$, this kernel maps data points to higher-dimensional polynomial terms, allowing for the creation of curved decision boundaries;
- RBF (Radial Basis Function) Kernel: $K(x, x_k) = \exp(-\gamma \|x - x_k\|^2)$, this kernel computes the similarity between data points using a radial basis function.

SVM has been employed across numerous regression and classification real-life applications, including pattern recognition in [15], and medical diagnostics in [126] and [102]. The major advantage of the support vector machine algorithm is its effectiveness in high-dimensional spaces, it is productive for problems with a large number of features compared to the number of samples, and it is most effective in cases with a clear margin of separation between classes. SVM uses a subset of training points in the decision function (support vectors), which makes it memory efficient, especially when dealing with large

datasets. On the other hand, time-consuming training is computationally expensive for large datasets and the sensitivity to the noise data are the main disadvantages of the support vector machine algorithm.

2.2.2 Unsupervised learning

Unsupervised learning is a type of machine learning, where the data doesn't have predefined categories or classifications (labels), the algorithm deals with raw, unlabeled data, and the goal is to find patterns and structure within this raw data which makes it more challenging than supervised learning.

Since there are no predefined target classes (no labeled training data), an unsupervised learning algorithm receives input data without corresponding output labels, the goal is to explore the structures of the data by discovering patterns such as clusters of similar data points, meaning uncover distributions in this data by analyzing similarities and differences between data points. Each cluster represents a collection of data points that are more similar to each other according to a specified similarity criterion, this means maximizing the intra-cluster similarity and minimizing the inter-cluster similarity which guarantees the dissimilar between the different clusters. Similarity Measures are used in unsupervised learning algorithms to determine how similar data points are, often relying on distance functions that calculate the distance between two data points in a mathematical way including Euclidean distance or Manhattan distance [65].

Unsupervised learning algorithms are powerful tools in machine learning, they vary in their approaches and methodologies regarding the data structure for discovering patterns and clusters, such as the number of clusters (k) to be formed, the shape or geometry of the clusters, and the distance (similarity) metric used to measure closeness between data points which makes them suitable for various types of problems. Some popular clustering algorithms include K-means, hierarchical clustering, Density-Based Spatial Clustering of Applications with Noise (DBSCAN), and Gaussian mixture models. To name a few, two popular unsupervised learning algorithms will be presented in further subsections.

2.2.2.1 k-means algorithm

K-means algorithm is a popular unsupervised learning algorithm, it is one of the most widely used clustering algorithms in many practical applications due to its simplicity, speed, and effectiveness [41]. K-means algorithm is an iterative algorithm that performs to partition the dataset into k clusters. Each cluster is represented by its centroid meaning assigning data points to the nearest centroid and updating the centroids until convergence.

k-means algorithm groups similar data points by minimizing the distance between this data and its assigned cluster's center, which ensures the objects within the same cluster are very similar and objects from different clusters are dissimilar (see Figure 2.4). Here's a basic outline of how the k-means algorithm works [115]:

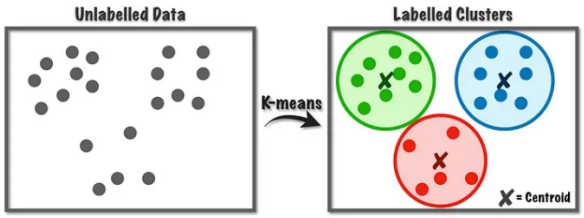


Figure 2.4: Clustering using K-means algorithm

1. Initialization step:
   - Define the Number of Clusters (k), this step specifies the desired number of clusters beforehand;
   - Randomly initialize centroids (points that represent the center of each cluster).
2. Assignment step:
   - Assign each data point to the nearest cluster centroid based on a distance metric (usually Euclidean distance);
   - Calculate the distance between each data point and each centroid;
   - Assign each data point to the cluster with the closest centroid.
3. Update step:
   - Recalculate the centroids of the K clusters based on the mean of all data points assigned to each cluster;
   - The new centroid is the average of all the points in that cluster.
4. Repeat step:
   - Repeat the Assignment and Update steps iteratively until convergence is reached;
   - The algorithm converges when centroid movement becomes significantly small or when a specified number of iterations is achieved.

The algorithm converges when the positions of the cluster centroids stabilize (i.e., change minimally between iterations) or when another stopping criteria are met. Then, the quality of the clustering results is evaluated using specific measures (clustering metrics). Note that the algorithm is easy to implement, making it a good choice for clustering tasks in various domains. Despite that, its performance is sensitive to the number of clusters (k) that need to be pre-specified and the initial centroid selection. However, running the algorithm multiple times with different random initializations and choosing appropriate distance metrics may be necessary to find and identify the optimal clustering result.

2.2.2.2 Hierarchical clustering algorithm (HCA)

Hierarchical clustering is an unsupervised learning technique, a classical method of clustering algorithms that groups similar data points based on a predefined distance measure. Its principle involves merging the closest data points (clusters) based on their similarities or dissimilarities iteratively, which creates a hierarchical structure called a dendrogram (see Figure 2.5). The dendrogram illustrates the hierarchical relationships between the clusters, with each level representing a different data partitioning. Higher levels represent more general clusters, lower levels result in more specific clusters [139].

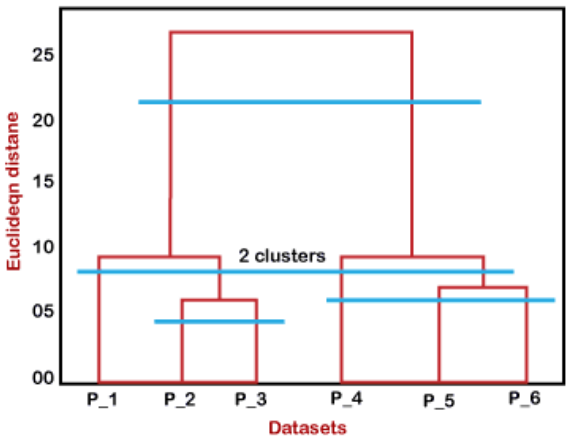


Figure 2.5: Working of Dendrogram in Hierarchical Cluster Analysis (HCA).

The process usually starts with each data point forming its cluster, it iterates the following steps:

- Compute the pairwise distance matrix, meaning calculate the distances between all pairs of clusters using a suitable distance metric (Euclidean distance, or Manhattan distance);
- Identify and merge the closest clusters based on the distance measure in order to form a new cluster;
- Update the distance matrix by recalculating the distances between the new cluster and all other clusters;
- Repeat the merging and updating process (steps 2 and 3) until only one cluster remains.

The goal of hierarchical clustering algorithms is to identify natural groupings or patterns in the data, making it a valuable tool for exploratory data and insightful in various fields, such as bioinformatics. Unlike certain clustering algorithms, hierarchical clustering doesn't necessitate determining the number of clusters beforehand. However, its computational complexity can be high, especially when dealing with large datasets.

2.2.3 Semi-supervised Learning

Semi-supervised learning is a machine learning paradigm that fills a gap between supervised and unsupervised learning by leveraging labeled and unlabeled data for training models. In many real-world problems, there may be a small amount of labeled data but a vast amount of unlabeled data, so semi-supervised learning is a powerful technique for this situation to build better machine learning models, where the algorithms can extract valuable information and patterns from this unlabeled data to improve the model by using the limited labeled data available which can be expensive while capitalizing on the is plentiful unlabeled data that may be readily accessible.

The model is first trained on the limited labeled data using a supervised learning algorithm like linear regression, or decision tree in order to provide the relationship between the features and the labels (a basic knowledge about the problem). Then, the model is utilized to make predictions on the vast amount of unlabeled data, it focuses on the model's most confident predictions and treats them as labeled data which are used to refine the model iteratively, allowing the model to learn from the unlabeled data, and improving its performance.

2.2.4 Reinforcement learning

Reinforcement Learning (RL) which is called learning through experience is a branch of machine learning that represents a contemporary approach to machine learning problems, it focuses on learning by interacting with an environment. In reinforcement learning, the agent interacts with a dynamic environment to learn behavior, learns through trial and error, and identifies optimal actions for future states based on those actions (its own experiences).

Reinforcement Learning involves producing intelligent programs (agents) through several steps. Initially, the agent observes the current state of the environment. Then, it employs a decision-making function to choose an action. After executing the action, the agent receives feedback, typically in the form of a reward or reinforcement, from the environment. This information, linking states and actions to rewards, is stored for future learning and decision-making processes (see Figure 2.6).

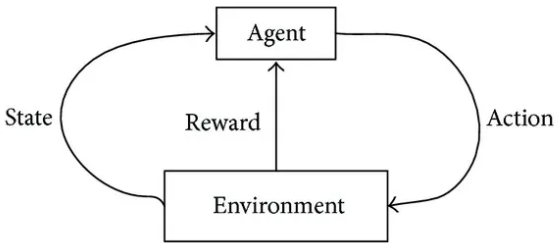


Figure 2.6: Reinforcement learning Model.

Key concepts include trading off exploration (trying new actions) and exploitation (taking actions known to be good), Markov decision theory, learning from delayed reinforcement, and constructing empirical models. Unlike supervised learning with limited environmental interaction (predefined datasets), where the training data has explicit input-output pairs, the agent engages extensively with the environment to evaluate the effectiveness of its past actions where there are no prescribed correct outputs, thus informing its subsequent decisions. Consequently, the environment serves as a benchmark for the agent, furnishing performance evaluations crucial for guiding its learning process. this process allows the agent to continuously improve its decision-making capabilities [69], [48].

Reinforcement Learning is widely studied in various disciplines such as game theory, control theory, and operations research. It has found success in various real-world applications such as board games, robotics, and self-driving cars. These applications demonstrate the ability of RL algorithms including adversarial inverse RL (learning reward functions) and kernel-based RL algorithms for efficient learning to learn complex behaviors and decision-making strategies in diverse and complex environments. Common RL algorithms include Q-learning and Temporal Difference (TD).

2.2.4.1 Q-learning algorithm

Q-learning is a widely used reinforcement learning algorithm that learns the value (expected future reward) of an action in a specific state without requiring a model of the environment, making it suitable for complex or dynamic environments where building an accurate model might be challenging. It can effectively handle stochastic (random) transitions and rewards in the environment without needing adaptations.

The algorithm computes the expected rewards for an action taken in a given state, aiming to maximize the total reward the agent receives over successive steps over time. Variants of Q-learning, such as Double Q-learning and Weighted Double Q-learning, have been developed to address issues like overestimation or underestimation of action values, which can lead to suboptimal performance [66].

2.3 Dimensionality reduction

Dimensionality reduction is a crucial technique in data analysis that involves transforming data from a high-dimensional space to a lower-dimensional space while preserving important information from the original data and retaining meaningful properties of these data. This process is essential due to the challenges posed by high-dimensional data (Curse of Dimensionality), such as computational complexity and data sparsity (fewer data points compared to dimensions) [52].

Dimensionality reduction is a crucial preprocessing step in machine learning to eliminate irrelevant and redundant data, enhancing learning accuracy and result interpretabil-

ity. Dimensionality reduction methods can be categorized into linear approaches that focus on preserving linear relationships, and nonlinear approaches that can handle more complex relationships in the data, as well as feature selection and feature extraction techniques [9].

Feature selection and extraction techniques play a vital role in the dimensionality reduction process. Feature selection aims to identify a subset of input variables (relevant features) from the original data using strategies like the filter, wrapper, and embedded approaches. On the other hand, feature extraction transforms data into a lower-dimensional space using methods including principal component analysis (PCA) which is a widely used linear technique to maximize the variance of data in the low-dimensional representation, and other nonlinear techniques [32].

This reduction process offers several advantages for various applications such as noise reduction that leads to more robust models, data visualization for gaining insights, and cluster analysis, dimensionality reduction can improve the performance of classification and clustering algorithms. It is a valuable tool commonly applied in various fields such as signal processing, speech recognition, and bioinformatics, allowing researchers to handle and analyze complex datasets effectively and efficiently.

Dimensionality reduction based on feature selection is a critical technique in data preprocessing, particularly when dealing with high-dimensional datasets, selecting the most informative individual features (a subset of relevant features) is a crucial step to reduce the number of input variables in a dataset, which can lead to several benefits for machine learning models, including better performance, reduced computational cost and avoiding the curse of dimensionality. In this thesis, we focus on achieving feature reduction through selection using metaheuristic algorithms. To achieve this, we will provide a concise introduction to feature selection and metaheuristics in the following sections.

2.4 Feature Selection with Metaheuristics

2.4.1 Feature Selection in Machine Learning

Feature selection is a technique for selecting a subset of relevant features that are the optimal set of input features from a given dataset. The optimal feature set contains the fewest features, which are relevant and non-redundant allowing decreasing the dimensionality of feature space, which definitely leads to simplifying the learned classifiers. This reduces the data's dimensionality which can improve the performance of machine learning models in data mining and reduce computation time. The goal is to find a good balance between the number of features and model performance. Feature selection simplifies models by focusing on the most informative features. Thus, an efficient algorithm is required for identifying those input variables that are most relevant to the classification task [143].

There are three main types of feature selection algorithms: filter-based, wrapper-based, and embedded methods (see Figure 2.7). Filter-based methods employ statistical measures

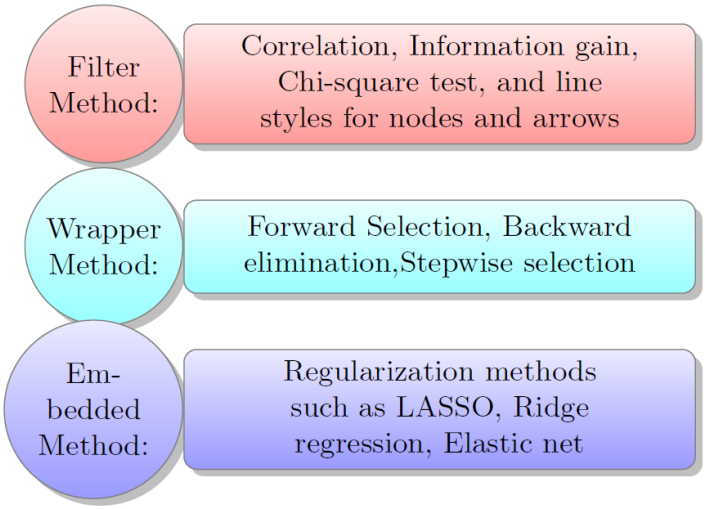


Figure 2.7: Feature selection methods

like Correlation, Distance, Mutual Information, Fisher Score,...etc to evaluate features independently of a specific classifier. Wrapper-based methods use a classifier to assess different feature subsets and select the one that delivers the best performance. While wrapper methods often lead to better accuracy, they can be computationally expensive. Embedded methods combine elements of both filter and wrapper approaches and offer a good compromise in terms of effectiveness and computational cost [18].

1. Filter-based methods offer a fast and computationally efficient approach to feature selection. They are independent of any specific machine learning classifier, making them easy to implement and use across various problems. However, a drawback of these methods is that they might not consider the potential interactions between features and the chosen classifier. This can lead to discarding features that are individually weak in terms of their statistical relationship with the target variable, but might become important when combined with other features. Some common examples of filter-based methods include correlation analysis, Chi-squared test, and Information Gain.

2. Wrapper-based methods stand out for their ability to achieve higher accuracy compared to filter-based methods. This advantage stems from their consideration of the interaction between features and the specific classifier being used. However, this benefit comes at a cost. Wrapper methods involve repeatedly training the classifier with different feature subsets to identify the optimal set. This process can be computationally expensive, especially for large datasets or complex models. Additionally, wrapper methods are more prone to overfitting due to their focus on maximizing the performance on the training data. Examples of wrapper-based methods include Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS).

3. Embedded methods bridge the gap between filter and wrapper approaches. They leverage the strengths of both, offering a good compromise between achieving high performance and maintaining computational efficiency. Embedded methods typically integrate feature selection directly into the model training process. This allows them to consider the interactions between features and the classifier while avoiding the repeated training overhead seen in wrapper methods. A prominent example of embedded methods involves L1/L2 regularization techniques used in penalized regression or classification models. These techniques penalize the model for having large coefficients, effectively pushing some coefficients towards zero and implicitly performing feature selection.

Selecting the optimal feature selection method hinges on several factors, including the specific characteristics of the data, the nature of the machine-learning task, and the available computational resources. Here are some general guidelines to navigate this decision:

- **Large Datasets**: When dealing with vast datasets, filter-based methods shine due to their computational efficiency. Their quick execution time makes them a valuable choice in these scenarios.
- **High-Dimensional Data**: For data characterized by a high number of features (dimensions), feature selection becomes even more critical. Techniques like L1/L2 regularization, which are embedded methods, can be very effective in such cases.
- **Interpretability**: If understanding the relationships between features and the target variable is a priority, filter methods with clear statistical measures like correlation or information gain might be preferable. These methods provide insights into the rationale behind feature selection.
- **High Accuracy**: If achieving the highest possible accuracy is paramount, wrapper methods can be considered. However, be mindful of their computational cost, which can be significant, especially for complex models or large datasets.

We have explored various feature selection methods, each with its advantages and disadvantages. Filter methods offer speed and ease of implementation, while wrapper methods often lead to higher accuracy. However, both approaches encounter limitations - filter methods might miss feature interactions, and wrapper methods can be computationally expensive for large datasets. This is where metaheuristic algorithms come into play. Metaheuristics are a class of optimization algorithms inspired by natural processes like evolution, swarm intelligence, and simulated annealing. They excel at tackling complex optimization problems, including those encountered in feature selection.

Metaheuristic algorithms present a compelling approach to feature selection, addressing some of the limitations encountered with traditional methods. Unlike filter-based

methods, which might overlook the interactions between features, metaheuristics can account for these interactions, potentially leading to more effective feature selection decisions. This capability translates into improved model performance. Furthermore, compared to computationally expensive wrapper methods, metaheuristics offer a significant advantage. They can achieve good results with a lower computational burden, making them particularly suitable for handling large datasets. Additionally, metaheuristics boast flexibility, allowing them to be adapted to various feature selection problems and work effectively with different data types. This versatility makes them a valuable tool in the feature selection toolbox [12].

In the following section, we'll delve deeper into how metaheuristic algorithms can be applied to feature selection. We'll explore specific algorithms like Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO) and discuss their effectiveness in this context.

2.4.2 Metaheuristics for Feature Selection

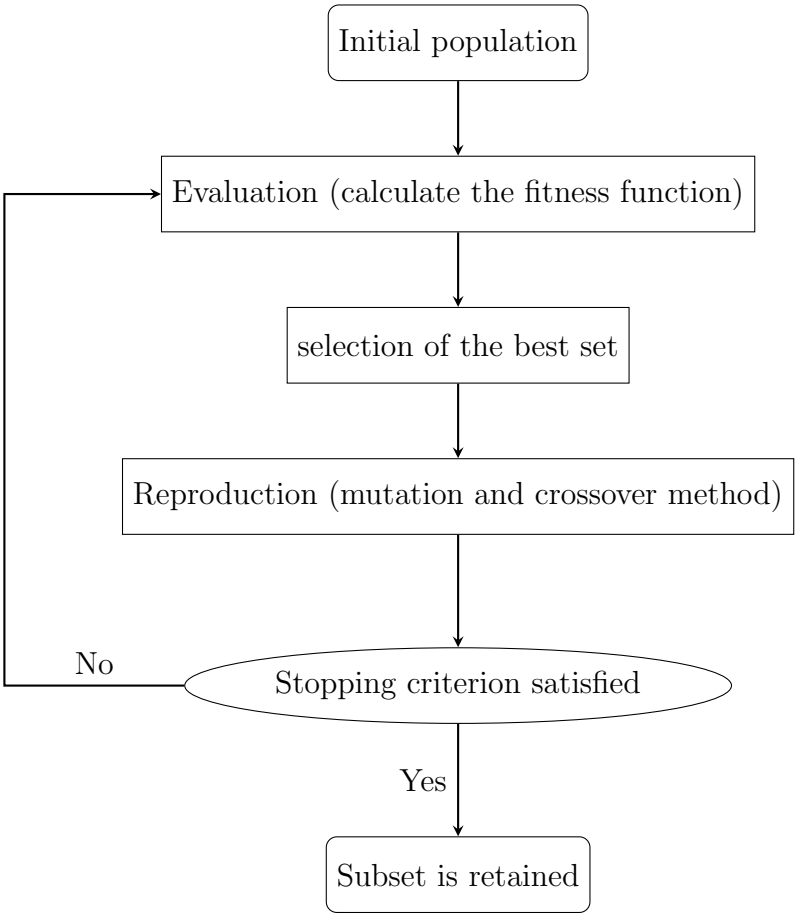
Feature selection approaches involve choosing a subset of the most relevant features from the original dataset. This process is achieved by evaluating the importance of each feature and discarding the redundant ones. Various techniques are used for feature selection, like filter methods that employ statistical properties to rank features, and wrapper methods that involve training models with different feature subsets to identify the best-performing ones.

Classification problems can be viewed as optimization tasks because choosing the right features for classification (the least number of features) while still achieving an accurate classification (maximizing classification accuracy) reflects this optimization perspective. Feature selection is a challenging problem, especially for high-dimensional datasets. Due to the exponential increase in possible feature subsets as the number of features grows, finding the optimal subset is computationally expensive and not practical. This makes feature selection an NP-hard optimization problem and a challenge for traditional optimization methods. Metaheuristic algorithms, like Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO), etc. are effective and commonly used for feature selection tasks. A specific type of metaheuristic algorithm such as stochastic approximation algorithms, offers a valuable alternative and can be just as effective and provide solutions close to the optimal solution in a reasonable amount of time [12].

These algorithms are inspired by natural processes like evolution, swarm intelligence, and simulated annealing. Here are some of the most popular metaheuristic algorithms proposed in the literature for feature selection:

Genetic Algorithms (GAs): are a type of optimization algorithm inspired by Charles Darwin's theory of evolution, they mimic the process of natural selection. An initial population of candidate solutions (feature subsets) evolves over multiple generations. Each candidate solution is evaluated based on a fitness function (perfor-

Figure 2.8: Genetic Algorithm (GA) for feature selection



mance on a machine learning task), and the fittest individuals are selected from the current population (the natural selection process) for reproduction. Through evaluation, selection, crossover, and mutation operators, new solutions are created (offspring' feature subsets), and the cycle continues. This iterative process helps find feature subsets that improve the machine learning model performance. Genetic Algorithm iteratively evaluates, selects, and reproduces candidate feature subsets, continuously improving them until a satisfactory solution is found or a stopping condition is met (see Figure 2.8). GAs are useful for feature selection and have been successfully utilized in [97, 151], a new and fast rival genetic algorithm for feature selection [132], Optimal column subset selection for image classification by genetic algorithms [74].

Particle Swarm Optimization (PSO): is a swarm intelligence-based optimization technique inspired by the behavior of bird flocks or fish schools. A swarm of particles is initialized to explore the solution space, where each particle represents a candidate feature subset. It iteratively updates its position based on its own best solution (pbest) and the best solution found in the entire swarm (gbest), allowing the swarm to converge on optimal or near-optimal feature subsets for classification tasks (see Figure 2.9). PSO is a robust and efficient optimization technique that can effectively explore the high-dimensional search space of feature subsets, it has been employed in [25, 94, 144], a recent binary Particle Swarm Optimization Approach [68], Self-adaptive parameter and strategy-based particle swarm optimization[146].

Grey Wolf Optimizer (GWO): is a swarm intelligence optimization algorithm inspired by the social hierarchy and hunting behavior of grey wolves in nature. It uses a swarm of wolves (candidate solutions) to search for the best feature subset. The algorithm mimics the leadership structure of grey wolf swarms, where alpha wolves represent the current best solution (lead the hunt), beta wolves assist them, and omega wolves follow. The hunting mechanism involves encircling, hunting, and attacking the prey (optimal solution). The positions of the alpha, beta, and omega wolves guide the movement of the rest of the swarm toward promising areas of the search space. The updating process allows the swarm to converge on optimal or near-optimal feature subsets. The grey wolves' social hierarchy and hunting behavior help the GWO algorithm effectively explore the search space and avoid local optima (see Figure 2.10). GWO has shown promising results for high-dimensional feature selection problems with complex search landscapes, it has been successfully employed in [121, 152, 133], binary variants (BGWO) in [45], and [19]. The hybrid ALO-GWO algorithm for feature selection in datasets with a large number of features but a small number of instances in [152].

Ant Colony Optimization (ACO): is an optimization algorithm inspired by the foraging behavior of ant colonies in nature. A colony of artificial ants is initialized, where each ant represents a feature subset, ants explore the solution space by constructing feature subsets iteratively. Ants deposit pheromones on paths they take, they explore the solution space. Future ants follow paths with stronger pheromone trails, which represent

Figure 2.9: Particle Swarm Optimization (PSO) for Feature Selection

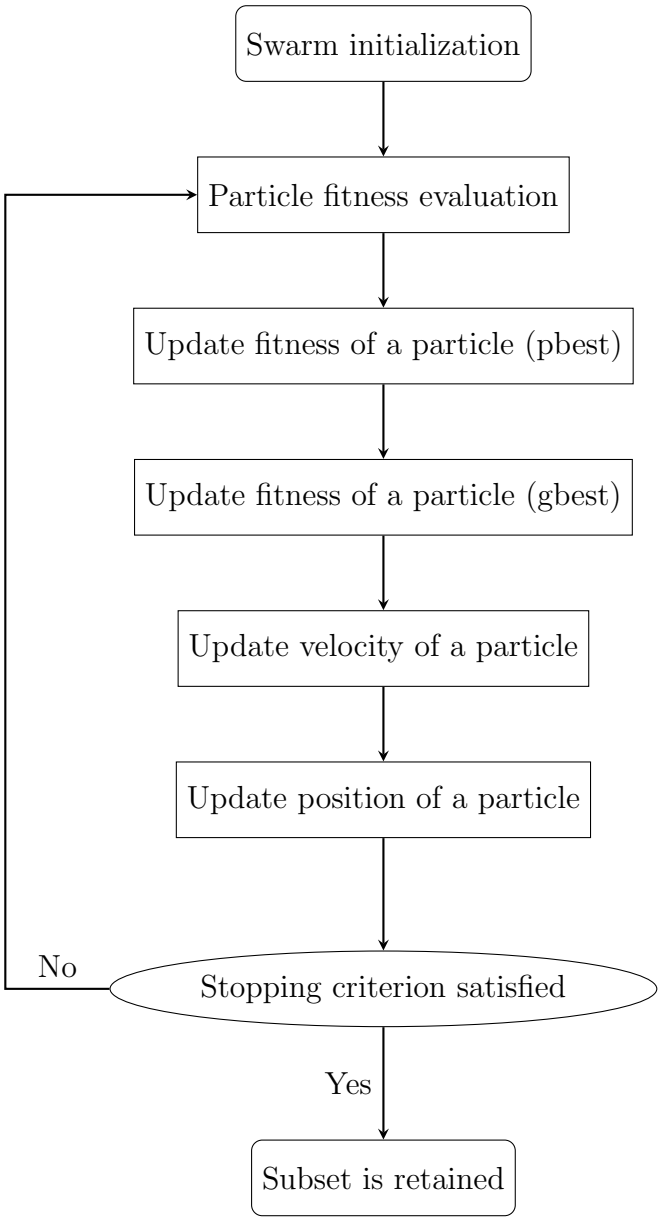
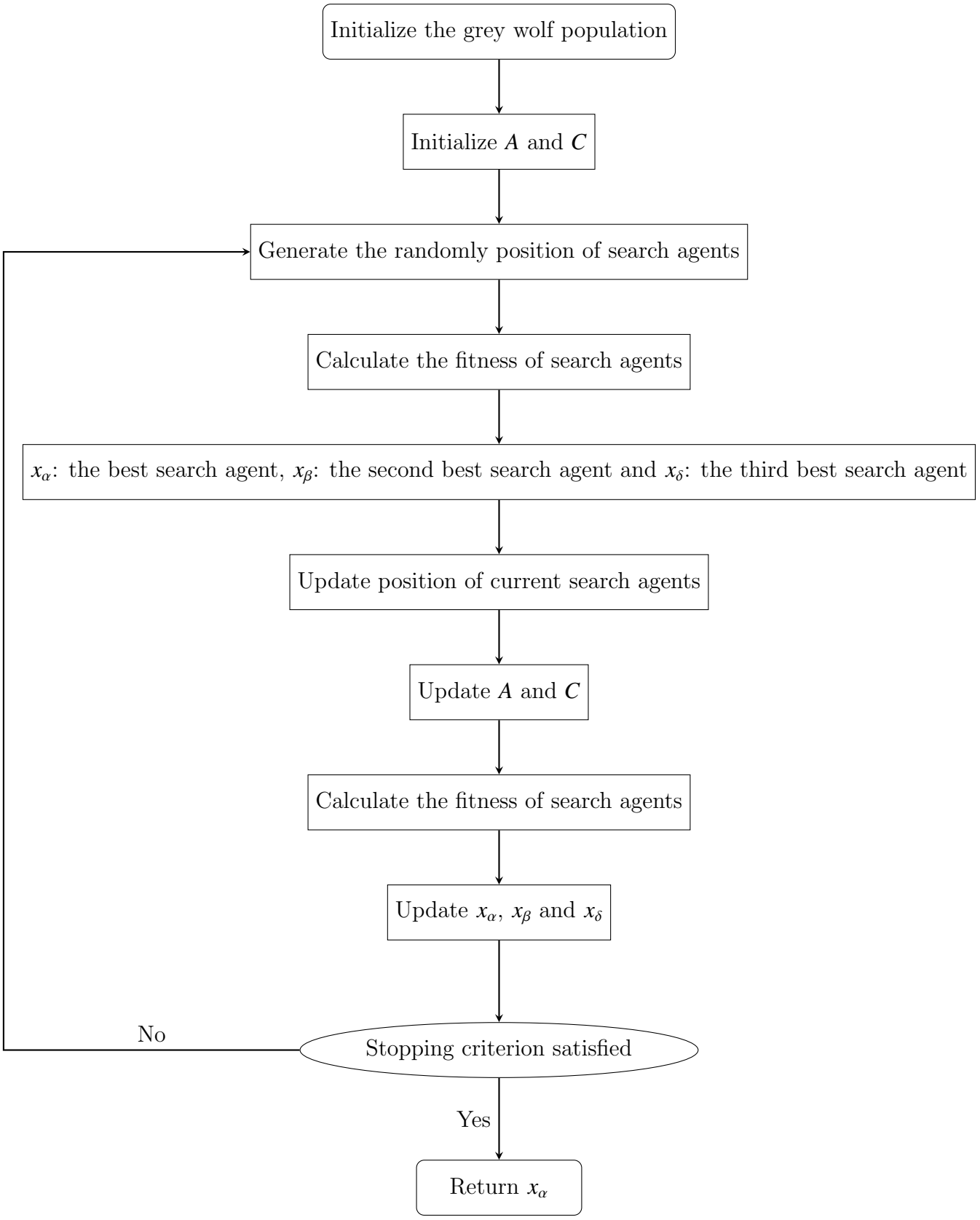


Figure 2.10: Grey Wolf Optimizer (GWO) for Feature Selection



promising feature subsets (e.g., high classification accuracy). This indirect communication allows the colony to converge on good solutions over time, effectively identifying optimal or near-optimal feature subsets for classification tasks (see Figure 2.11) [101].

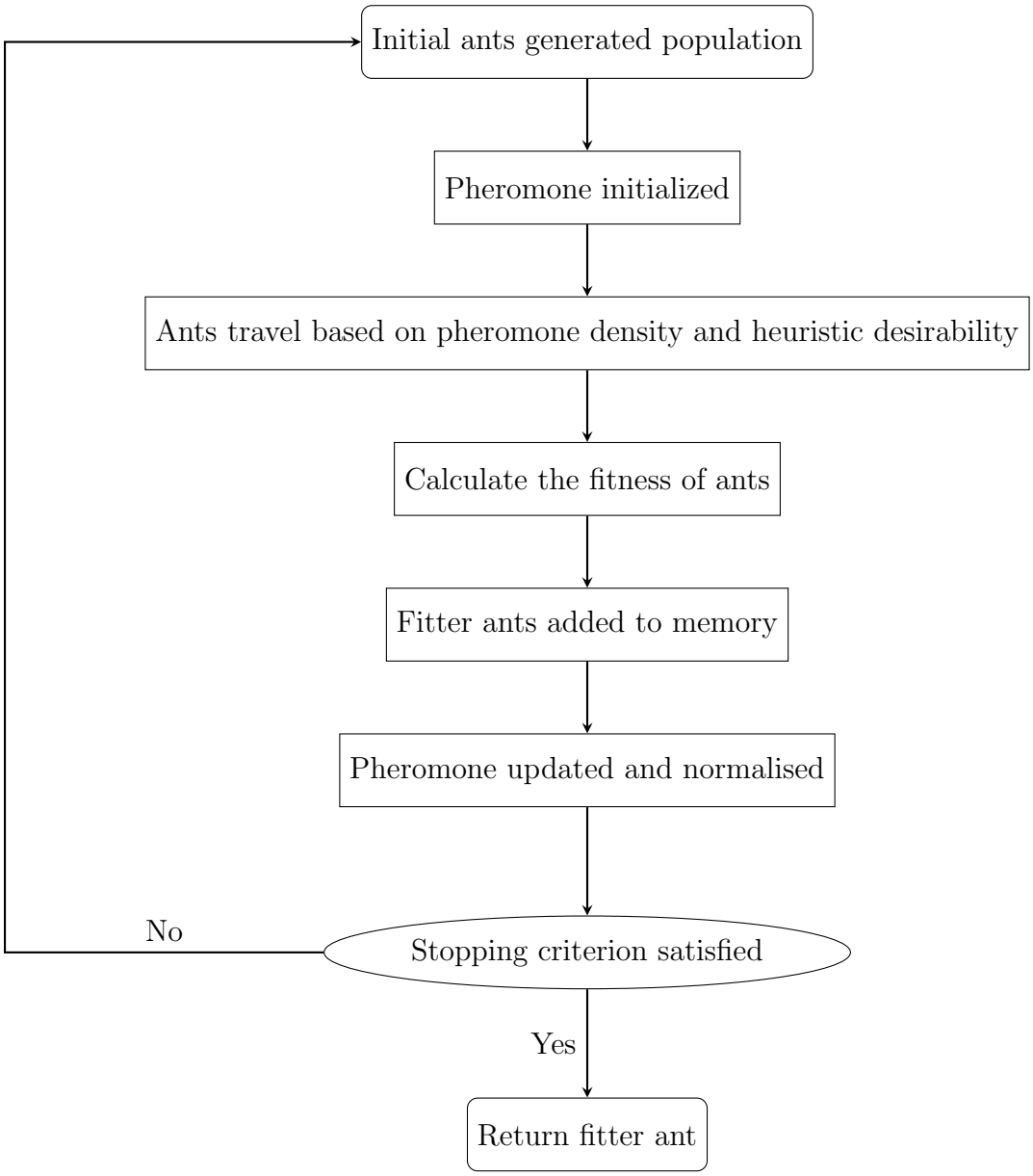
In feature selection, the objective function will impact the behavior of the feature selection approach and the resulting feature subset. The choice between minimizing the feature subset size, maximizing the classification accuracy, or using a merger (weighted sum) of these objectives will depend on the specific problem and the trade-off between reducing the dimensionality of the data and improving the classifier's performance. Several metaheuristic algorithms, such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), artificial bee colony (ABC), and others, have been proposed in the literature for feature selection. They are effective in a variety of applications, these algorithms are effective in a variety of applications.

In recent years, there has been a growing interest in using metaheuristic algorithms for feature selection, as they provide a flexible and powerful way to balance the trade-off between feature subset size and classification accuracy. From an optimization viewpoint, there are several existing works that can be cited, including: three improvement algorithms that were based on the original WOA algorithm: WOA-T and WOA-R used Tournament and Roulette Wheel selection mechanisms, respectively [83]. WOA-CM used crossover and mutation operators are proposed to search the optimal feature subsets for classification tasks [85]. Two improvements to the native ABC algorithm have been applied to the feature selection problems in [60, 118]. FOA has been employed in [50], and FPA in [2] for solving feature selection problems. A binary coded ant colony optimizer [137], an improved harris hawk optimizer (IHHO) [159], another application of the Binary Harris Hawks Optimizer in high-dimensional datasets with low sample sizes, contributing to the field for feature selection by presenting a novel optimization algorithm tailored for this specific challenge, along with experimental results that validate its effectiveness [131] , Gaussian mutational chaotic fruit fly-built optimization [155], a binary version of the Vortex Search Algorithm (VSA) [123], an hybrid binary coral reefs optimization algorithm (BCROSAT) [147], chaotic selfish herd optimizer [8]. In [82], the Binary Dragonfly Algorithm with time-varying transfer functions is applyed to the feature selection problem, by introducing a novel binary optimization technique tailored for this task.

According to existing research on feature selection methods, only a limited number of approaches consider feature selection as a multi-objective optimization problem. To simultaneously address both conflicting objectives (minimize the number of selected features and optimize the model performance), the existing studies have demonstrated that approaches based on metaheuristic techniques can treat feature selection as a multi-objective optimization problem thanks to the dynamic searching behaviors and global search capability.

By taking an overview of these studies, we can say that the most popular multi-

Figure 2.11: Ant Colony Optimization (ACO) for Feature Selection



objective algorithm is PSO-based in the area of feature selection approaches. For example, in [145] the first study on MOPSO for feature selection is suggested, this work presented two separated MOPSO-based algorithms: the first one is based on the concept of non-dominated sorting to address the collection of feature issues, in the second algorithm, the concepts of crowding, mutation, and dominance are applied to identify the non-dominated subsets. In [160] the first prominent attempt in solving cost-based feature selection problems is suggested based on MOPSO to generate the non-dominated subset of features by integrating probability encoding technique, an effective hybrid operator, besides the crowding distance, and the external archive. In [150] an effective multi-objective feature selection algorithm based on PSO is suggested by incorporating a reinforced memory strategy and combined mutations to optimize both the reliability and the classification performance. In [64] a MOPSO-based feature selection with the fuzzy cost is suggested, called PSOMOFS. This method determines a fuzzy dominance relationship as means of comparison, and a fuzzy crowding distance measure for updating the elitist archive and defining the global leader. In[6], a MOPSO-based approach is suggested to classify multispectral satellite images, this work proposes finding simultaneously the most discriminative spectral bands and optimal number of hidden layer nodes. In[162], an evolutionary multi-objective optimization for discretization-based feature selection is proposed, an efficient updating and an adaptive mutation operator are employed to further explore the relevant features, and delete the redundant features. In [110] another feature selection approach is suggested, which focuses on combining the concept of node centrality and the PSO algorithm. In this approach, the original features are shown as a model for graph representation, after which the centrality of the feature for all nodes in this graph is calculated. Finally, the PSO-based search process is utilized for the final feature selection. In [122], two multi-objective optimization algorithms MOPSO and NSGA-II was presented to identify the impressive features for the warfarin dose prediction problem where the Multi-layer Perceptron (MLP) is employed to evaluate the generated subset of features. The results show that MOPSO was more accurate than NSGA-II. In [158] a PSO-based of feature selection for a multi-label classification problem is presented, with the use of the Pareto dominance relation and the crowding distance to search for the feature subsets. An adaptive uniform mutation together with the local learning strategy was employed to enhance the search capability of this algorithm.

Another metaheuristic algorithm inspired by MOGWO is proposed in [7], a binary version of MOGWO is suggested to handle as well as feature selection using the sigmoid transfer function, and a wrapper-based Artificial Neural Network (ANN) is invoked to evaluate the classification efficiency of a subset of selected features. The work in [44] employed GWO based on filter concepts to search for an optimal subset of features by using mutual information. Later, wrapper method used classifier performance to further refine the obtained subsets to better classification performance. In [111] two different approaches to multi-objective binary GWO algorithms are integrated. One is a multi-objective GWO scalarized form and the other is a Non-dominated Sorting GWO, where

wrapper-based feature selection is employed that chooses the optimal textural feature subset for improved cervix lesions classification. In [70] the MOGWO exploration for the multi-robot system is suggested to optimize both searching for the new area and improving the map's accuracy.

In [156], to treat the problem of cost-sensitive features, a multi-objective ABC algorithm is suggested with two archives. An external archive was used to ensure the distribution of non-dominated solutions in the objective space. The leader archive was used to enhance the kind of those solutions in the variable space. In [26] a multi-objective feature selection approach based on the ABC algorithm was suggested. Multi-strategy hybrid search and variable space crowing degree-based search operator are combined for the multi-objective band selection model by considering the bands' information and the correlation between bands [161]. Also, in [138] a multi-objective feature selection approach based on the ABC algorithm was presented in order to balance the accuracy of evaluating individuals and the scale of utilized samples. The K-means clustering-based differential selection strategy and the ladder-like sample utilization strategy are assigned to reduce the computational cost of the suggested approach.

Recently, a hybridization of two algorithms MOFOA and MOHHO was suggested in [3] using the S-shaped transfer function, and MLP-ANN was employed to investigate the classification error for the feature selection problem. A novel hybrid multi-objective feature selection based on a modified moth flame-optimization algorithm is presented in [124], with the combination of the filter-based method and the wrapper-based method to find the optimal feature subset that realizes data description with minor redundancy based on mutual information as well as maintains the classification accuracy.

Based on Differential Evolution algorithms, a multi-objective binary differential evolution algorithm is suggested in[55] to treat the problem of view selection for materializing in the data warehouse. Firstly, the solutions are ranked according to their Pareto dominance levels, to guarantee faster convergence. Then, the diversity among solution vectors in the solution space is calculated. A multi-objective Differential Evolution-based wrapper approach is suggested in [142] to deal with the feature selection problem by reducing simultaneously the classification error rate and the number of features of the data. This approach searches for a set of non-dominated feature subsets, which involves a small subset of features and obtains better classification performance. In [58] a multi-objective differential evolution-based filter method is suggested to search for feature subsets (set of non-dominated solutions) and improve the classification accuracy. In [157] Binary Differential Evolution with self-learning is suggested. Binary mutation operator based on probability difference, OPS operator (One-bit Purifying Search), and the concept of non-dominated sorting operator with crowding distance is all merged in this approach to improve its performance and make it able to deal with feature selection problems. In [57], a novel multi-objective differential evolution method is suggested for performing clustering and feature selection concurrently, aiming to improve the quality of cluster partitions while utilizing a reduced set of features from the data. In [100] a new feature selection

and weighting method assisted with the MOEA/D approach is presented in which, the inter-class and intra-class distances are simultaneously optimized. K-Nearest Neighbor (k-NN) is employed to classify the data points having the reduced and weighted feature set.

Similarly, a multi-objective genetic algorithm is adopted to effectively address filter-based feature selection in classification[141], where four algorithms were developed by applying mutual information and entropy to the NSGA-II and SPEA2 algorithms. In [76] a novel filter-based multi-objective genetic algorithm is suggested for text feature selection. This method is able to select a subset of features with maximum relevance and minimum redundancy by using the Relative Discriminative Criterion and the Correlation Metric to measure the relevance of text features and the redundancy between features, respectively. In [107] a multi-objective genetic algorithm is adopted with cross-generational elitist selection, heterogeneous recombination, and cataclysmic mutation (CHC) for feature selection by combining the idea of non-dominated sorting with CHC genetic algorithm to obtain a set of Pareto subset of features. In [14] an approach based on the NSGA-II algorithm is suggested with the aim of optimizing the classification accuracy, the margin, and the total number of support vectors to select the best kernel for the used data set and its parameters. In addition, an efficient feature selection algorithm based on multi-objective GA is proposed in [54], to select the optimal feature subset which enhances the performance of any pattern recognition problem. In [74] a supervised feature selection algorithm is presented for a multi-label dataset using the mutual information to check local feature selection and GA to select the global optimal feature subset.

2.5 Conclusion

In this chapter, we comprehensively explored the application of metaheuristic algorithms, particularly successful algorithms like Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC), for feature selection. Numerous studies have demonstrated their effectiveness in handling multi-objective optimization problems, making them valuable tools when complete information about the objective function is unavailable. This ability allows for simultaneous optimization of feature subset size and classification accuracy, a key challenge in feature selection.

The future of feature selection seems to lie in exploring novel metaheuristic algorithms and hybrid approaches that effectively balance this trade-off. This underscores the motivation behind our focus on developing a multi-objective metaheuristic algorithm to address feature selection tasks, which will be explored in detail in the contributions section.

CHAPTER 3

Versions and Performance of the Multi-Objective Henry Gas Solubility Optimizer

3.1 Introduction

Recently, a wide variety of metaheuristic approaches inspired by nature has been suggested for finding the optimal solutions or, at least, approximate solutions to the optimum by extending these approaches originally designed to tackle single-objective optimization problems, these algorithms have been enhanced with various techniques to enable them to address the optimization problems with multiple and conflict objectives. For example, Grey Wolf Optimizer (GWO) and Particle Swarm Optimizer (PSO) are recent population-based metaheuristics of recent creation that have attracted the attention of many researchers in the last few years, a wide variety of multi-objective extensions of these methods were recently suggested to prove their high performance. By taking an overview of the success of MOPSO [28] and MOGWO [92] in handling MOPs, we can propose a new multi-objective algorithm by extending a new single-objective algorithm in the literature, which is called Henry Gas Solubility Optimizer.

3.2 Multi-Objective Henry Gas Solubility Optimizer (MOHGSO)

The Henry Gas Solubility Optimizer (HGSO) algorithm is a recently developed population-based metaheuristic algorithm designed for single-objective optimization problems. It draws inspiration from Henry's law, which describes gas solubility behavior in liquids under varying conditions. HGSO has key features including: utilizing a multi-

population concept, enhancing the search space's exploration, and increasing the probability of obtaining well-distributed solutions [62].

These characteristics make HGSO particularly interesting for adaptation to multi-objective optimization, where solution distribution is a critical challenge. HGSO is structured as a global optimization approach, primarily targeting continuous optimization problems. Its fundamental steps involve:

- The initial population is divided into multiple clusters with the same Henry's coefficient;
- The best and optimal gases are obtained with a high fitness value;
- Updating Henry's coefficient and solubility for cluster and gases, respectively;
- The position update is carried out according to the best and optimal gases selected;
- Rank and select the worst agents, then updating their position is performed;
- Update the best gas and optimal gas.

HGSO algorithm is regarded as a global optimization method, and its mathematical formulation is presented as follows:

- **Step1:** Initialization: A starting population of N gas particles, along with their positions, is generated randomly by:

$$X_i(t+1) = X_{min} + r(X_{max} - X_{min}) \tag{3.2.1}$$

where X_i is the position of i^{th} gas, with r is a random coefficient between 0 and 1. X_{min} and X_{max} represent the lower and upper bounds of the decision variables, respectively, while t denotes the iteration counter. Each gas particle i is assigned a Henry's constant of type $(H_j(t))$, a partial pressure $P_{(i,j)}$ of gas i in cluster j, and a constant value of type $j(C_i)$ for $\frac{\nabla_{sol}E}{R}$. These parameters are initialized as follows:

$$H_j(t) = l_1 \times rand(0,1); P_{ij} = l_2 \times rand(0,1); C_j = l_3 \times rand(0,1) \tag{3.2.2}$$

where $l_i = 5E-02$, $l_2 = 100$, and $l_3 = 1E-02$

- **Step2:** In the context of HGSO, the population is organized into clusters, with each cluster containing gases that share similar characteristics and corresponding Henry's constant values (H_j).

- **Step3:** During the evaluation phase, the performance of the j^{th} cluster is assessed using the relevant objective function. After this assessment, the population is ranked to identify the optimal gas within the entire swarm, denoted as x_{best}.

- **Step4:** Over the course of the iterations, the Henry's coefficient is updated according to the formula provided below:

$$H_j(t+1) = H_j(t) \times exp\,(-C_j \times (1/T - 1/T^{\theta}), \tag{3.2.3}$$

where, $T(t) = exp\,(-\frac{t}{t_{max}})$ and $T^0 = 298.15$. H_j is the Henry's coefficient for cluster j, T represents the temperature, while t_{max} indicates the maximum number of iterations.

- **Step5:** During the optimization process, the solubility $S_{(i,j)}$ of the i^{th} gas in the j^{th} cluster is updated as follows:

$$S_{(i,j)}(t) = K \times H_j\,(t+1) \times P_{(i,j)}(t) \tag{3.2.4}$$

In this equation, $P_{(i,j)}$ denotes the partial pressure of gas i in cluster j, and K represents a constant.

- **Step6:** Position Update: The position $X_{i,j}$ is modified and updated based on the equation provided below:

$$\begin{aligned} X_{(i,j)}(t+1) = X_{(i,j)}(t) + F \times r_1 \times \gamma \times (X_{(i,best)}(t) - X_{(i,j)}(t)) \\ +F \times r_2 \times \alpha \times (S_{(i,j)}(t) \times X_{best}(t) - X_{(i,j)}(t)) \end{aligned} \tag{3.2.5}$$

$$\gamma = \beta \times exp(-\frac{F_{Lleaders}(t) + \epsilon}{F_{(j,i)}(t) + \epsilon}), \quad and \quad \epsilon = 0.05$$

where $X_{(i,j)}$ represents the position of the i^{th} gas particle within the j^{th} cluster, while r_1 and r_2 are two random numbers drawn from the interval $[0, 1]$. The term $X_{(i,best)}$ refers to the best gas particle i in cluster j, whereas X_{best} signifies the optimal gas in the entire swarm. Additionally, γ indicates the capacity of the i^{th} gas to interact with other gases in its cluster j, while α, which is set to 1, represents the influence of other gases on the i^{th} gas within group j. The variable β is a constant. The fitness values for X_{best} and $X_{(i,best)}$ are denoted as $F_{best}(t)$ and $F_{(i,j)}(t)$, respectively. Lastly, F serves as a flag that directs the search process, taking on values of either 1 or -1.

- **Step7:** Avoiding Local Optima: The HGSO algorithm utilizes a robust strategy to escape local optima by re-initializing N_w of the worst candidate solutions as follows:

$$N_w = N \times (rand\,(c_2 - c_1) + c_1), \quad c_1 = 0.1 \quad and \quad c_2 = 0.2 \tag{3.2.6}$$

Here, N represents the total number of search agents, while c_1 and c_2 are constants that determine the proportion of the least effective agents.

- **Step8:** Update the position of the least effective (worst) agents using:

$$G_{(i,j)} = G_{min(ij)} + r \times (G_{max(i,j)} - G_{min(i,j)}) \tag{3.2.7}$$

In this case, $G_{(i,j)}$ represents the position of gas i in cluster j, r is a random number within the interval [0, 1], and G_{max} and G_{min} define the problem's upper and lower bounds.

The HGSO (Henry Gas Solubility Optimization) algorithm was introduced as a global optimization method for continuous problems. It divides the population into groups with uniform gas coefficients, where gas movement is determined by solubility. The algorithm ranks agents by fitness, storing X_{best} and $X_{(i,best)}$ to update other agents' positions. Parameters $S_{(i,j)}$, γ, and F maintain a balance between exploration and exploitation, helping HGSO avoid local optima through controlled randomness. HGSO's potential for multi-objective optimization stems from two key factors: i) gas behavior influenced by solubility, utilizing rapid updates of Henry's law for improved movement, and ii) the algorithm's ability to save and use multiple best solutions throughout the optimization process.

This framework allows HGSO to effectively navigate complex solution spaces, making it a promising candidate for extension to multi-objective optimization tasks. In order to extend the basic ideas of HGSO to solve MOPs, our approach follows the same search behavior of HGSO with some modifications. let us summarize the utilized techniques in this study for multi-objective optimization before outlining the two versions of the Multi-Objective HGSO.

- Firstly, in MOHGSO, Pareto dominance is adopted as a criterion for comparing solutions to determine their relationships;
- An $K+1$ external-archives are incorporated into HGSO to retain the non-dominated solutions that have been identified and maintain the elitism concept during optimization;
- MOHGSO employs a density operator (Grid Mechanism/Crowding Distance) to identify the most and least populated segments of the Pareto front;
- Employing multi-leaders from multi-archives facilitates local region searches while also enabling a broader exploration of the search space;
- Multi-leaders help direct individuals to areas of the search space that are rarely explored, thereby enhancing the coverage capability of the MOHGSO algorithm;

Therefore, it is needed to an effective updating process for the $K+1$ external-archives and an effective leader selection strategy to produce a well-distributed Pareto front.

The mathematical steps of the proposed MOHGSO are presented as follows:

- Initial population is generated, the n gases are distributed uniformly in the entire search space.
- The population is segmented into equal clusters (groups), with n/N gases randomly distributed within each cluster.
- K initial local-archives and an initial global-archive are generated. The algorithm identifies the elitism concepts to save the best non-dominated solutions for the next iteration.
- Once the fixed number of iterations is defined, the procedure starts with update Henry coeficient H_j for each group using the formula given in 3.2.3. Then, the solubility $S_{(i,j)}$ of i^{th} gas in the j^{th} cluster is updated as in 3.2.4
- The position $X_{i,j}$ in iteration $t+1$ is updated as follow:

$$\begin{aligned} X_{(i,j)}(t+1) =& X_{(i,j)}(t) + F \times r_1 \times \gamma \times (X_{(i,leader)}(t) - X_{(i,j)}(t)) \\ &+ F \times r_2 \times \alpha \times (S_{(i,j)}(t) \times X_{Gleader}(t) - X_{(i,j)}(t)) \end{aligned} \tag{3.2.8}$$

$$\gamma = \beta \times exp(-\frac{F_{Lleaders}(t)+\epsilon}{F_{(j,i)}(t)+\epsilon}), \quad and \quad \epsilon = 0.05$$

  In this context, $X_{(i,leader)}$ refers to the local-leader within cluster j, while $X_{Gleader}$ denotes the global-leader chosen from the entire population. The term $F_{(j,i)}$ represents the total fitness accumulated from all gases in cluster j, whereas $F_{Lleaders}$ indicates the cumulative fitness of the K local-leaders. Additionally, F acts as a flag that determines the search direction, taking on values of either 1 or -1.
- The updated local-archives and the current global-archive are merged into a single set, which is then used to identify the globally non-dominated solutions, when $|global - archive| > T_{gmax}$, the outsized solutions are deleted from the most crowded segments

Algorithm3 outlines the pseudocode of the proposed MOHGSO algorithm.

Two new multi-objective versions of the recently proposed Henry Gas Solubility Optimization (HGSO) are called Multi-Objective Henry Gas Solubility Optimization (MOHGSO). The MOHGSO algorithm utilizes dominance rules and incorporates two types of archives to enhance the standard HGSO for addressing multi-objective problems (MOPs). One archive is designated for storing the Pareto solutions acquired during the optimization process, while the other external archives are intended for retaining the best local solutions from each cluster. The multi-objective extension of the HGSO was introduced proving its performance to address the MOPs by equipping them with both diversity mechanisms called crowding distance computation and the grid mechanism to improve

Algorithm 3: Pseudo-code of MOHGSO algorithm.

Input:
Initialize a population of n gases $X_i (i = 1, 2, ..., n)$
Evaluate each gas particle in X_i
Divide the population into equal K clusters
Initialize Local-Archives and Global-Archive
while $t < Max\_It$ **do**
    for *each search agent* **do**
        Update Henry's coefficient for each cluster by equation3.2.3
        Update solubility of each gas by equation 3.2.4
        $X_{(j.leader)}$ = Select a local-leader from the local-archive
        $X_{Gleader}$ = Select a global-leader from the global-archive
        Update the position of the current individual by equation 3.2.8
    end
    Update the local-archives
    if *(the local-archive is full)* **then**
        Run the grid mechanism/crowding distance to to prune the current
         archive members:
        Add the new solution to the local-archive
    end
    Global-archive = $\bigcup_{i=1}^{K+1} External - Archives$
    Update the Global-archive
    t = t+1.
end
Output: Global-archive(Pareto set).

the coverage of the search space and delete solutions of both global and local archives. An efficient leader selection strategy is adopted to select the local leaders and the global leader.

3.2.1 MOHGSO based on Grid Mechanism(MOHGSO1)

MOHGSO1 is a Pareto-based algorithm, MOHGSO1's performance primarily relies on two key components:

Multi-Archive System:

- Elite archive: Stores Pareto solutions found throughout the evolutionary process.
- K external archives: Retain the best local solutions for every cluster.

The benefits of using multi-archive are: ensures effective collection of best non-dominated solutions, employs grid mechanism to identify crowded segments in objective space, and improves the diversity of non-dominated solutions during $K+1$ archive updates.

Leader Selection Strategy:

- Select non-dominated solutions from less crowded segments as leaders.
- For each search agent: Local-leader selected from local-archive and Global-leader chosen from global-archive.
- Selection method: Roulette wheel approach applied to $K+1$ external archives

The probability of selecting a segment in the roulette wheel is determined by its crowdedness, with less crowded segments having higher selection probabilities. This approach balances exploration and exploitation in the search space.

$$P_i = \frac{c}{N_i} \tag{3.2.9}$$

In this equation, c represents a constant greater than 1, while K denotes the number of Pareto optimal solutions obtained within the i^{th} segment.

This dual-component structure allows MOHGSO1 to maintain a diverse set of high-quality solutions while effectively exploring the search space, contributing to its performance in multi-objective optimization tasks.

The proposed incorporates the grid mechanism and roulette wheel to select both local and global leaders, this process maintains the coverage of the algorithm by forcing the other solutions to update their position toward the areas of the objective space with the fewest individuals. The leader selection strategy can significantly improve both convergence and diversity because:

- The leader is chosen from the archive, allowing for a closer approximation to the true Pareto front.

- The leader is chosen from the archive with the fewest individuals using a roulette wheel selection, which helps direct the individuals towards infrequently explored regions of the solution space.
- Each unique solution is assigned two leaders, which enhances the algorithm's coverage capability.

3.2.2 MOHGSO based on Crowding Distance(MOHGSO2)

It should be noted here that the employed components are very similar to those of the suggested MOHGSO1. The mechanism of crowding distance is incorporated into the HGSO algorithm specifically in the selection and deletion method of an external archive of non-dominated solutions, where effective archiving and leader selection strategies that utilize crowding distance calculations are introduced. A widely adopted technique for maintaining diversity is the "crowding distance" method [33]. This approach evaluates the density of solutions in a particular area of the Pareto front by measuring the distance between the two nearest points (Figure 3.1). Larger distances indicate less crowded regions. Algorithm 4 outlines the crowding distance computation process.

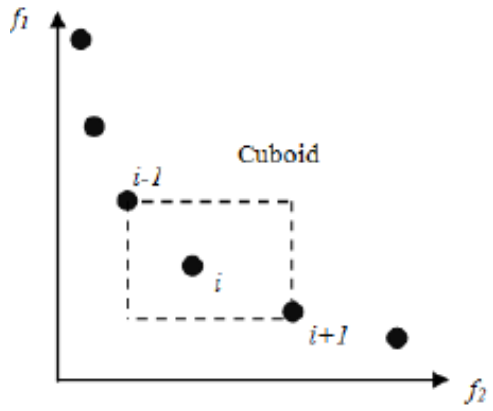


Figure 3.1: Crowding distance computation algorithm

In summary, the main ideas of the MOHGSO2 algorithm are listed as follows:

- Pareto dominance: Employed as a comparison criterion to determine relationships between solutions.
- Multi-archives: $K + 1$ external archives store non-dominated solutions, preserving elitism throughout the optimization process.
- Crowding distance: Applied at two stages to enhance solution diversity.
- Leader selection: Integrates crowding distance with random selection to choose two leaders per search agent.

Algorithm 4: Crowding distance computation algorithm

$I = |A|$
for $i \in A$ **do**
  $A[i]_{distance} = 0$
end
for *each objective* M **do**
  $A = \text{Sort}\ (A, M)$
  $A[1]_{distance} = A[I]_{distance} = \infty$
  for $i = 2$ *to* $(I - 1)$ **do**
    $A[i]_{distance} = A[i]_{distance} + (A[i + 1]_M - A[i - 1]_M)$
  end
end

The $K + 1$ external archives in MOHGSO2 facilitate a balance between convergence and diversity, allowing for an accurate approximation of the Pareto optimal front with well-distributed solutions. Additionally, crowding distance calculations during archive updates enhance the diversity of non-dominated solutions.

The leader selection mechanism aims to guide search agents towards promising regions, considering both convergence and diversity. This approach helps maintain a diverse set of high-quality solutions throughout the optimization process.

3.3 Results and discussion

This study employs MATLAB to evaluate MOHGSO's approximation capabilities, benchmarking it against two established swarm intelligence-based multi-objective optimization algorithms: MOGWO and MSSA. These algorithms are chosen for their proven performance in solving Multi-Objective Problems (MOPs) and their widespread use as benchmarking techniques.

Multi-Objective Grey Wolf Optimizer(MOGWO): derived from the GWO algorithm, key control parameters: A and c, utilizes grid mechanism for archive maintenance, employs roulette wheel for leader selection, and elects three leaders when updating each solution's position [92].

Multi-Objective Salp Swarm Algorithm (MSSA): based on the SSA algorithm, primary control parameter c_1, uses external archive to store best non-dominated solutions, ranks solutions and applies roulette wheel for archive maintenance, and selects one leader from the external archive [87].

The comparisons are carried out using tabular, and graphical presentations. To quantify the obtained results, two wide performance metrics are considered: i) Inverted Generational Distance (IGD) calculates the convergence–coverage of the algorithms [119], ii)

Spacing (Sp) quantifies the coverage of the algorithms [113]. Knowing that lower values for IGD and Sp metrics imply better performance.

- Inverted Generational Distance: The IGD metric assesses the convergence-coverage of an algorithm [119].

$$IGD = \frac{1}{n}\sqrt{\sum_{i=1}^{n} d_i^2}$$

where $d_i = min_j(|f_1^i(\vec{x}) - |f_1^j(\vec{x})| + |f_2^i(\vec{x}) - |f_2^j(\vec{x})|\ i; j = 1; 2; ...; M$

- Spacing: The Sp metric evaluates the coverage provided by an algorithm [113].

$$Sp = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(d_i - \bar{d})^2}$$

where $\bar{d}$: is the average of d_i.

Here, we take into account the controlling parameters of the competitor algorithms, most parameters have been adopted as they are described by their developers. The parameters of the competitor algorithms are listed in 3.1.

Table 3.1: Parameters values of competitor algorithms.

Algorithm	**Parameters**
MOGWO	$\alpha = 0.1$: grid inflation $\gamma = 2$: archive member selection pressure $nGrid = 10$: grids number per each dimension $\beta = 4$: leader selection pressure parameter
MSSA	$c_1 = 2 * exp(-(4 * iter/max_iter)^2)$:
MOHGSO	$\alpha = 1, \beta = 1$ and $K = 1$, Cluster number: $K = 5$ The constants l_1, l_2, and l_3 were assigned the values of $5E-03$, 10 and $E-02$, respectively. $\epsilon = 0.05$

The efficiency of the competitor algorithms is validated via ZDT-series [163], DTLZ-series [34] with diverse Pareto optimal fronts, and the CEC2009 test suite [153] which is considered one of the most challenging benchmark sets in the literature and includes highly biased, rotated, shifted, hybridized, and composite test functions. In all cases, the statistical results, including best, worst, median, mean, and standard deviation (std) of IGD, and Spacing(Sp) metrics of the different algorithms are obtained after 31 runs where the minimum average (mean) of each metric are marked in boldface, and the qualitative results are displayed corresponding to the lowest IGD value obtained from the 31 independent runs.

3.3.1 Results on ZDT-series

Tables 3.2 and 3.3 display the simulation results for IGD and Sp metrics, respectively, across various algorithms.

Table 3.2 illustrates that MOHGSO2 excels in all statistical metrics for IGD on three test functions, demonstrating superior solution stability compared to competitors. MOHGSO2 achieves a close second place on ZDT2 and ZDT4, following MOHGSO1, with highly competitive results.

MOHGSO1 and MOHGSO2 show comparable performance across most bi-objective benchmark problems, indicating strong convergence to the true Pareto front.

These results highlight the effectiveness of both MOHGSO variants in addressing bi-objective optimization problems, with MOHGSO2 showing particular strength in solution stability and overall performance.

Table 3.2: IGD values for Algorithms on ZDT-Series Problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
ZDT1	Best	5.89E-04	3.15E-03	3.70E-04	**9.62E-05**
	Worst	6.10E-03	7.20E-03	5.99E-04	**1.41E-04**
	Mean	1.50E-03	4.73E-03	5.09E-04	**1.10E-04**
	Median	1.29E-03	4.46E-03	5.13E-04	**1.08E-04**
	Std	9.63E-04	1.06E-03	7.39E-05	**8.69E-06**
ZDT2	Best	7.93E-04	3.65E-03	4.10E-04	**1.01E-04**
	Worst	2.31E-02	1.16E-02	**7.39E-04**	2.31E-02
	Mean	6.45E-03	5.30E-03	**5.43E-04**	1.65E-03
	Median	1.59E-03	5.10E-03	5.06E-04	**1.15E-04**
	Std	9.16E-03	1.43E-03	**1.11E-04**	5.83E-03
ZDT3	Best	3.06E-04	4.50E-03	5.83E-04	**1.12E-04**
	Worst	3.26E-03	1.16E-02	1.22E-03	**1.81E-04**
	Mean	9.66E-04	7.18E-03	9.10E-04	**1.29E-04**
	Median	8.02E-04	6.93E-03	8.72E-04	**1.27E-04**
	Std	5.93E-04	1.78E-03	1.82E-04	**1.58E-05**
ZDT4	Best	2.89E-02	6.93E-02	3.89E-04	**9.79E-05**
	Worst	7.19E-01	5.01E-01	**6.48E-04**	2.89E-02
	Mean	2.88E-01	1.94E-01	**4.78E-04**	1.07E-03
	Median	2.34E-01	1.74E-01	4.71E-04	**1.10E-04**
	Std	2.09E-01	9.52E-02	**7.03E-05**	5.25E-03
ZDT6	Best	2.88E-04	6.68E-04	3.55E-04	**8.35E-05**
	Worst	2.99E-03	4.21E-03	6.04E-04	**1.06E-04**
	Mean	1.45E-03	2.05E-03	4.77E-04	**9.32E-05**
	Median	1.87E-03	1.74E-03	4.45E-04	**9.32E-05**
	Std	8.96E-04	1.00E-03	1.10E-04	**4.95E-06**

Table 3.3 illustrates that MOHGSO2 achieves best results across all statistical measures of Sp-metric for ZDT1, ZDT2, ZDT3, and ZDT6, and ranks second on ZDT4, closely following MSSA with highly competitive results. MOHGSO1 shows reasonable results,

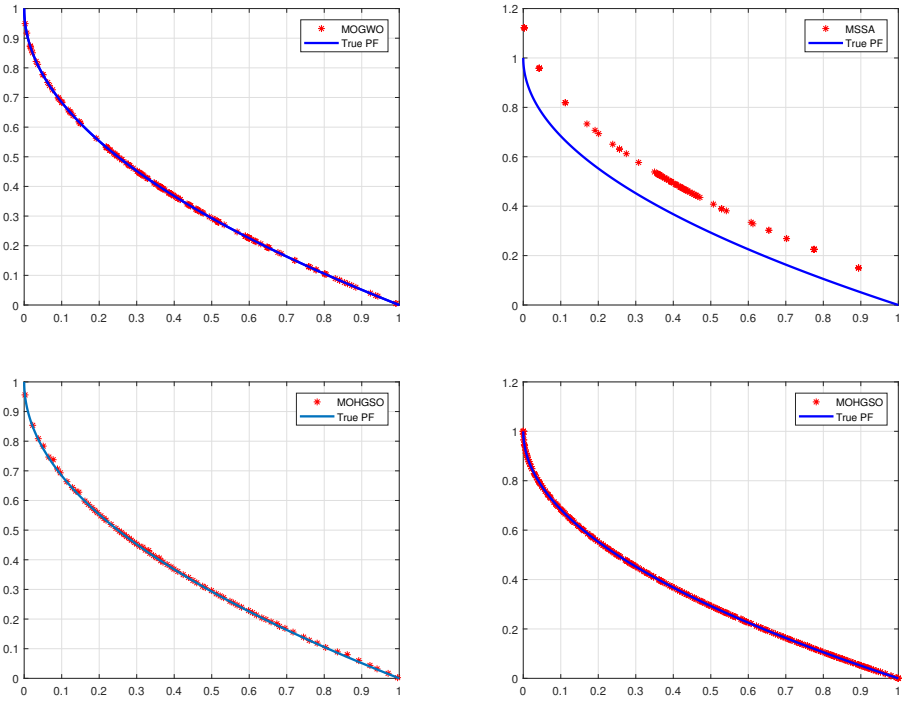


Figure 3.2: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for ZDT1.

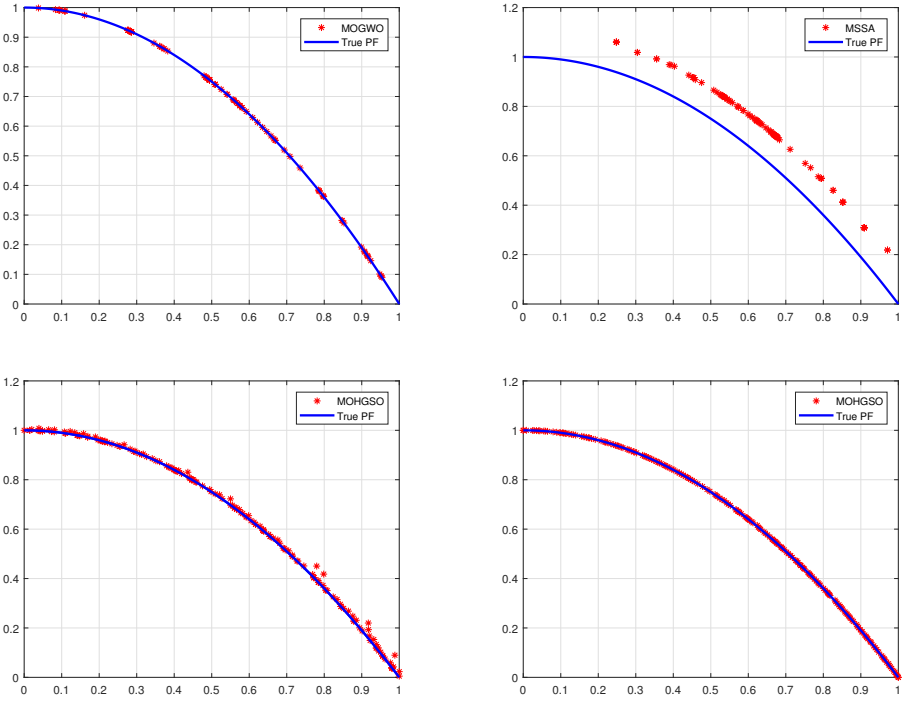


Figure 3.3: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for ZDT2.

though generally underperforming compared to MOHGSO2. MOGWO and MSSA exhibit similar behavior across all test functions.

This analysis demonstrates MOHGSO2's superior performance in maintaining solution diversity across multiple ZDT problems, while also highlighting areas for potential improvement in MOHGSO1. The consistent performance of MOGWO and MSSA provides a stable benchmark for comparison.

The obtained Pareto optimal solutions of the four algorithms on all bi-objective test functions are presented in Figures 3.2, 3.3, 3.4, 3.5 and 3.6.

Table 3.3: Sp Values for Algorithms on ZDT-Series Problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
ZDT1	Best	7.71E-03	5.66E-03	1.04E-02	**3.25E-03**
	Worst	3.90E-02	2.49E-02	1.51E-02	**4.16E-03**
	Mean	1.57E-02	1.29E-02	1.19E-02	**3.70E-03**
	Median	1.36E-02	1.19E-02	1.15E-02	**3.64E-03**
	Std	6.63E-03	5.30E-03	1.47E-03	**2.32E-04**
ZDT2	Best	**0.00E+00**	3.65E-03	1.19E-02	**0.00E+00**
	Worst	2.09E-02	2.84E-02	1.76E-02	**4.59E-03**
	Mean	1.02E-02	1.16E-02	1.36E-02	**3.66E-03**
	Median	1.10E-02	1.09E-02	1.34E-02	**3.90E-03**
	Std	6.28E-03	5.06E-03	1.64E-03	**1.02E-03**
ZDT3	Best	6.19E-03	4.11E-03	7.07E-03	**3.33E-03**
	Worst	3.23E-02	3.03E-02	2.34E-02	**4.57E-03**
	Mean	1.80E-02	1.30E-02	1.40E-02	**3.96E-03**
	Median	1.80E-02	1.20E-02	1.24E-02	**3.99E-03**
	Std	6.72E-03	6.28E-03	5.39E-03	**2.82E-04**
ZDT4	Best	**0.00E+00**	7.76E-05	8.90E-03	**0.00E+00**
	Worst	6.60E+00	2.66E-02	1.64E-02	**5.12E-03**
	Mean	1.32E+00	**2.85E-03**	1.19E-02	3.63E-03
	Median	8.19E-01	**1.24E-03**	1.11E-02	3.71E-03
	Std	1.63E+00	5.02E-03	2.44E-03	**7.84E-04**
ZDT6	Best	5.70E-03	7.19E-03	8.95E-03	**2.58E-03**
	Worst	1.05E-01	1.65E-01	1.12E-02	**3.86E-03**
	Mean	1.76E-02	2.06E-02	1.01E-02	**3.17E-03**
	Median	9.04E-03	1.40E-02	9.85E-03	**3.17E-03**
	Std	2.12E-02	2.77E-02	1.00E-03	**3.26E-04**

On ZDT1 (Convex Pareto Front), Figure 3.2 illustrates algorithms performance on ZDT1. MOHGSO2, employing crowding distance, demonstrates better convergence and significant diversity across both objectives. MOGWO and MOHGSO1 (grid mechanism-based) show similar Pareto fronts, while MSSA exhibits weaker diversity and convergence to the true Pareto front.

On ZDT2 (Non-convex Front), as seen in Figure 3.3, MOHGSO1 achieves a notably superior Pareto front compared to MOGWO and MSSA algorithms on this non-convex problem.

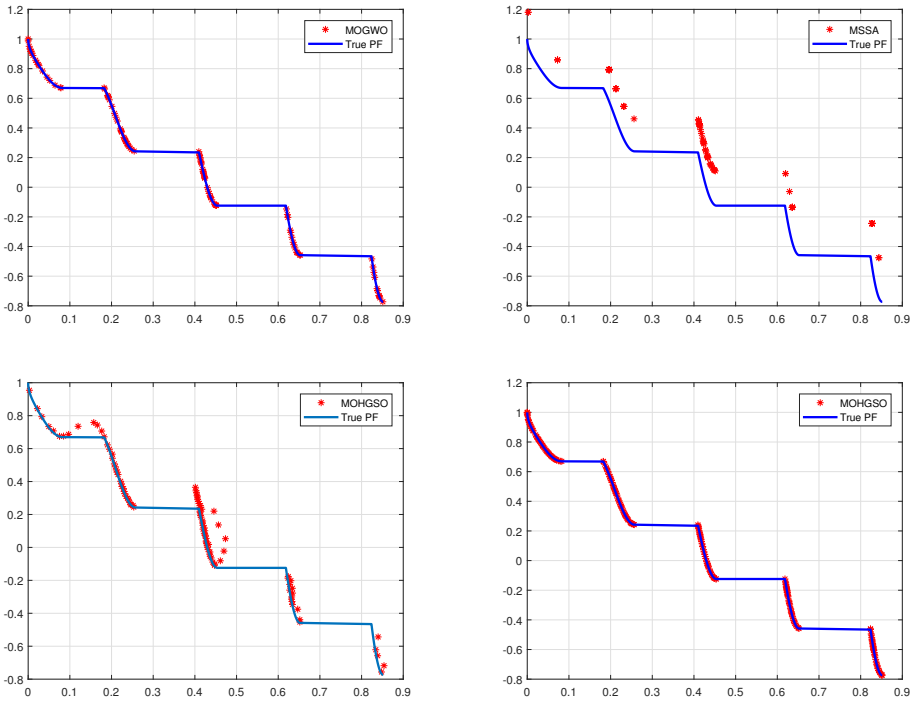


Figure 3.4: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for ZDT3.

On ZDT3 (Discontinuous Front), Figure 3.4 showcases MOHGSO2's ability to effectively identify the five separate areas of the true Pareto optimal front exhibiting high diversity. This performance underscores the algorithm's proficiency in addressing discontinuous search spaces.

These results highlight the adaptability of the MOHGSO algorithms across diverse Pareto front geometries, with each showing strengths in different problem types.

On ZDT4 (Multiple Local Pareto Fronts), Figure 3.5 demonstrates the performance on ZDT4, which features 2^{21} local Pareto fronts. MOHGSO1 and MOHGSO2 both achieve a close approximation to the true Pareto front with well-distributed solutions, this indicates robust capability in overcoming local optima challenges. In contrast, MOGWO and MSSA exhibit poor convergence and coverage on this complex landscape.

Despite ZDT6's challenging non-uniform search space, Figure 3.6 reveals that MOHGSO2 maintains superior diversity compared to competing algorithms, this showcases the algorithm's adaptability to varied problem characteristics.

These results highlight the MOHGSO algorithms' effectiveness in addressing complex multi-objective optimization problems, particularly those involving multiple local optima or non-uniform search spaces.

According to the obtained Pareto optimal solutions of the competitor algorithms, the following observations can be made:

- Multi-objective HGSO versions perform well in solving MOPs with convex and non-

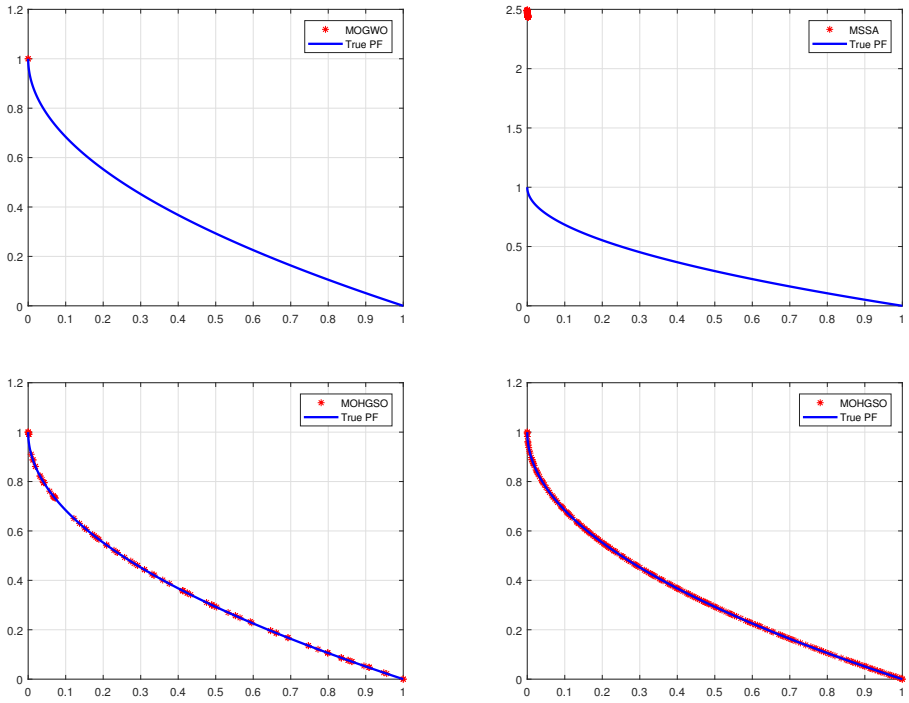


Figure 3.5: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for ZDT4.

convex Pareto front;

- The ability of Multi-objective HGSO versions to identify the separate areas of the true Pareto front;
- Multi-objective HGSO versions have a strong capability to navigate past the challenges of local optima;
- The good exploration capability of Multi-objective HGSO versions in the non-uniform search space.

3.3.2 Results on DTLZ-series

Tables 3.4 and 3.5 display the IGD and Sp metric results for DTLZ test functions, revealing varied optimizer performance across different functions. MOHGSO2 demonstrates superior performance across all statistical measures for DTLZ2, DTLZ5, DTLZ6, and DTLZ7.

For DTLZ1, MOHGSO2 outperforms others in mean IGD value with competitive standard deviation. MSSA achieves the best median and convergence mean.

For DTLZ3, MOGWO leads, followed by MOHGSO1. MOHGSO2 ranks third with higher stability.

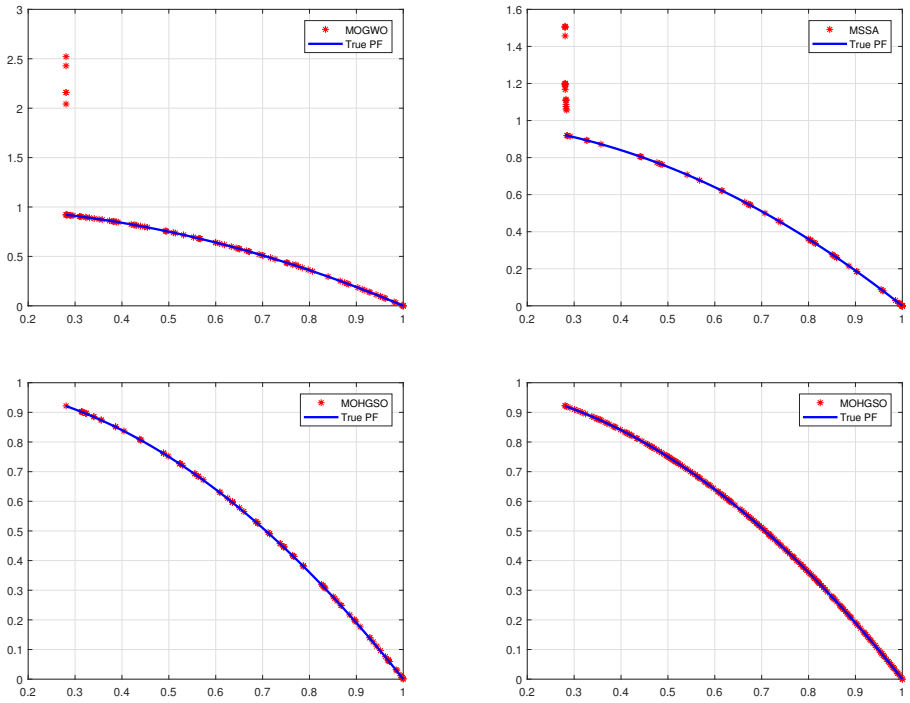


Figure 3.6: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for ZDT6.

For DTLZ4 (highly non-uniform Pareto optimal solution distribution), MOGWO slightly edges out MOHGSO2, both showing similar convergence and superior Pareto front approximation.

MOHGSO1 consistently ranks second in mean IGD value for DTLZ3, DTLZ6, and DTLZ7, following MOGWO.

This analysis highlights the robust performance of MOHGSO algorithms across various DTLZ test functions, often matching or surpassing established algorithms like MOGWO and MSSA.

Table 3.5 presents Sp values, indicating MOHGSO2's superior diversity performance compared to MOHGSO1, MOGWO, and MSSA.

MOHGSO2 achieves best mean Sp values for DTLZ1, DTLZ3, DTLZ4, DTLZ5, and DTLZ6. MOGWO leads in mean Sp values for DTLZ2 and DTLZ7

These quantitative results demonstrate the effectiveness of MOHGSO2's crowding distance-based approach in maintaining solution diversity.

Qualitative analysis through Pareto front visualizations reveals: that MSSA struggles with the linear Pareto front (DTLZ1), while other algorithms show better performance (see Figure 3.7). On DTLZ2, MOHGSO2 exhibits superior distribution and stability on the spherical Pareto front (see Figure 3.8).

These visual representations corroborate the quantitative findings, highlighting MOHGSO2's robust performance across various Pareto front geometries.

Table 3.4: IGD Values for Algorithms on DTLZ-series Problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
DTLZ1	Best	3.49E-02	**6.01E-03**	1.40E-01	8.31E-02
	Worst	**2.01E-01**	2.70E-01	2.70E-01	2.06E-01
	Mean	1.25E-01	**6.39E-02**	2.09E-01	1.43E-01
	Median	1.26E-01	**2.20E-02**	2.16E-01	1.34E-01
	Std	3.96E-02	8.12E-02	3.91E-02	**3.30E-02**
DTLZ2	Best	6.69E-03	5.28E-03	5.74E-03	**7.57E-04**
	Worst	1.03E-02	8.21E-03	9.35E-03	**8.24E-04**
	Mean	7.88E-03	7.32E-03	7.36E-03	**7.82E-04**
	Median	7.53E-03	7.45E-03	7.45E-03	**7.79E-04**
	Std	8.98E-04	6.18E-04	1.35E-03	**1.73E-05**
DTLZ3	Best	1.70E+00	**1.01E-01**	1.31E+00	1.75E+00
	Worst	2.97E+00	**2.37E+00**	2.69E+00	2.64E+00
	Mean	2.78E+00	**1.56E+00**	1.59E+00	2.41E+00
	Median	2.84E+00	**1.90E+00**	2.14E+00	2.47E+00
	Std	2.22E-01	8.08E-01	**1.58E-01**	2.13E-01
DTLZ4	Best	1.76E-03	4.57E-03	9.14E-03	**1.46E-03**
	Worst	**2.80E-03**	1.02E-02	1.18E-02	3.73E-03
	Mean	**2.20E-03**	6.80E-03	1.02E-02	2.33E-03
	Median	**2.19E-03**	6.87E-03	9.87E-03	2.26E-03
	Std	**2.62E-04**	1.27E-03	9.49E-04	5.70E-04
DTLZ5	Best	8.71E-04	6.53E-04	5.11E-03	**5.06E-05**
	Worst	2.40E-03	3.72E-03	6.72E-03	**6.29E-05**
	Mean	1.50E-03	1.55E-03	6.14E-03	**5.69E-05**
	Median	1.53E-03	1.43E-03	6.19E-03	**5.71E-05**
	Std	4.23E-04	6.05E-04	6.21E-04	**3.24E-06**
DTLZ6	Best	7.58E-04	6.17E-04	2.46E-04	**4.88E-05**
	Worst	7.31E-03	5.75E-03	2.99E-04	**6.31E-05**
	Mean	2.19E-03	1.84E-03	2.73E-04	**5.56E-05**
	Median	1.78E-03	1.29E-03	2.73E-04	**2.59E-04**
	Std	1.44E-03	1.26E-03	3.78E-05	**3.41E-06**
DTLZ7	Best	2.71E-03	6.72E-03	**1.69E-03**	**1.69E-03**
	Worst	9.95E-03	2.52E-02	4.77E-03	**3.77E-03**
	Mean	5.58E-03	1.12E-02	2.77E-03	**2.29E-03**
	Median	5.70E-03	1.08E-02	2.15E-03	**2.05E-03**
	Std	2.07E-03	3.76E-03	6.95E-04	**6.74E-04**

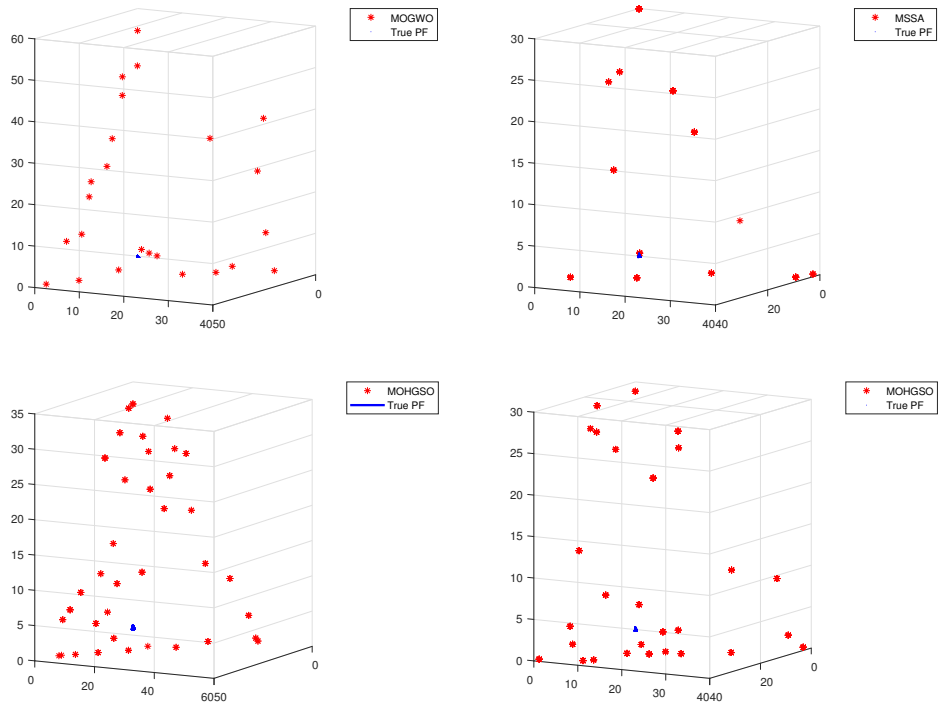


Figure 3.7: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ1.

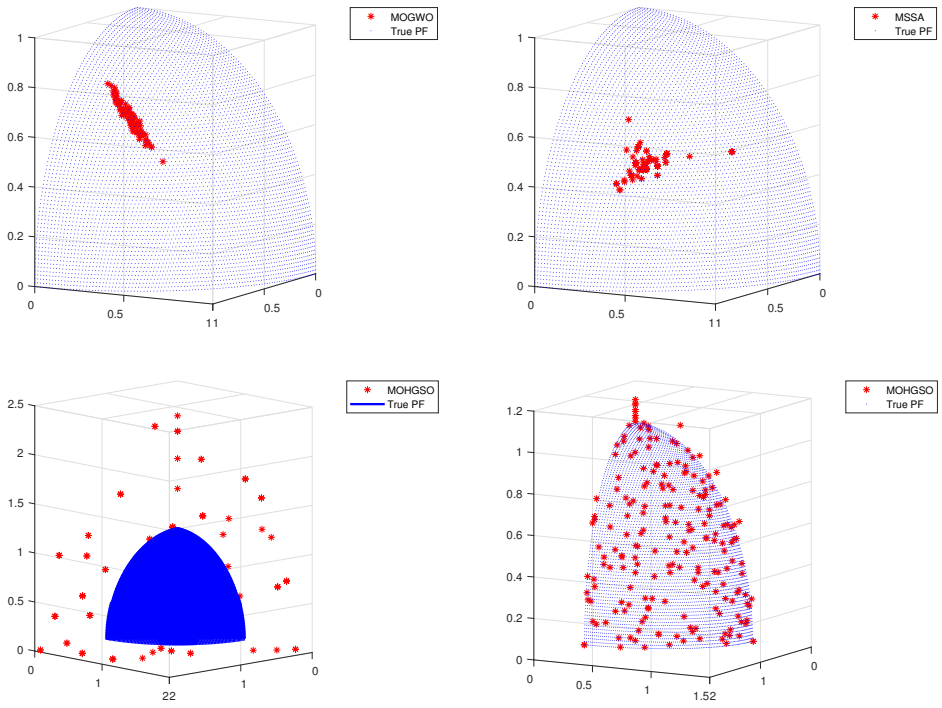


Figure 3.8: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ2.

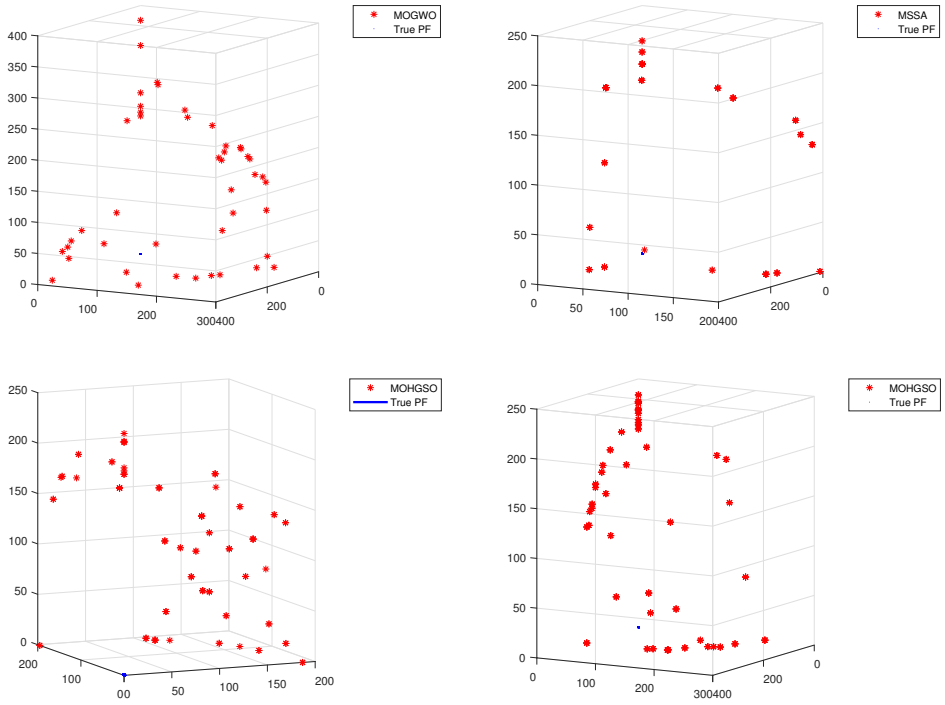


Figure 3.9: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ3.

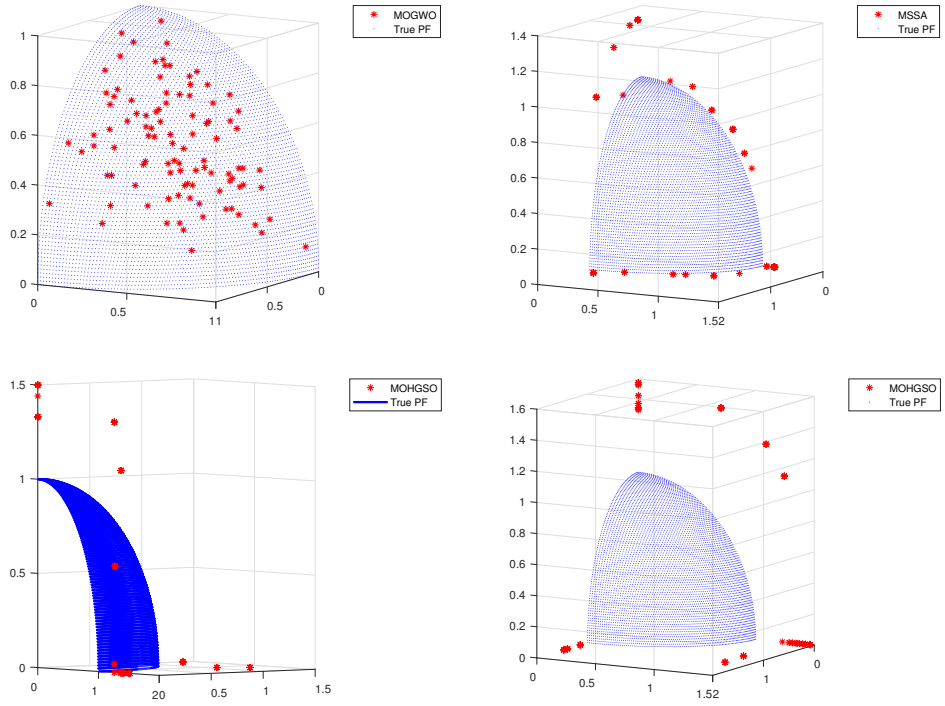


Figure 3.10: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ4.

Table 3.5: Sp Values for Algorithms on DTLZ-series Problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
DTLZ1	Best	2.28E+00	8.42E-06	7.52E-01	**0.00E+00**
	Worst	5.08E+00	2.45E+00	2.38E+00	**5.55E-01**
	Mean	3.45E+00	5.42E-01	1.43E+00	**1.85E-02**
	Median	3.44E+00	6.81E-05	1.44E+00	**0.00E+00**
	Std	6.46E-01	7.96E-01	4.93E-01	**1.01E-01**
DTLZ2	Best	2.38E-03	**1.72E-03**	4.43E-02	3.66E-02
	Worst	**1.31E-02**	2.72E-02	1.13E-01	4.88E-02
	Mean	**4.96E-03**	7.59E-03	9.04E-02	4.03E-02
	Median	**4.56E-03**	6.62E-03	9.78E-02	4.02E-02
	Std	**2.36E-03**	5.39E-03	2.05E-02	2.57E-03
DTLZ3	Best	1.02E+01	2.68E-04	4.94E+00	**0.00E+00**
	Worst	2.78E+01	1.80E+01	1.10E+01	**1.02E+00**
	Mean	1.75E+01	3.04E+00	7.60E+00	**3.45E-02**
	Median	1.76E+01	1.92E-02	6.98E+00	**0.00E+00**
	Std	3.83E+00	4.86E+00	2.32E+00	**1.87E-01**
DTLZ4	Best	3.41E-02	3.59E-05	**0.00E+00**	**0.00E+00**
	Worst	7.96E-02	1.88E-01	1.18E-02	**1.15E-02**
	Mean	5.29E-02	2.65E-02	3.73E-03	**3.33E-03**
	Median	5.13E-02	1.78E-02	2.82E-03	**2.42E-03**
	Std	1.14E-02	3.29E-02	3.93E-03	**3.63E-03**
DTLZ5	Best	8.74E-03	**3.10E-03**	2.23E-02	3.71E-03
	Worst	2.77E-02	3.58E-02	6.78E-02	**5.73E-03**
	Mean	1.70E-02	1.43E-02	5.37E-02	**4.59E-03**
	Median	1.62E-02	1.20E-02	5.96E-02	**4.62E-03**
	Std	4.62E-03	7.22E-03	1.80E-02	**4.34E-04**
DTLZ6	Best	8.23E-02	4.33E-03	1.34E-02	**3.57E-03**
	Worst	1.09E+00	9.07E-02	1.44E-02	**5.02E-03**
	Mean	2.96E-01	1.78E-02	1.39E-02	**4.48E-03**
	Median	2.35E-01	1.31E-02	1.39E-02	**4.54E-03**
	Std	2.30E-01	1.71E-02	7.11E-04	**3.23E-04**
DTLZ7	Best	1.25E-02	**7.41E-03**	6.29E-02	4.05E-02
	Worst	1.29E-01	1.42E-01	1.24E-01	**7.98E-02**
	Mean	**4.22E-02**	4.56E-02	9.21E-02	6.35E-02
	Median	**3.58E-02**	3.88E-02	8.95E-02	6.49E-02
	Std	2.86E-02	2.66E-02	2.22E-02	**8.78E-03**

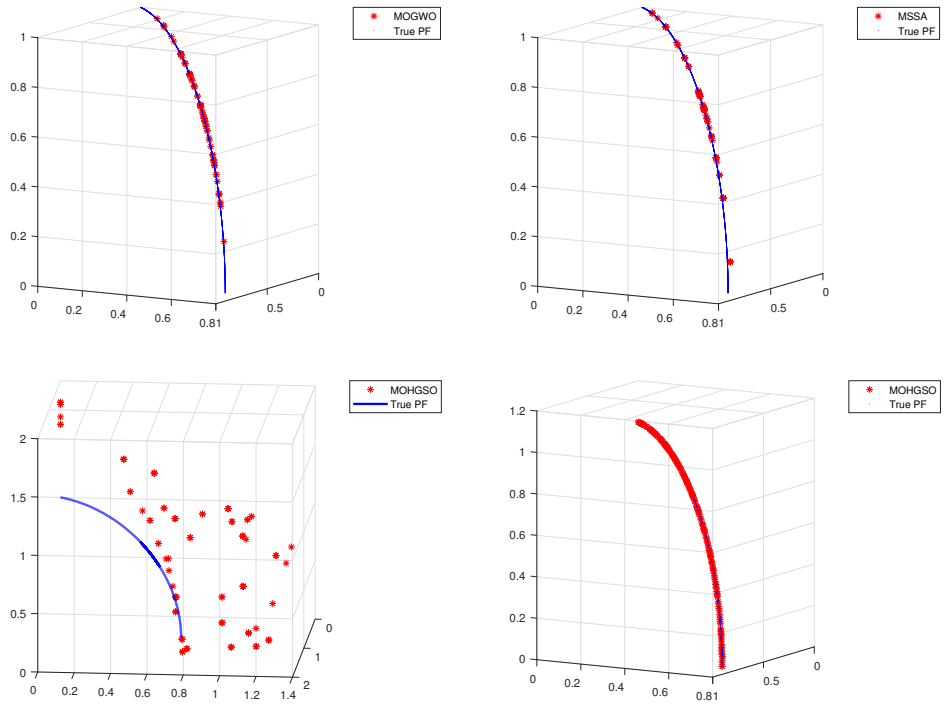


Figure 3.11: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ5.

Figure 3.9 demonstrates the superior convergence of MOHGSO1 and MOHGSO2 towards the true Pareto front. Despite DTLZ4's highly non-uniform Pareto optimal solution distribution, Figure 3.10 illustrates well-distributed fronts obtained by both MOHGSO variants.

On the discontinuous Pareto front of DTLZ5, Figure 3.11 shows MOHGSO2's markedly improved performance. Figure 3.12 further highlights the strong convergence and coverage of both MOHGSO algorithms for DTLZ6.

All algorithms exhibit competitive performance on the challenging DTLZ7 problem. However, MOHGSO2 achieves the closest approximation to the true Pareto front. Visual results confirm the algorithms' distribution quality.

For three-objective problems, Multi-objective HGSO versions demonstrate a slight edge over competitors, while maintaining highly competitive results overall.

3.3.3 Results on UF-series

Tables 3.6, 3.7 present the results of the simulations of the algorithms for IGD and Sp metrics, respectively. Table 3.6 demonstrates that the MOHGSO2 algorithm delivers the most favorable results on the majority of statistical metrics for UF1, UF2, and UF7, the obtained results in terms of the mean for the IGD metric of MOHGSO1 and MOHGSO2 are almost similar for most of the bi-objective benchmark problems (UF1, UF4, UF5, UF6

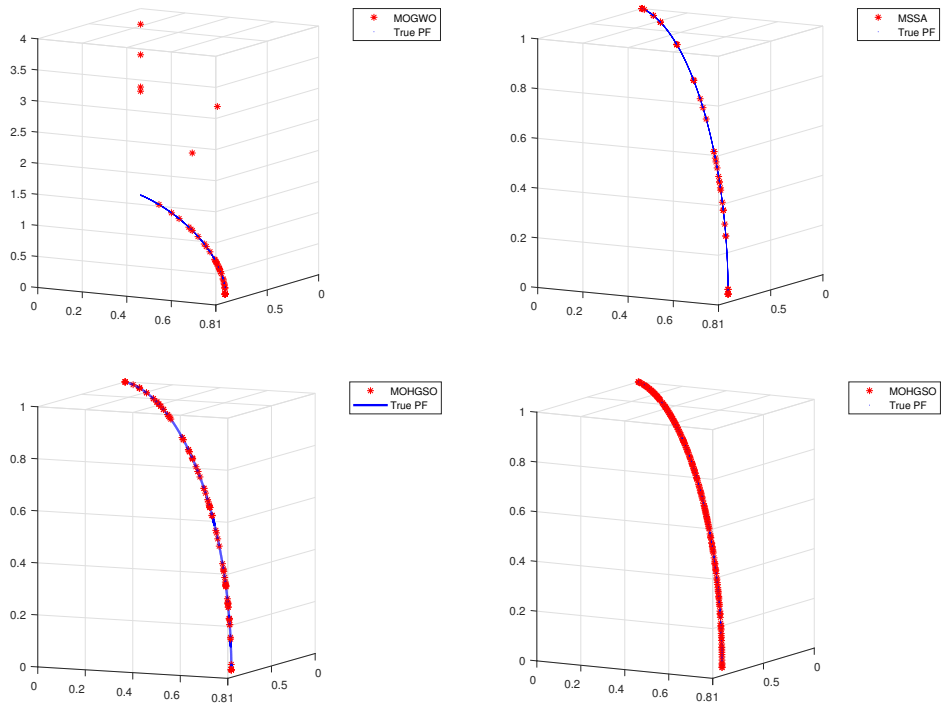


Figure 3.12: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ6.

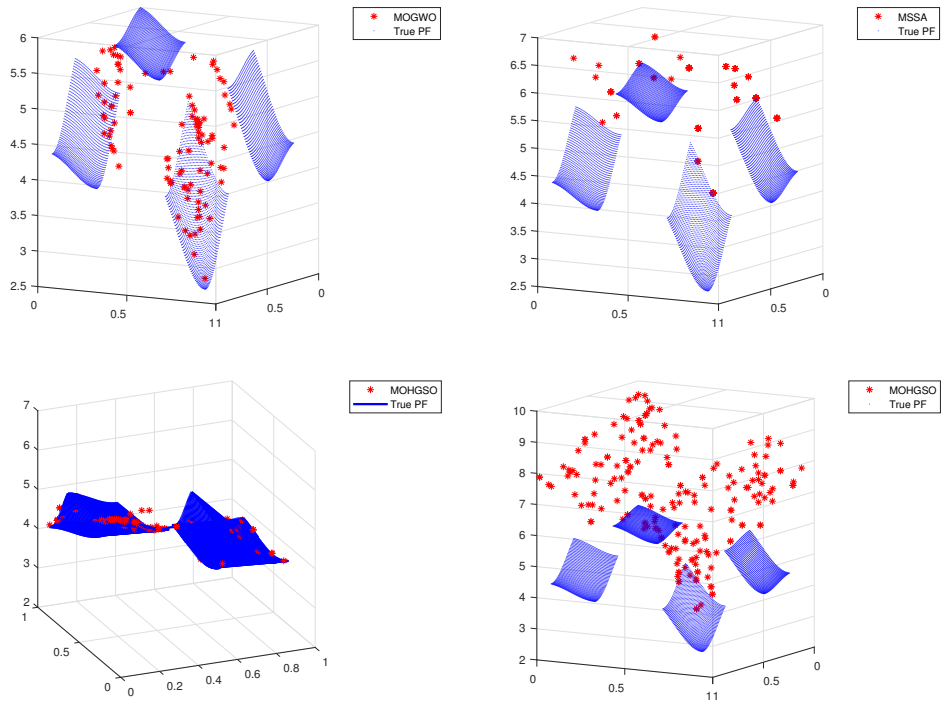


Figure 3.13: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for DTLZ7.

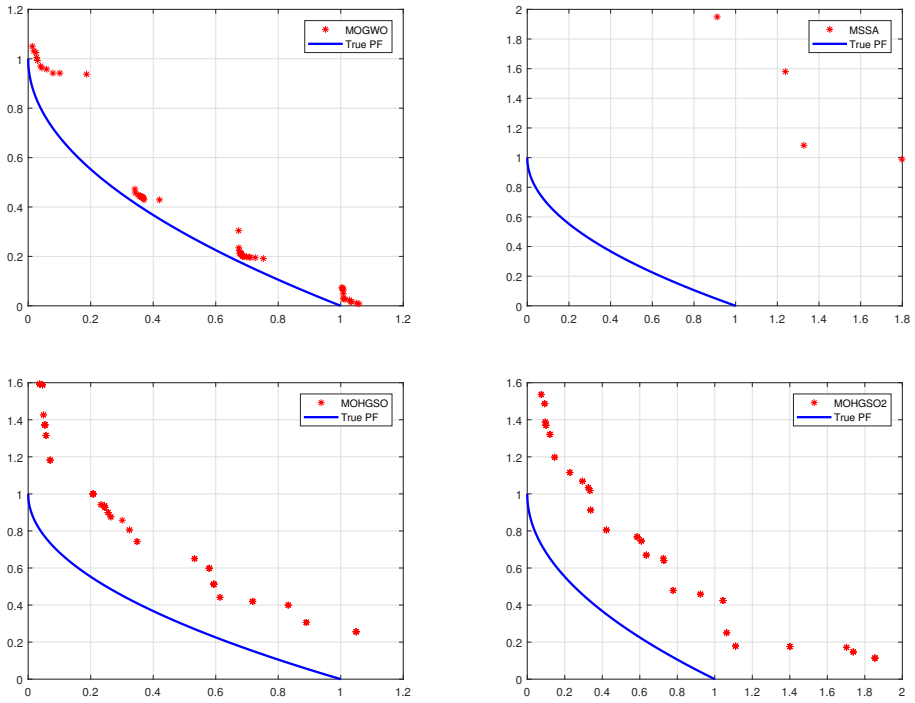


Figure 3.14: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF1.

and UF7).

On the other hand, the best mean is achieved by the suggested MOHGSO1 in UF3 and UF7 and ranks second after MOGWO in UF6, where the suggested MOHGSO2 ranks second in UF7 with competitive results. Also, it is interesting to record the remarks regarding the obtained results for the IGD metric in UF5 and UF6 test functions, where MOGWO achieves the best results.

Indeed, the results in Table 3.7 show that the MOHGSO2 that is based on crowding distance achieved the lowest mean Sp-metric value across all seven test functions, indicating that the proposed algorithm provides good coverage. The proposed MOHGSO1 which is based on crowding distance ranks second with competitive results in UF2, UF3, and UF5.

The obtained Pareto optimal solutions of the four algorithms on all bi-objective test functions are presented in Figures 3.14, 3.15, 3.16, 3.17, 3.18, 3.19 and 3.20. In most bi-objective test functions, it may be seen that the convergence of MOGWO is slowly better than the suggested algorithms with discontinuities on the obtained Pareto front. The diversity of the obtained Pareto optimal sets of both MOHGSO1 and MOHGSO2 is broader than both MOGWO and MSSA without discontinuities in most test functions. According to all obtained Pareto fronts for the MSSA algorithm, this algorithm could not get Pareto optimal solutions close to the true front.

UF8, UF9, and UF10 are three-objective problems, making them more complex than

Table 3.6: IGD values for Algorithms on UF1 to UF7 Problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
UF1	Best	3.91E-02	3.72E-02	**7.46E-03**	1.02E-02
	Worst	4.35E-02	4.52E-02	2.49E-02	**2.09E-02**
	Mean	4.08E-02	4.21E-02	1.63E-02	**1.56E-02**
	Median	4.02E-02	4.27E-02	1.93E-02	**1.60E-02**
	Std	**1.89E-03**	3.43E-03	7.43E-03	4.21E-03
UF2	Best	2.12E-02	1.88E-02	3.43E-03	**2.62E-03**
	Worst	2.89E-02	2.32E-02	4.46E-03	**3.41E-03**
	Mean	2.46E-02	2.10E-02	3.94E-03	**2.94E-03**
	Median	2.49E-02	2.10E-02	3.99E-03	**2.87E-03**
	Std	2.89E-03	1.55E-03	3.96E-04	**2.90E-04**
UF3	Best	1.08E-02	3.38E-02	**3.43E-03**	1.11E-02
	Worst	1.15E-02	3.94E-02	**4.46E-03**	1.19E-02
	Mean	1.11E-02	3.75E-02	**3.94E-03**	1.16E-02
	Median	1.11E-02	3.76E-02	**3.99E-03**	1.18E-02
	Std	**2.44E-04**	2.26E-03	3.96E-04	2.83E-04
UF4	Best	2.91E-03	5.91E-03	2.71E-03	**2.61E-03**
	Worst	3.55E-03	6.45E-03	3.02E-03	**3.05E-03**
	Mean	3.14E-03	6.22E-03	2.88E-03	**2.85E-03**
	Median	2.95E-03	6.27E-03	2.87E-03	**2.84E-03**
	Std	3.00E-04	2.01E-04	**1.38E-04**	1.73E-04
UF5	Best	**1.49E-01**	1.18E+00	7.61E-01	7.51E-01
	Worst	**3.06E-01**	1.30E+00	8.32E-01	8.17E-01
	Mean	**1.95E-01**	1.24E+00	7.94E-01	7.78E-01
	Median	**1.75E-01**	1.24E+00	7.87E-01	7.61E-01
	Std	6.28E-02	5.38E-02	3.36E-02	**2.99E-02**
UF6	Best	**1.34E-02**	1.34E-01	2.72E-02	4.61E-02
	Worst	**1.98E-02**	2.02E-01	1.03E-01	8.16E-02
	Mean	**1.54E-02**	1.76E-01	5.75E-02	5.78E-02
	Median	**1.47E-02**	1.81E-01	5.83E-02	5.22E-02
	Std	**2.47E-03**	2.74E-02	3.15E-02	1.46E-02
UF7	Best	2.68E-02	3.99E-02	**6.68E-03**	1.06E-02
	Worst	3.50E-02	5.18E-02	2.19E-02	**2.10E-02**
	Mean	2.98E-02	4.57E-02	**1.48E-02**	1.58E-02
	Median	2.98E-02	4.63E-02	1.58E-02	**1.53E-02**
	Std	**3.20E-03**	4.40E-03	7.39E-03	3.93E-03

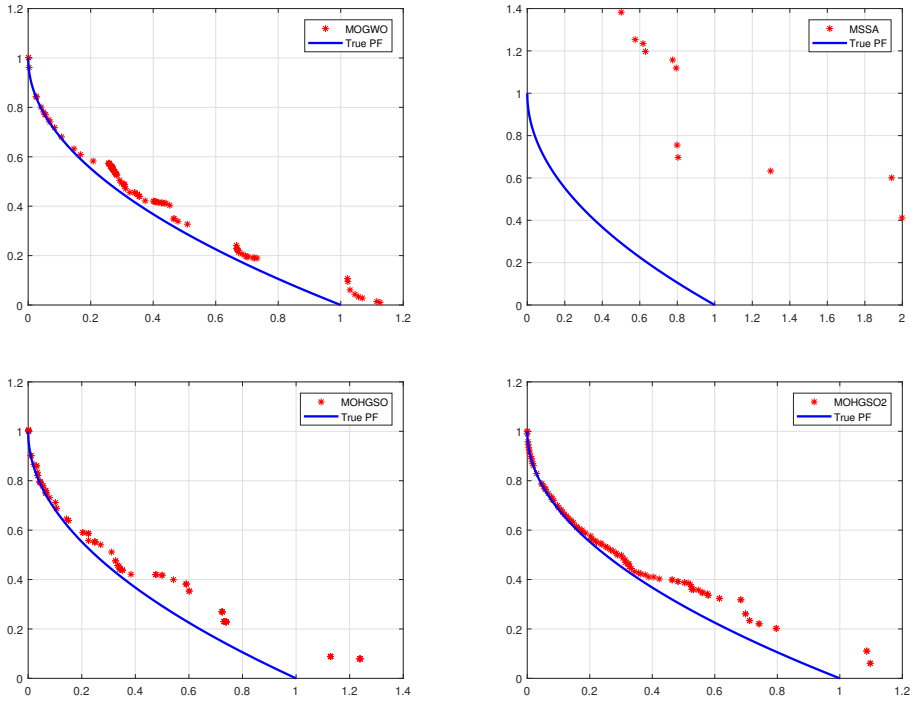


Figure 3.15: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF2.

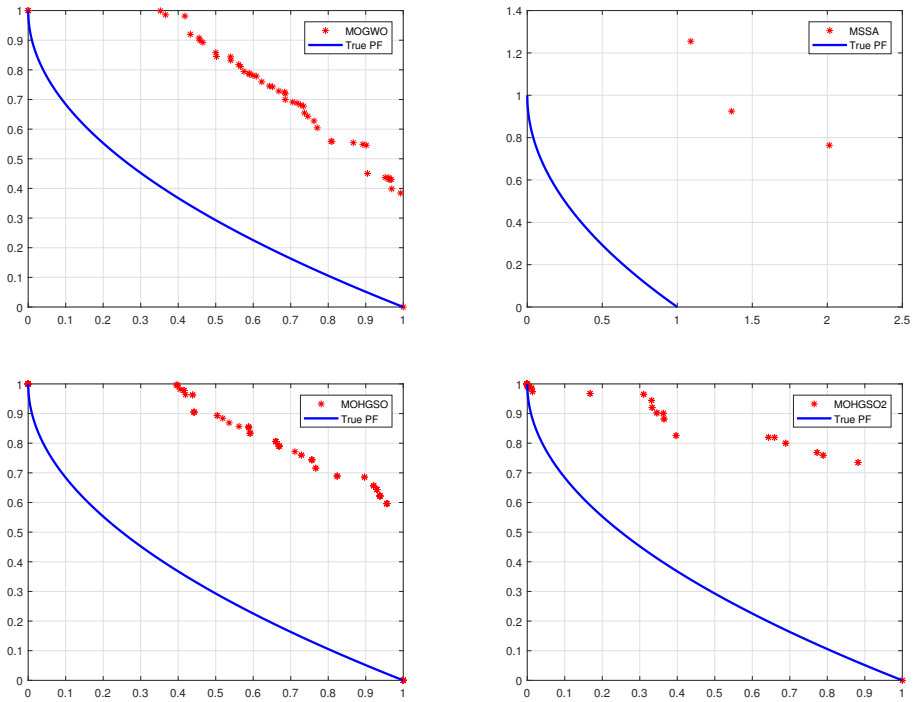


Figure 3.16: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF3.

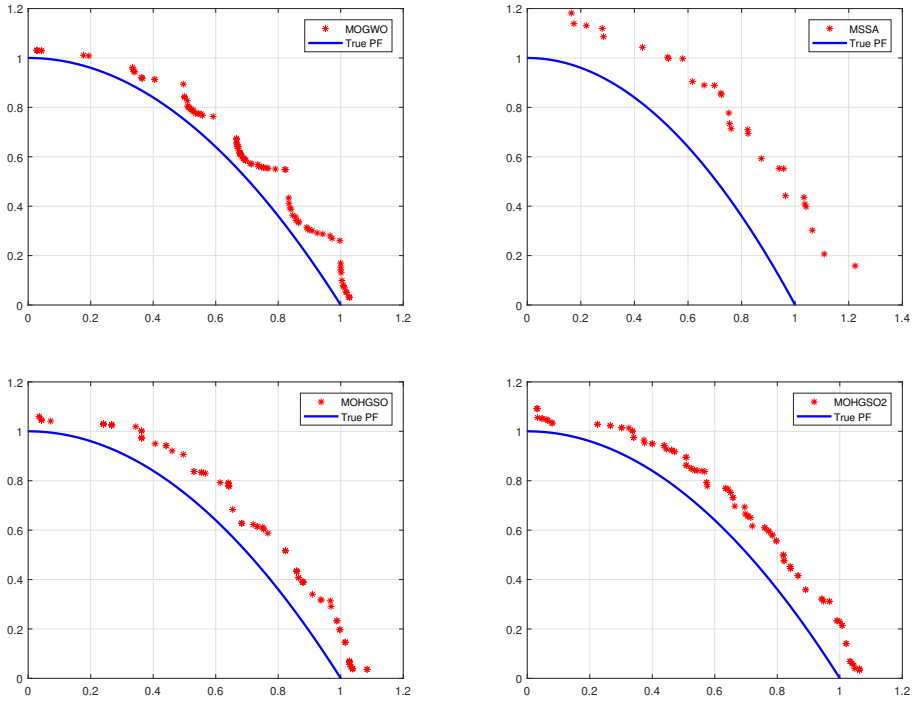


Figure 3.17: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF4.

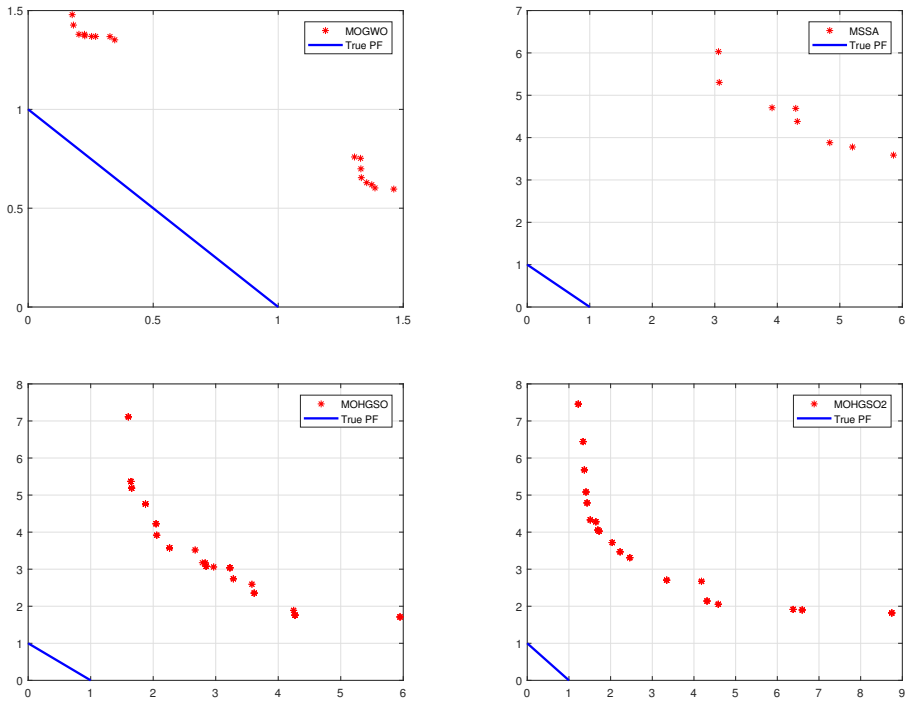


Figure 3.18: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF5.

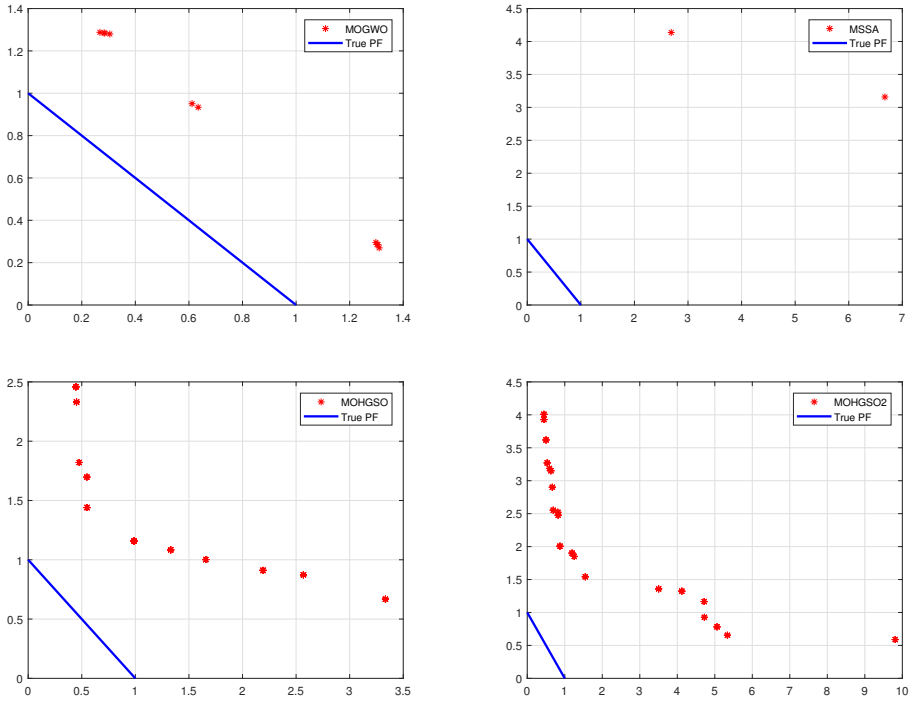


Figure 3.19: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF6.

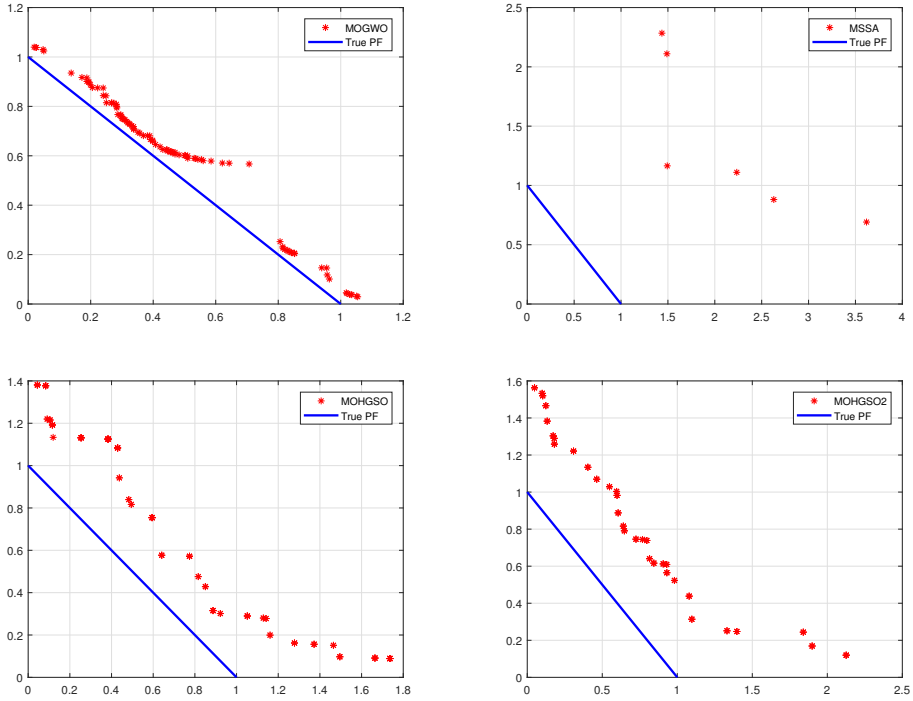


Figure 3.20: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF7.

Table 3.7: SP Values for Algorithms on UF1 to UF7 Problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
UF1	Best	4.12E-03	1.92E-02	4.31E-03	**0.00E+00**
	Worst	1.58E-02	2.80E-01	3.08E-02	**3.39E-03**
	Mean	7.06E-03	1.42E-01	1.67E-02	**2.85E-03**
	Median	4.86E-03	1.22E-01	1.75E-02	**2.67E-03**
	Std	4.94E-03	1.09E-01	1.18E-02	**1.49E-04**
UF2	Best	1.23E-02	1.88E-02	**6.55E-03**	6.93E-03
	Worst	1.86E-02	2.32E-02	2.04E-02	**7.93E-03**
	Mean	1.43E-02	2.10E-02	1.24E-02	**7.37E-03**
	Median	1.29E-02	2.10E-02	1.14E-02	**7.39E-03**
	Std	2.61E-03	1.55E-03	5.05E-03	**4.15E-04**
UF3	Best	4.56E-02	7.85E-02	2.62E-03	**0.00E+00**
	Worst	6.58E-02	4.83E-01	1.94E-02	**2.52E-03**
	Mean	5.59E-02	1.95E-01	8.46E-03	**5.05E-04**
	Median	5.55E-02	1.21E-01	5.07E-03	**5.23E-07**
	Std	7.24E-03	1.64E-01	6.75E-03	**1.13E-03**
UF4	Best	6.87E-03	1.65E-02	8.87E-03	**6.56E-03**
	Worst	1.82E-02	4.31E-02	1.46E-02	**7.94E-03**
	Mean	1.10E-02	3.17E-02	1.27E-02	**7.74E-03**
	Median	1.02E-02	3.69E-02	1.34E-02	**7.68E-03**
	Std	4.31E-03	1.26E-02	2.27E-03	**1.90E-04**
UF5	Best	9.07E-03	2.04E-01	**0.00E+00**	**0.00E+00**
	Worst	1.41E-01	9.29E-01	3.72E-02	**3.23E-03**
	Mean	5.52E-02	5.18E-01	9.19E-03	**4.50E-03**
	Median	2.97E-02	5.37E-01	3.67E-03	**3.58E-03**
	Std	5.46E-02	2.70E-01	1.58E-02	**2.07E-04**
UF6	Best	1.40E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Worst	5.32E-02	2.63E+00	1.17E-01	**7.03E-03**
	Mean	2.81E-02	9.47E-01	4.10E-02	**3.07E-03**
	Median	2.23E-02	7.52E-01	0.00E+00	**4.57E-03**
	Std	1.52E-02	1.03E+00	5.70E-02	**1.07E-04**
UF7	Best	7.97E-03	1.48E-01	6.21E-03	**0.00E+00**
	Worst	1.96E-02	5.96E-01	3.31E-02	**4.13E-03**
	Mean	1.31E-02	3.38E-01	2.20E-02	**8.27E-04**
	Median	1.34E-02	3.54E-01	2.69E-02	**2.63E-03**
	Std	4.50E-03	1.70E-01	1.25E-02	**1.85E-03**

bi-objective ones. The results evidence that both MOHGSO1 and MOHGSO2 algorithms are able to provide very competitive and promising results. The statistical results of MOGWO on the UF9 benchmark problem in Tables 3.8 show that MOGWO is able to outperform our suggested algorithms. However, the statistical results for the IGD metric show the significantly superior performance of MOHGSO2 on UF8 and UF10. The superior performance of our proposed MOHGSO2 can be observed in all statistical results on UF8 and UF10. It is worth mentioning the competitive results obtained from the proposed MOHGSO1 in all statistical results in the tri-objective test functions.

Table 3.8: IGD values for Algorithms on UF8 to UF10 problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
UF8	Best	1.66E-02	1.87E-02	2.82E-03	**2.81E-03**
	Worst	3.07E-02	3.62E-02	3.01E-03	**2.84E-03**
	Mean	2.49E-02	3.15E-02	2.91E-03	**2.83E-03**
	Median	2.40E-02	3.51E-02	2.91E-03	**2.83E-03**
	Std	5.83E-03	7.43E-03	7.41E-05	**1.38E-05**
UF9	Best	**1.26E-03**	3.08E-02	4.59E-03	4.87E-03
	Worst	**3.09E-03**	3.65E-02	5.04E-03	5.10E-03
	Mean	**1.85E-03**	3.35E-02	4.81E-03	5.02E-03
	Median	**1.74E-03**	3.31E-02	4.79E-03	5.05E-03
	Std	7.22E-04	2.07E-03	1.64E-04	**8.63E-05**
UF10	Best	5.69E-03	1.13E-01	4.25E-03	**2.93E-03**
	Worst	**6.97E-03**	1.75E-01	1.09E-02	1.09E-02
	Mean	6.36E-03	1.49E-01	7.29E-03	**4.66E-03**
	Median	6.42E-03	1.55E-01	7.36E-03	**3.18E-03**
	Std	**6.44E-04**	2.59E-02	2.51E-03	3.51E-03

The more challenging UF9 test problem has a separated Pareto front. The results in Table 3.9 show that the coverage of MOHGSO2 is very good along the objectives. MOGWO provides better statistical results on the Sp metric in UF10. However, the best results for Std belong to MOGWO. The obtained results of MOHGSO1 and MOHGSO2 are almost similar in UF10, whereas MOHGSO1 ranks third in all three-objective test functions. Based on Table 3.8, the findings show that the MOHGSO2 algorithm which is based on crowding distance provides superior performance in terms of diversity.

Table 3.9: SP values for Algorithms on UF8 to UF10 problems

Problem		MOGWO	MSSA	MOHGSO1	MOHGSO2
UF8	Best	1.63E-02	6.42E-01	4.58E-02	**1.04E-02**
	Worst	3.41E-02	2.29E+00	1.27E-01	**1.24E-02**
	Mean	2.52E-02	1.31E+00	7.74E-02	**1.12E-02**
	Median	2.36E-02	1.06E+00	6.61E-02	**1.13E-02**
	Std	**8.01E-03**	6.42E-01	3.23E-02	8.19E-03
UF9	Best	3.51E-02	8.32E-01	2.25E-02	**1.76E-02**
	Worst	5.52E-02	1.26E+00	5.65E-01	**4.81E-02**
	Mean	4.13E-02	1.00E+00	1.79E-01	**3.03E-02**
	Median	3.86E-02	8.66E-01	1.08E-01	**2.82E-02**
	Std	**7.95E-03**	2.18E-01	2.25E-01	1.23E-02
UF10	Best	1.65E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Worst	**1.09E-01**	5.16E+00	3.21E-01	2.47E-01
	Mean	**4.88E-02**	2.61E+00	1.76E-01	1.70E-01
	Median	**2.08E-02**	2.76E+00	1.76E-01	2.11E-01
	Std	**5.23E-02**	1.94E+00	1.15E-01	1.00E-01

The Pareto optimal solutions obtained by the competitor algorithms are illustrated in

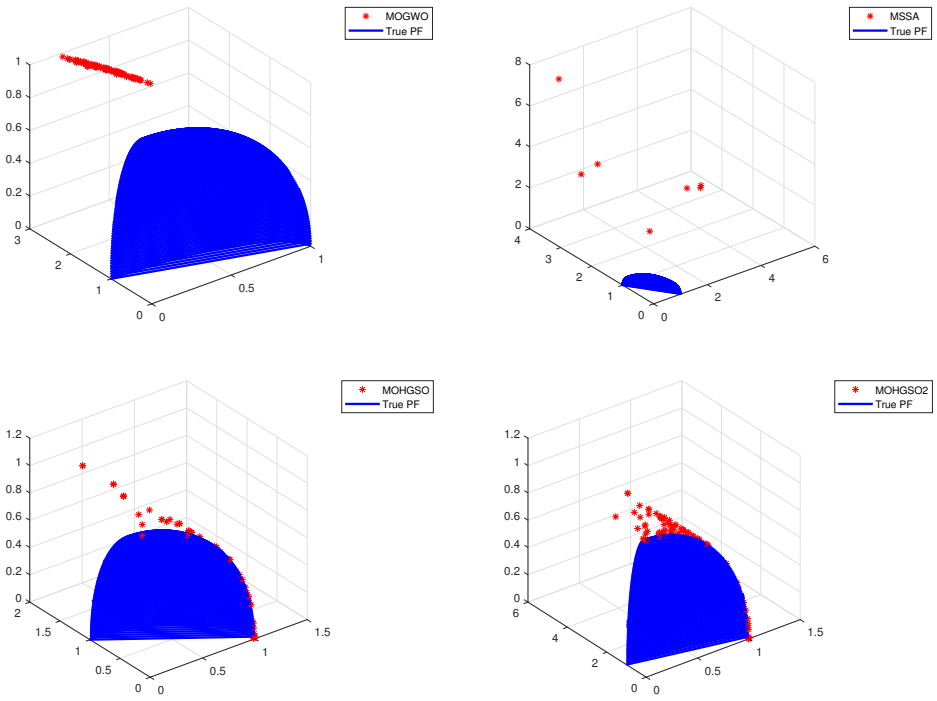


Figure 3.21: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF8.

Figures 3.21, 3.22, and 3.23. The Pareto optimal front obtained by MOHGSO2 on UF8 is close to the true Pareto, in which the MOGWO algorithm shows very low convergence and coverage. The convergence and coverage of the MOGWO algorithm can be clearly seen in the Pareto front obtained in Figure 3.22. Another notice here is the poor distribution of all obtained front of MSSA, which shows that this algorithm is not able to show good coverage on these benchmark problems.

In fact, the adopted UF test problems have been designed to evaluate the performance of an algorithm when dealing with complicated Pareto set topologies. UF benchmark problems present different properties regarding separability, multi-modality, and different Pareto front geometries including convexity, concavity, discontinuities, etc. As a consequence, we can see the good performance of the suggested MOHGSO1 and MOHGSO2 in most of the benchmark problems adopted. From our experimental study, we can consider that these algorithms are a good choice for dealing with MOPs with complicated Pareto sets.

MOHGSO2 is a Pareto-based algorithm, where an effective archiving and leader selection techniques that utilize crowding distance calculations are introduced. While the MOHGSO1 incorporates the grid mechanism and roulette wheel method in the manner of selection leaders and the updating process of the external archives. The experimental study showed competitive results of the proposed MOHGSO2 when approximating the Pareto front of well-known benchmark problems. In contrast, the low performance of

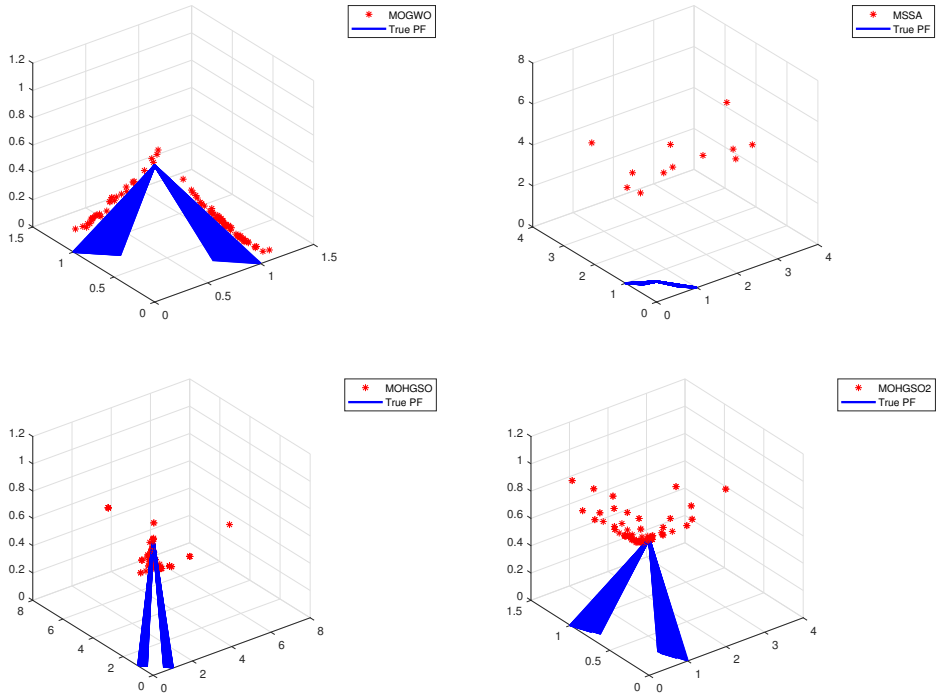


Figure 3.22: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF9.

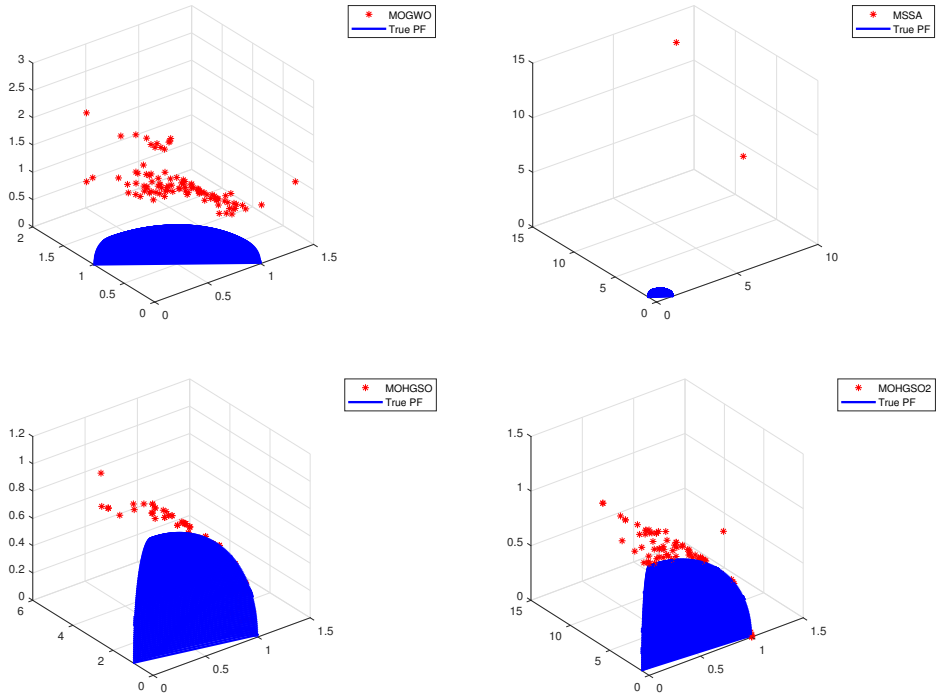


Figure 3.23: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for UF10

MOGWO1 in proximity and distribution of solutions along the Pareto optimal front could be in most of the problems because they are considered among the most challenging test problems in the literature.

3.3.4 Results on multi-objective engineering design problems

This section examines the application of MOHGSO2, the best-performing approach, to four real engineering design problems including Four-bar truss design, Speed reducer design, Disk brake design, and Welded beam design.

- **Four-bar truss design problem**
  The objective of this design problem is to minimize the volume and displacement of a four-bar truss concurrently. The design space is unconstrained, with four independent design variables.

$$minimize \begin{cases} f_1(x) = L(2x_1 + \sqrt{2x_2} + \sqrt{x_3} + x_4) \\ f_2(x) = \frac{FL}{E}(\frac{2}{x_2} + \frac{2\sqrt{2}}{x_2} - \frac{2\sqrt{2}}{x_3} + \frac{2}{x_4}) \end{cases} \tag{3.3.1}$$

  where
  $F = 10,\ E = 2e^5,\ L = 200$
  $1 \leq x_1,\ x_4 \leq 3, \quad \sqrt{2} \leq x_2,\ x_3 \leq 3$

- **Speed reducer design problem**
  This design challenge seeks to optimize a gear assembly by simultaneously minimizing weight and transverse deflection. Seven design parameters are considered: gear face width, module, pinion tooth count, bearing 1 spacing, bearing 2 spacing, and the diameters of shafts 1 and 2. Constraints are imposed on several factors, including gear tooth bending and contact stresses, shaft transverse deflections, and shaft stresses [75].

$$min \begin{cases} f_1 = 0.7854x_1x_2^2 \times (3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1 \\ \qquad \times(x_6^2 + x_7^2) + 7.4777 * (x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \\ f_2 = \sqrt{(745x_4)/(x_2x_3)^2 + 1.69e^7}/(0.1x_6^3) \end{cases} \tag{3.3.2}$$

  Subject to

  $g_1(x) = 27/(x_1x_2^2x_3) - 1$

  $g_2(x) = 397.5/(x_1x_2^2x_3^2) - 1$

$$g_3(x) = (1.93x_4^3)/(x_2x_3x_6^4) - 1$$

$$g_4(x) = (1.93x_5^3)/(x_2x_3x_7^4) - 1$$

$$g_5(x) = \sqrt{(745x_4/x_2x_3)^2 + 16.9e^6}/110x_6^3 - 1$$

$$g_6(x) = \sqrt{(745x_5/x_2x_3)^2 + 157.5e^6}/85x_7^3 - 1$$

$$g_7(x) = x_2x_3/40 - 1$$

$$g_8(x) = 5x_2/x_1 - 1$$

$$g_9(x) = x_1/12x_2 - 1$$

$$g_{10}(x) = 1.5x_6 + 1.9/x_4 - 1$$

$$g_{11}(x) = 1.1x_7 + 1.9/x_5 - 1$$

$$2.6 \le x_1 \le 3.6, \quad 0.7 \le x_2 \le 0.8, \quad 17 \le x_3 \le 28,$$
$$7.3 \le x_4 \le 8.3, \quad 7.3 \le x_5 \le 8.3, \quad 2.9 \le x_6 \le 3.9, \quad 5 \le x_7 \le 5.5$$

- **Disk brake design problem**
  This design problem aims to minimize both the brake's mass and stopping time. Four design variables are considered: inner and outer disk radii, engaging force, and the number of friction surfaces. Five constraints are imposed, including friction surface spacing, brake length, surface pressure, maximum temperature, and braking torque [108].

$$minimize \begin{cases} f_1(x) = 4.9e - 5(x_2^2 - x_1^2)(x_4 - 1) \\ f_2(x) = (9.82e^6)\frac{x_2^2 - x_1^2}{x_3x_4(x_2^3 - x_1^3)} \end{cases} \tag{3.3.3}$$

  Subject to
  $g_1(x) = 20 + x_1 - x_2$
  $g_2(x) = 2.5(x_4 + 1) - 30$
  $g_3(x) = \frac{x_3}{3.14(x_2^2 - x_1^2)^2} - 0.4$
  $g_4(x) = 2.22e - 3x_3\frac{x_2^3 - x_1^3}{(x_2^2 - x_1^2)^2} - 1$
  $g_5(x) = 900 - \frac{2.66e-2x_3x_4(x_3^3 - x_1^3)}{x_2^2 - x_1^2}$
  where

$\forall g_i \leq 0$

$55 \leq x_1 \leq 80, \quad 75 \leq x_2 \leq 110, \quad 1000 \leq x_3 \leq 3000, \quad 2 \leq x_4 \leq 20$

- **Welded beam design problem**

  This design problem seeks to minimize both fabrication cost and end deflection in a welded beam. Four design variables are considered: height, welded joint length, thickness, and width. Four constraints are imposed: shear stress, bending stress, weld length, and buckling load [108].

$$minimize \begin{cases} f_1(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2) \\ f_2(x) = \frac{4PL^3}{Ex_3^3x_4} \end{cases} \tag{3.3.4}$$

  Subject to

  $g_1(x) = 13600 - \tau$

  $g_2(x) = 30000 - \sigma$

  $g_3(x) = x_4 - x_1$

  $g_4(x) = P - 6000$

  where,

  $p = (\frac{4.013*E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2})(1 - ((\frac{x_3}{2L})\sqrt{\frac{E}{4G}}))$

  $sig = \frac{6PL}{x_4x_3^2}$

  $J = 2 * (\sqrt{2}x_1x_2((\frac{x_2^2}{12}) + (\frac{x_1+x_3}{2})^2))$

  $R = \sqrt{(\frac{x_2^2}{4}) + (\frac{x_1+x_3}{2})^2}$

  $M1 = P * (L + \frac{x_2}{2})$

  $\tau 2 = \frac{M1R}{J}$

  $\tau 1 = \frac{P}{\sqrt{2}x_1x_2}$

  $\tau = \sqrt{\tau 1^2 + 2\tau 1\tau 2.\frac{x_2}{2R} + \tau 2^2}$

  $P = 6000, \quad L = 14, \quad E = 30e^6, \quad G = 12e^6$

  $0.125 \leq x_1, \quad x_4 \leq 5, \quad 0.1 \leq x_2, \quad x_3 \leq 10$

A comparative analysis of MOHGSO2 against MOGWO and MSSA is presented for these problems. The IGD metric was computed for 31 runs, with the results summarized in Table 3.10, demonstrated MOHGSO2's superior performance on three of the four problems: Speed reducer design, Disk brake design, and Welded beam design. MOHGSO2 outperforms its competitors across all statistical measures of the IGD-metric for these problems.

Table 3.11 displays the statistical results of the Sp metric. These findings further reinforce MOHGSO2's effectiveness as a viable solution for multi-objective optimization problems with complex constraints.

Table 3.10: IGD Values Obtained for Engineering Design Problems

Problem		MOGWO	MSSA	MOHGSO2
4-b Truss	Best	**2.36E-01**	6.39E-01	3.03E-01
	Worst	**5.63E-01**	2.02E+00	1.01E+00
	Mean	**3.52E-01**	1.02E+00	5.02E-01
	Median	**3.26E-01**	8.95E-01	4.43E-01
	Std	**8.67E-02**	3.31E-01	1.66E-01
SRD	Best	1.03E+01	1.16E+01	**9.51E+00**
	Worst	1.34E+01	2.04E+01	**9.92E+00**
	Mean	1.15E+01	1.62E+01	**9.74E+00**
	Median	1.14E+01	1.63E+01	**9.75E+00**
	Std	7.23E-01	2.79E+00	**1.06E-01**
Disk	Best	5.05E-03	1.10E-02	**4.32E-03**
	Worst	1.22E-02	5.98E-02	**6.49E-03**
	Mean	6.63E-03	2.34E-02	**5.47E-03**
	Median	6.23E-03	2.17E-02	**5.42E-03**
	Std	1.56E-03	1.04E-02	**5.01E-04**
W Beam	Best	1.67E-02	2.41E-02	**1.25E-02**
	Worst	1.08E-01	3.03E-01	**1.72E-02**
	Mean	4.47E-02	9.88E-02	**1.47E-02**
	Median	4.07E-02	6.49E-02	**1.48E-02**
	Std	2.17E-02	7.85E-02	**1.50E-03**

Tables 3.10 and 3.11 demonstrate that the proposed MOHGSO2 algorithm significantly outperforms its competitors on three of the four engineering design problems. The exception is the Four-bar truss problem, where MOGWO shows superior performance. Specifically for the 4-bar truss design problem: MOGWO ranks first in terms of convergence, MSSA demonstrates poor performance, and MOHGSO2 achieves a strong second place.

As illustrated in Figure 3.24, MOHGSO2 exhibits excellent convergence to the true Pareto front with very good coverage for this problem. This visual representation reinforces the algorithm's strong performance, even when not ranking first.

Speed Reducer and Disk Brake Design Problems, MOHGSO2 demonstrates superior performance, achieving higher mean values for IGD and Sp metrics compared to other algorithms. MOGWO consistently ranks second in all statistical results for the IGD metric. Figures3.25 and 3.26 visually confirm MOHGSO2's excellent convergence and coverage for these problems.

Welded Beam Design Problem, in this four-constraint problem, MOHGSO2 achieves the best statistical results for both IGD and Sp metrics. MSSA ranks second in most statistical results for the Sp metric. Figure 3.27 illustrates MOHGSO2's efficiency in accurately estimating the Pareto optimal front with excellent coverage of the objective space.

MOHGSO2 outperforms MOGWO and MSSA in three out of four engineering de-

Table 3.11: Sp Values Obtained on Engineering Design Problems

Problem	Metric	MOGWO	MSSA	MOHGSO2
4-bar truss	Best	2.41E+00	1.19E+00	**5.96E-01**
	Worst	6.44E+00	6.56E+00	**8.96E-01**
	Mean	4.14E+00	3.38E+00	**7.29E-01**
	Median	3.86E+00	3.20E+00	**7.26E-01**
	Std	1.17E+00	1.21E+00	**6.68E-02**
SRD	Best	1.99E+01	1.36E-02	**0.00E+00**
	Worst	4.48E+01	2.66E+01	**1.20E+01**
	Mean	2.78E+01	9.63E+00	**1.41E+00**
	Median	2.68E+01	7.01E+00	**3.69E-01**
	Std	6.25E+00	7.99E+00	**2.67E+00**
Disk	Best	6.68E-02	**4.53E-02**	4.77E-02
	Worst	2.59E-01	5.41E-01	**8.21E-02**
	Mean	1.28E-01	1.40E-01	**6.80E-02**
	Median	1.20E-01	9.45E-02	**7.01E-02**
	Std	3.61E-02	1.15E-01	**8.62E-03**
Welded beam	Best	1.47E-01	**1.10E-01**	1.25E-01
	Worst	5.89E-01	7.82E-01	**2.27E-01**
	Mean	3.19E-01	2.23E-01	**1.73E-01**
	Median	3.18E-01	1.87E-01	**1.72E-01**
	Std	9.78E-02	1.34E-01	**2.42E-02**

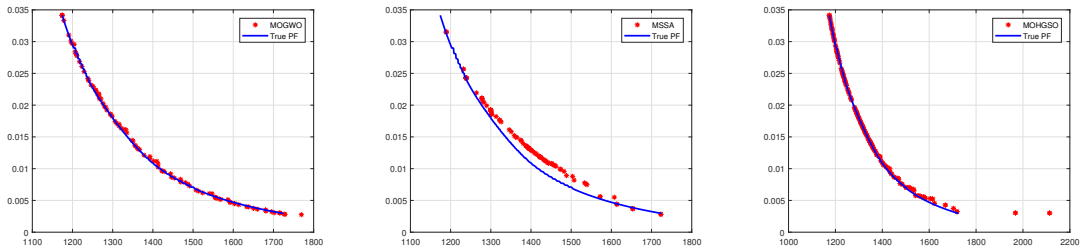


Figure 3.24: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for Four bar truss.

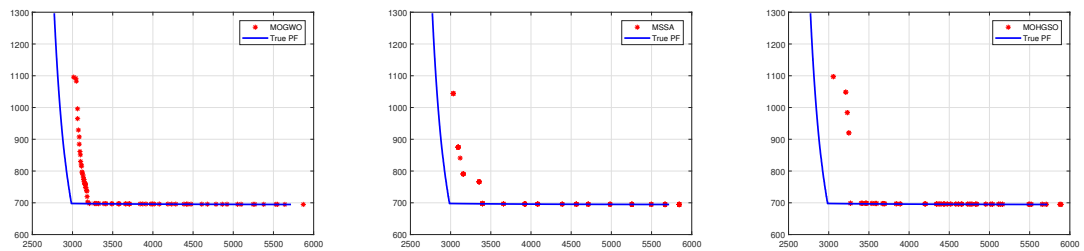


Figure 3.25: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for the SRD.

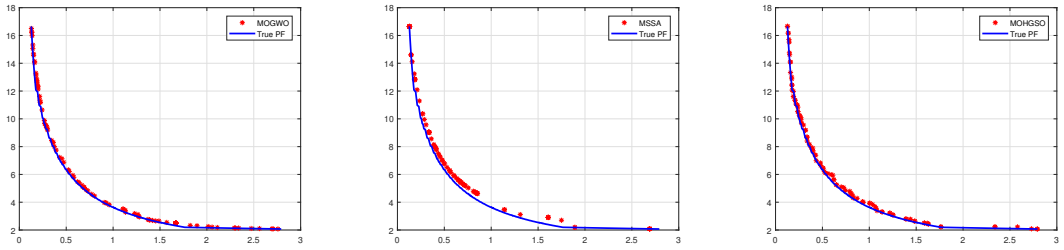


Figure 3.26: Optimal Pareto Sets: Comparing MOGWO, MSSA, and MOHGSO algorithms for the Disk brake.

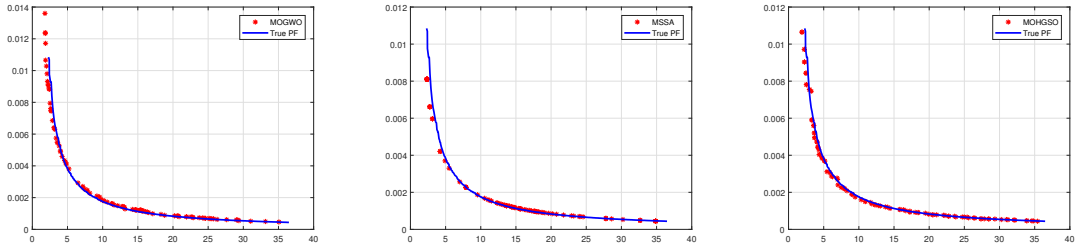


Figure 3.27: Best Pareto optimal fronts obtained by MOGWO, MSSA, and MOHGSO2 of the Welded beam.

sign problems. These results demonstrate MOHGSO2's ability to produce highly diverse Pareto frontiers and provide very satisfactory solutions across multiple problem types.

3.3.5 Discussion of the results

Multi-objective HGSO algorithms divide search agents into groups to mimic Henry's law, which describes gas behavior. This approach maintains a balance between exploration and exploitation within the search space. Adjusting Henry's coefficient during optimization demonstrates the improvement of results for all agents across iterations, demonstrating the algorithms' capability to address complex Multi-objective Optimization Problems (MOPs).

The algorithms' superiority is evident both quantitatively and graphically. MOHGSO algorithms consistently identify well-distributed collections of non-dominated solutions that closely approach the true Pareto front across conflicting objectives. These algorithms effectively discover discontinuous regions of the true Pareto and avoid local optima, even in high-dimensional problems with diverse shapes. While their efficiency slightly decreases as the number of objectives increases, both proposed algorithms remain highly competitive compared to other selected algorithms, even when dealing with more than two objectives.

The comparative analysis demonstrates that Multi-objective HGSO algorithms are highly competitive against the selected algorithms. The IGD metric results highlight the proposed algorithms' strong ability to converge towards the true Pareto front. This convergence stems from updating search agent positions which are attracted by non-

dominated solutions in the external archives. Moreover, Multi-objective HGSO algorithms select two leaders from different archives to update a single individual's position, rather than the entire population. This mechanism enhances solution diversity compared to MOGWO and MSSA.

Based on two performance metrics, the proposed algorithms outperform MOGWO and MSSA. Multi-objective HGSO versions provided competitive results across the ZDT-series, which are known for diverse Pareto optimal fronts (convex, non-convex, discontinuous, and multi-modal). The algorithms' performance and ability to approach the true Pareto front are also evident in the DTLZ-series. Furthermore, Multi-objective HGSO versions consistently find promising non-dominated solutions for Uf problems in independent runs, demonstrating their capability to identify Pareto optimal fronts of any shape.

The MOGWO algorithm demonstrates superior performance on most ZDT test functions and the four engineering design problems, owing to its robust exploration and exploitation capabilities controlled by parameters A and c. However, its leader selection strategy, which chooses three leaders from the same archive, may potentially reduce solution diversity if identical leaders are selected. In some cases, this could guide the population towards a single region rather than diverse areas.

In contrast, the MSSA algorithm relies heavily on the control parameter c_1 for position updates. The exponential decrease of this parameter during optimization might lead to premature convergence or local Pareto optima stagnation, particularly in problems with multiple local optima. This limitation is evident in Figure 3.5, where MSSA struggles to avoid the 2^{21} local Pareto optima in ZDT4. Additionally, MSSA's leader selection based on archive solution ranking may not effectively maintain archive diversity. This deficiency can result in poor coverage of higher-dimensional problems.

The Multi-objective HGSO algorithms address these limitations by selecting leaders from different archives, potentially enhancing solution diversity and improving performance across various problem types.

The superior performance of the proposed MOHGSO2 algorithm, which utilizes crowding distance, can be primarily attributed to its archiving and leader selection strategies. Using an external archive preserves non-dominated solutions, ensuring good convergence to the Pareto front. These archives serve as repositories from which two leaders are selected to update the main population's position. This leader selection process guides the population towards:

- Regions of non-dominance within the search space, promoting convergence;
- A well-distributed Pareto front, facilitated by the crowding distance operator.

By selecting leaders from different archives, MOHGSO2 enhances both the diversity and quality of solutions. This dual-focus approach allows the algorithm to maintain a balance between exploration of the search space and exploitation of promising areas, resulting in improved overall performance across various multi-objective optimization problems.

The Multi-objective HGSO algorithms employ crowding distance or grid mechanisms as density estimators in two crucial steps: leader selection strategy and archive updating when full, this dual application significantly enhances solution diversity.

Unlike many multi-objective approaches, our proposed Multi-objective HGSO algorithms utilize multiple external archives. This strategy reinforces the elitism concept and generates diverse solutions, enabling efficient global exploration of the search space without excessive time wasting.

Based on the preceding analysis, we can conclude that the proposed Multi-objective HGSO algorithms serve as effective tools for solving multi-objective optimization problems. Notably, MOHGSO2, the best-performing approach, demonstrates remarkable capability in tackling four engineering design problems. The key factors contributing to its success include:

- High convergence inherited from the standard HGSO algorithm's performance.
- Strong exploration and exploitation capabilities due to the updating procedure that relies on the parameters $S_{(i,j)}$ and γ.
- Guaranteed coverage and convergence through the decreasing of γ and $S_{i,j}$ parameters during optimization.
- Utilization of external archives to retain the optimal non-dominated solutions.
- Multiple archives for storing local and global non-dominated solutions while preserving elitism.
- Effective archiving strategy ensuring both convergence and diversity within the population.
- Archive maintenance improving solution accuracy and convergence speed while avoiding local optima.
- Multi-leader selection from multiple archives, enabling both local and global search space exploration.
- Multi-leaders guiding individuals towards rarely explored parts of the search space, increasing coverage.
- Two leaders per particle improving both convergence and diversity abilities.
- Incorporation of grid mechanisms/crowding distance in selection and deletion strategies, enhancing diversity in objective space.
- Grid mechanism use in archive maintenance and leader selection positively affects the diversity of non-dominated solutions (specific to MOHGSO1).

- Roulette-wheel selection with a lower probability for crowded segments, ensuring local optima avoidance.
- Combination of crowding distance and random leader selection from $N + 1$ external archives maintaining solution diversity (specific to MOHGSO2).

3.4 Conclusion

This work introduces two novel versions of the Multi-objective HGSO algorithm: MOHGSO1 and MOHGSO2, designed to solve multi-objective optimization problems. These Pareto-based algorithms incorporate efficient archiving and leader selection strategies, utilizing local and global archives to rapidly identify high-quality solutions from the multi-population and effectively capture the Pareto optimal front.

To adapt the single-objective HGSO for multi-objective problems. MOHGSO1 integrates a grid mechanism and roulette wheel method for leader selection and external archive updating. MOHGSO2 combines crowding distance with random selection in archiving and leader selection strategies to maintain solution diversity. Both algorithms incorporate diversity techniques, specifically in selection and deletion methods for the external archive of non-dominated solutions.

The Multi-objective HGSO algorithms were evaluated using: i) twelve benchmark functions (ZDT and DTLZ functions) with diverse Pareto optimal fronts, ii) ten multi-objective benchmark problems (CEC2009 benchmark functions), and compared with MOGWO and MSSA using two established metrics. Results show superior convergence and highly competitive diversity, especially in bi-objective problems, with slightly decreased effectiveness but still competitive results for problems with more than two objectives, and sensitivity to several controlling parameters, potentially reducing flexibility.

A limitation of MOHGSO2's crowding distance implementation is its potential to decrease solution diversity. Sorting the archive based on crowding distance values raises the probability of selecting Pareto point boundaries (extremities) as leaders (due to their infinite crowding distance value). This may steer the main population towards a single region instead of varied areas in the search space, potentially compromising solution diversity.

CHAPTER 4

Feature Selection Using Binary Henry Gas Solubility Optimization

4.1 Introduction

In this chapter, we applied the proposed Binary Henry Gas Solubility Optimization (B-HGSO) for single-objective feature selection. B-HGSO focuses on optimizing a single objective that balances feature reduction and classification accuracy into a binary format using eight transfer functions, to identify feature subsets that maximize the performance of the predictive K-NN model. The proposed method leverages the principles of Henry Gas Solubility Optimization to enhance feature selection processes, demonstrating effectiveness in improving classification accuracy and reducing feature dimensionality across various datasets, including those with high-dimensional features. By incorporating single objective optimization strategies, the method aims to address the challenges of noisy and inconsistent features in data mining applications, showcasing promising results in enhancing the quality of the K-NN model through efficient feature subset identification.

4.2 Implementation of Binary HGSO

Before describing our contributions, we start by presenting the ideas of the suggested feature selection solution which is a three-phased process including initialization, evaluation, and classification.

4.2.1 Initialization

An initial population of N candidate solutions are generated randomly as in 3.2.1, which represents a feature subsets to be selected for the evaluation process.

The values for each candidate solution are continuous and limited to the range [0, 1]. Here, the gas population is updated, and the position updating must be converted to a binary representation, where the continuous positions are transformed into binary values using the chosen transfer functions **??** .

As given in Equation 4.2.1, each position represents a feature subset, where 1 implies that the feature is selected; otherwise considered the feature is unselected.

$$X = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^D \\ x_2^1 & x_2^2 & \cdots & x_2^D \\ \vdots & \vdots & \ddots & \vdots \\ x_N^1 & x_N^2 & \cdots & x_N^D \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{4.2.1}$$

4.2.1.1 Transfer Function

In this work, two different types of transfer functions are used and their impacts on the performance of the suggested algorithm are studied. The transfer function determines the probability of updating the binary solution's values from 0 to 1 and vice-versa. Four S-Shaped and four V-Shaped transfer functions determine how continuous values are transformed into binary values and influence the algorithm's exploration and exploitation capabilities[89]. Finally, each individual's dimension is converted into a binary value. The conversion process ensures that the solutions are represented as binary feature subsets, with each feature being either selected (1) or not selected (0).
In S-shaped functions, the solution is updated based on 4.2.2:

$$x_i^d(t+1) = \begin{cases} 1 & if\ rand\ <\ S function(x_i^d(t+1)) \\ 0 & Otherwise \end{cases} \tag{4.2.2}$$

where, rand is a random number with uniform distribution in $[0, 1]$.
In V-shaped functions, the solution is updated based on 4.2.3:

$$x_i^d(t+1) = \begin{cases} \neg x_i^d(t), & rand\ <\ V function(x_i^d(t+1)) \\ x_i^d(t), & Otherwise \end{cases} \tag{4.2.3}$$

Table4.1 outlines the mathematical formulas of the eight transfer functions used in our study.

Table 4.1: **S-shaped and V-shaped families of Transfer functions**

Family of S-shaped functions		Family of V-shaped functions	
Name	Transfer function	Name	Transfer function
S1	$T(x) = \frac{1}{1+e^{x}}$	V1	$T(x) = \lvert\tanh(x)\rvert$
S2	$T(x) = \frac{1}{1+e^{2x}}$	V2	$T(x) = \lvert\operatorname{erf}(\frac{\pi}{2}x)\rvert$
S3	$T(x) = \frac{1}{1+e^{3x}}$	V3	$T(x) = \lvert\frac{x}{\sqrt{1+x^2}}\rvert$
S4	$T(x) = \frac{1}{1+e^{x/2}}$	V4	$T(x) = \lvert\frac{2}{\pi}\arctan(\frac{\pi}{2}x)\rvert$

4.2.2 Evaluation

The objective function is used to determine the quality of the solutions. Iteratively, the fitness function is computed for each solution by representing both classification performance and the number of selected features. This can be achieved by combining the classification performance and feature selection objectives into a single objective function, which can be optimized to find the best binary solution that balances the trade-off between these two objectives. The objective value is given in equation 4.2.4:

$$f_i = \alpha \times Err_i + \beta \times \frac{d_i}{D} \tag{4.2.4}$$

where $\alpha = 0.5$ and $\beta = 1 - \alpha = 0.5$ are parameters designed to balance the classification error rate Err_i and the number of selected features d_i, and D represents the total number of features in the original data.

Here, the updating solutions consist of applying the HGSO process, it will iteratively update the binary solution and compute the fitness of the solution, then evaluating the fitness of the new population and the best solution is recorded as the best feature subset. This process is repeated until a pre-defined maximum number of iterations.

4.2.3 Classification

In this study, the dataset is divided randomly into two parts: 50% for the training set to build the learning model, and 50% for the testing set to assess the selected features. Using the K-Nearest Neighbor (KNN) classifier for evaluation purposes in a wrapper-based feature selection method to assess the quality of the features selected by the feature selection method, by fixing the value of k to 5.

The basic steps of the suggested b-HGSO algorithm are listed briefly as follows:

- Initial population is generated, which is consisted of N gases. Each gas will be considered as a subset of features (position).

- Evaluate the initial position (subset). In this step, we calculate the fitness function of each gas, each position is evaluated according to the k-NN classifier to detect the accuracy rate and the number of features selected.
- Select the best and optimal gases with the highest fitness value as the best feature subset and the optimal feature subset, respectively.
- Updating Henry's coefficient and solubility for cluster and gases, respectively;
- The position update is carried out according to the best and optimal feature subsets selected;
- Rank and select the worst agents, then updating their position is performed;
- Update the best and the optimal feature subset.

Eventually, the pseudo-code of the suggested algorithm is presented in Algorithm 5

Algorithm 5: Pseudo-code of b-HGSO algorithm.

Initialize the number of N gas and types i, H_j, $P_{(i,j)}$, C_j, l_1, l_2 and l_3
Randomly Initialize an agent of gases positions $\in$ [0,1]
Divide the population into equal n clusters
Evaluate the fitness of agents by using equation4.2.4
Get the best gas $X_{(i,best)}$ in each cluster, and the best search agent X_{best}
while $t < Max\_It$ **do**
    for *each search agent* **do**
        Update the positions of all search agents using equation3.2.5
    end
    Update Henry's coefficient for each cluster by equation3.2.3
    Update solubility of each gas by equation 3.2.4
    Rank and select the number of worst agents using 3.2.6
    Update the position of the worst agents using 3.2.7
    Update the position of agents into a binary position based on equation 4.2.1
    Evaluate all search agents using the objective function
    Update the positions of the best agents $X_{(i,best)}$ and X_{best}
end
t = t+1.
return X_{best}

4.3 Experimental results and discussion

In this section, the efficiency of both types of transfer functions is investigated, and compared with each other in order to estimate them and select the best. A comparison

with eight recent modifications of B-HGSO is conducted to validate the performance of the selected best transfer function.

4.3.1 Datasets

The suggested eight binary HGSO algorithms are evaluated using 12 benchmark functions from UC Irvine Machine Learning Repository [11]. The datasets are described in terms of the number of features, the number of instances, and the domain of dataset in Table 4.2.

Table 4.2: Benchmark Datasets used

Dataset	Instances	No. Features	Data area
Breast-cancer	699	9	Biology
Dermatology	366	34	Biology
DNA	318	57	Biology
Exactly	1000	13	Biology
Exactly2	1000	13	Biology
HeartEW	270	13	Biology
Lung-Cancer	32	56	Biology
Lymphography	148	18	Biology
M-of-n	1000	13	Biology
Mushroom	8124	22	Biology
Primary-tumor	339	17	Biology
Soybean-small	47	35	Biology

4.3.2 Performance measures

To evaluate the performance of the suggested eight B-HGSO algorithms, some statistical measures are defined as follows:

1. Classification accuracy: the classification accuracy values obtained are reported (mean, and Std) after performing 20 independent runs for each dataset.

   - Mean accuracy: This metric measures the accuracy of the classifier with the selected feature set. It can be calculated using the following equation:

   $$mean = \frac{1}{N}\sum_{k=1}^{N} AvgAcc^{k}$$

   where $AvgAcc^{k}$ is the value of accuracy obtained for k^{th} run.

   - Standard Deviation (Std) measures the variation of the best solutions obtained from running a stochastic optimizer for 20 runs. It uses as an indicator of the stability and robustness of the optimizer. A smaller Std value indicates that the optimizer consistently converges to the same solution, while larger Std values indicate greater variability in the results.

$$Std = \sqrt{\frac{1}{1-N}\sum_{k=1}^{N}(AvgAcc^k - mean)^2}$$

2. Average of Selected Features: This measure shows the average size of the selected features compared to the total number of features when all algorithms are run 20 times and can be formulated as follows:

$$AvgSelection = \frac{1}{N}\sum_{k=1}^{N}\frac{AvgSelection^k}{M}$$

where $AvgSelection^k$ is the selected features for k^{th} run, and M is the total number of features of a dataset.

3. Fitness function: the fitness value metric estimates the performance of all algorithms. We report the results gained (best, worst, mean, and Std) after performing 20 independent runs for each dataset.

   - best: represents the minimum value of the fitness function obtained when the algorithm runs 20 times:

   $$best = \min_k f_k^*$$

   where f_k^* is the best fitness value obtained for k^{th} run.

   - worst: represents the maximum value of the fitness function obtained when the algorithm runs 20 times:

   $$worst = \max_k f_k^*$$

   where f_k^* is the worst fitness value obtained for k^{th} run.

   - mean: represents the average value of the fitness function obtained when the algorithm runs 20 times:

   $$mean = \frac{1}{N}\sum_{k=1}^{N} f_k^*$$

   where f_k^* is the mean fitness value obtained for k^{th} run.

   - Std (Standard Deviation): as we mentioned above, this measure is used as an indicator of the stability and robustness of an optimizer.

   $$Std = \sqrt{\frac{1}{1-N}\sum_{k=1}^{N}(f_k^* - mean)^2}$$

4. The average computational time: (average runtime) is a measure of the typical time taken by the algorithm to the evaluation process, it provides an estimate of the average performance of the algorithm

$$AvgCT = \frac{1}{N}\sum_{k=1}^{N} AvgCT^k$$

where $AvgCT^k$ is the value of computational time in seconds obtained for k^{th} run.

In order to check the effectiveness of the suggested methods, experiments are carried out in four steps:

- First, the performance of the eight suggested methods is assessed based on the selected datasets, as previously mentioned;
- Then, the three best-performing methods in terms of accuracy, number of features, and fitness are compared against four wrapper-based feature selection (FS) meta-heuristic algorithms;
- After that, the Wilcoxon test is used to verify whether the three best-performing methods provide a significant improvement over the selected algorithms.

4.3.3 Sensitivity analysis on B-HGSO

In this study, the eight binary versions of HGSO are implemented in Matlab, and then the obtained results are compared to each other. Several criteria are used to evaluate the performance of the eight different methods for feature selection and determine which one provides the best results for a classification problem. The parameters are an important step in order to achieve optimal performance for the B-HGSO. In this work, the parameter values are adopted by the authors of the standard HGSO algorithm (Table 4.3).

Table 4.3: Parameters setting for experiments

Algorithm	Parameters
B-HGSO	Gases Number: $N = 10$ Maximum number of iterations:100 Dimension corresponds to the number of features $\alpha = 1$, $\beta = 1$ and $K = 1$ Cluster number: $n = 5$ $l_1 = 5E-03$, $l_2 = 1E+02$, and $l_3 = 1E-02$ $M_1 = 0.1$ and $M_2 = 0.2$

Table 4.4 contains the obtained classification accuracies expressed as percentages with their standard deviations, from the eight feature selection (FS) methods suggested in this study. As highlighted all the best results in bold text, the results show that the suggested B-HGSOS1 outperformed the others in all datasets except for Dermatology data, where it ranked second. It is also worth mentioning that B-HGSOs1 outperformed the seven

suggested methods in terms of the mean accuracy with the best stability for the seven datasets used, and obtained 100% accuracy in five different datasets, while the behavior of the remaining suggested methods is almost similar.

Table 4.4: Classification accuracy values obtained using 8 different transfer functions.

	B-HGSOs1	B-HGSOs2	B-HGSOs3	B-HGSOs4	B-HGSOv1	B-HGSOv2	B-HGSOv3	B-HGSOv4
Breast-cancer								
mean(%)	**97,43**	96,29	93,73	94,36	94,87	94,30	94,29	91,73
Std.	2,47E-02	**7,59E-03**	1,23E-02	1,57E-02	2,20E-02	2,91E-02	2,67E-02	5,56E-02
Dermatology								
mean(%)	98,36	97,27	98,36	97,27	**98,91**	97,81	**98,91**	97,27
Std.	8,81E-02	**2,48E-02**	3,57E-02	5,99E-02	7,53E-02	9,79E-02	1,15E-01	1,68E-01
DNA								
mean(%)	**49,06**	42,14	39,62	36,13	35,75	35,72	34,72	33,43
Std.	**3,80E-02**	3,81E-02	2,87E-02	3,75E-02	6,31E-02	6,21E-02	7,46E-02	5,48E-02
Exactly								
mean(%)	**100**	82,00	68,40	66,60	69,43	62,22	59,12	62,50
Std.	**0,00E+00**	5,94E-02	4,31E-02	5,08E-02	1,27E-01	7,41E-02	9,20E-02	1,10E-01
Exactly2								
mean(%)	**78,00**	75,40	74,60	70,60	71,94	66,37	73,50	66,13
Std.	4,93E-02	3,99E-02	**3,21E-02**	4,30E-02	5,12E-02	1,04E-01	6,36E-02	1,23E-01
HeartEW								
mean(%)	**84,44**	73,33	73,33	74,07	80,74	80,67	80,00	79,26
Std.	7,75E-02	6,30E-02	**5,57E-02**	6,74E-02	1,00E-01	1,16E-01	7,89E-02	1,53E-01
Lung-Cancer								
mean(%)	**100**	87,50	70,00	87,50	80,56	80,56	74,38	71,53
Std.	**0,00E+00**	9,64E-02	1,12E-01	1,12E-01	3,76E-02	3,76E-02	1,52E-01	2,40E-01
Lymphography								
mean(%)	**93,06**	81,08	82,43	78,38	73,41	76,53	66,95	70,22
Std.	**4,79E-02**	5,75E-02	6,25E-02	5,83E-02	1,21E-01	1,04E-01	8,76E-02	8,64E-02
M-of-n								
mean(%)	**100**	99,00	72,55	79,60	72,16	75,95	71,89	70,40
Std.	**0,00E+00**	8,77E-02	1,04E-01	7,43E-02	1,01E-01	5,86E-02	1,15E-01	1,03E-01
Mushroom								
mean(%)	**100**	99,33	99,53	99,46	89,80	92,73	89,25	82,42
Std.	**0,00E+00**	1,32E-02	7,23E-03	8,23E-03	1,58E-01	8,83E-02	1,33E-01	1,65E-01
Primary-tumor								
mean(%)	**58,49**	36,47	37,65	36,53	41,77	57,89	35,93	34,34
Std.	8,57E-02	**4,89E-02**	5,34E-02	5,41E-02	7,12E-02	8,86E-02	6,93E-02	8,71E-02
Soybean-small								
mean(%)	**100**	92,08	92,50	95,63	91,04	87,50	86,67	79,37
Std.	**0,00E+00**	9,36E-02	7,84E-02	4,77E-02	1,33E-01	1,92E-01	2,45E-01	2,36E-01

Additionally, Table 4.5 outlines the average number of features selected using the eight suggested feature selection (FS) methods. The findings show that the V-shaped transfer functions family could improve B-HGSO's ability in obtaining a smaller average number of features. Where the B-HGSOv4 approach achieved a lower average number of features i.e., it selects the minimal number of features in comparison to the other seven methods on most of the datasets evaluated.

Table 4.5: Average Of Selected Features obtained using 8 different transfer functions.

	B-HGSOs1	B-HGSOs2	B-HGSOs3	B-HGSOs4	B-HGSOv1	B-HGSOv2	B-HGSOv3	B-HGSOv4
Breast-cancer	4,50	6,39	4,50	5,78	4,06	3,33	3,89	**2,61**
Dermatology	6,47	7,94	7,05	6,76	6,17	6,18	5,88	**5,29**
DNA	2,02	5,67	5,83	5,47	1,76	1,63	1,63	**1,27**
Exactly	5,04	6,08	5,46	5,46	4,15	4,08	3,08	**3,04**
Exactly2	**1,65**	4,80	4,73	5,04	2,53	1,67	1,92	2,05
HeartEW	3,84	7,69	7,69	8,46	**2,31**	3,07	4,61	3,08
Lung-Cancer	0,69	5,77	5,25	5,33	**0,44**	**0,44**	**0,44**	0,50
Lymphography	2,83	2,83	5,31	5,33	2,83	2,67	**1,94**	2,22
M-of-n	5,27	6,23	5,46	5,23	3,27	4,08	3,39	**3,19**
Mushroom	1,93	5,66	5,64	5,11	1,43	1,41	1,36	**1,34**
Primary-tumor	7,22	8,33	7,22	7,77	6,67	6,11	4,44	**3,88**
Soybean-small	0,54	5,49	5,23	5,57	0,89	0,67	0,69	**0,53**

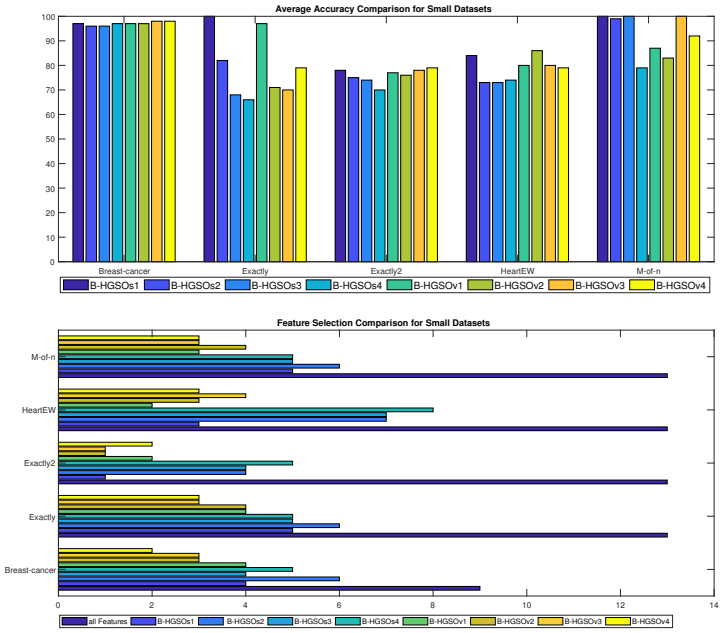


Figure 4.1: Small datasets comparison in term of average accuracy and number of selected features for the 8 different transfer functions.

As mentioned above, the suggested algorithms are tested on twelve benchmark datasets which are categorized into Small datasets(0 – 15 features), Medium datasets(16 - 25 features), and Large datasets (>30 features). Figures 4.1, 4.2, and 4.3 show the performance of the binary suggested B-HGSO algorithms on both performance measures: mean accuracy and the average number of selected features. According to the achieved results by the optimizers, it can be seen that B-HGSOs1 and B-HGSOv4 performed best in terms of classification accuracy and reducing the dimensionality of the data, respectively.

Based on the V-shaped family, it can be observed that B-HGSO achieved better results as compared to B-HGSO with S-shaped transfer functions in order to reduce the number of features for all benchmark datasets. Similarly, B-HGSO based on S-shaped transfer functions shows their best results for the mean classification accuracy as compared to other V-shaped's optimizers.

The fitness function is a key component of binary optimization algorithms, which are used to evaluate the quality of a binary solution by considering the performance of the classifier (accuracy performance metric) and the number of selected features. In this work, classification performance and selection objectives were combined into one objective function to find the best binary solution that balances the trade-off between these two objectives.

The eight binary versions of B-HGSO iteratively update the binary solution and compute the fitness of the solution until a maximum number of iterations is reached. Then, the obtained fitness values of the competitor versions on the 12 datasets are listed in Table 4.6 to outline the comparison results in terms of best, worst, mean, and Std (standard

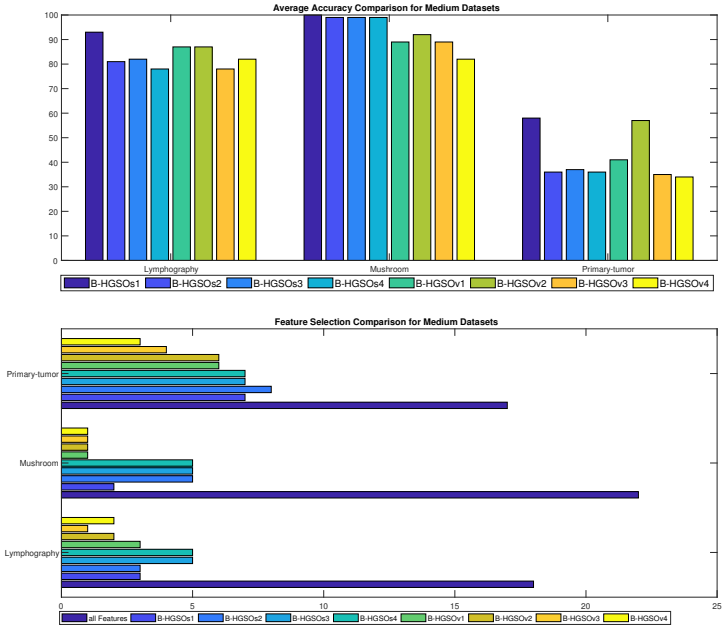


Figure 4.2: Medium datasets comparison in term of average accuracy and number of selected features for the 8 different transfer functions.

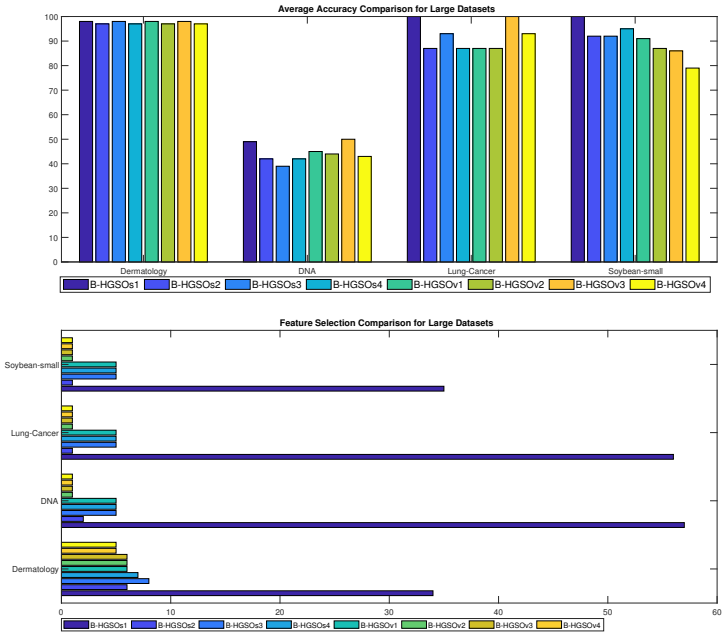


Figure 4.3: Large datasets comparison in term of average accuracy and number of selected features for the 8 different transfer functions.

deviation) for the eight binary methods.

From the comparison between the eight optimizers, the findings illustrate that the suggested B-HGSOs2 presents significant results in terms of mean fitness on eight benchmark datasets and is able to provide the minimum results on best, worst, and Std in some cases, it is clearly seen that the B-HGSOs2 was able to provide very competitive results.

Because the efficiency of any model depends mainly on the classification accuracy and the selected important features, specific values for α and β parameters are used for the fitness function in order to provide a balanced measure of the model's performance. Therefore, in all experiments, we put $\alpha = 0,5$ and $\beta = 0,5$ to reflect the balancing between classification quality and length of the subset. Here, we take into account the obtained results from B-HGSOs2 in terms of accuracy and the average number of selected features, where we clearly see that B-HGSOs2 ranked second in six benchmark datasets and achieved a maximum average number of features in eight benchmark datasets, this explains providing of the best results in terms of mean fitness as compared to all other versions.

The convergence graphs depicting the values of the fitness function versus the number of iterations for all eight versions of the suggested approach with S-shaped and V-shaped transfer functions are presented in Figure 4.4. It is evident that most approaches have converged between 20 and 40 iterations, highlighting the effectiveness of this approach in binary optimization algorithms in terms of convergence to the global optimum.

Table 4.7 outlines the average running time taken by each version of the B-HGSO to complete the feature selection process. It can be noticed that B-HGSO with V-shaped transfer functions takes the lowest average running time as compared to other optimizers that are based on S-shaped transfer functions in most cases. Also, the B-HGSOv4 takes the minimum average running time compared to other versions, which has the advantage of selecting a fewer number of features, so, it will require a short time to complete the feature selection process. In other words, this resulting the algorithm running faster.

Carefully selecting a relevant subset of features is necessary for accurate classification as reducing the number of features may remove important information necessary for classification tasks.

Table 4.7: Average computational time using 8 different transfer functions

	B-HGSOs1	B-HGSOs2	B-HGSOs3	B-HGSOs4	B-HGSOv1	B-HGSOv2	B-HGSOv3	B-HGSOv4
Breast-cancer	8,28	7,58	7,60	7,44	9,44	7,65	**7,10**	8,65
Dertermology	3,16	3,82	**2,99**	3,02	3,01	3,04	3,26	3,15
DNA	17,95	16,58	13,97	14,52	13,69	**13,01**	14,22	15,20
Exactly	6,42	6,09	5,16	5,60	5,72	5,11	4,92	**4,90**
Exactly2	5,87	5,32	7,59	6,73	5,93	5,10	6,12	**4,44**
HeartEW	1,65	1,73	1,72	1,69	1,46	1,51	1,56	**1,43**
Lung-Cancer	12,12	11,06	**11,02**	11,44	14,09	17,33	12,73	11,19
Lymphography	6,64	5,80	5,73	5,54	5,46	6,28	6,08	**5,43**
M-of-n	5,84	5,16	5,56	5,28	5,29	**4,94**	5,79	5,48
Mushroom	**20,40**	26,95	25,52	20,95	21,29	22,45	24,17	23,68
Primary-tumor	1,46	2,05	2,00	2,03	2,02	1,90	**1,37**	5,36
Soybean-small	8,23	10,35	8,29	10,79	9,14	10,69	**7,46**	9,76

Table 4.6: The obtained fitness values using 8 different transfer functions.

Dataset	B-HGSOs1	B-HGSOs2	B-HGSOs3	B-HGSOs4	B-HGSOv1	B-HGSOv2	B-HGSOv3	B-HGSOv4
Breast-cancer								
best	2,88E-02	2,54E-02	**2,36E-02**	2,93E-02	2,71E-02	2,65E-02	2,65E-02	2,42E-02
worst	4,45E-02	**4,40E-02**	**4,40E-02**	5,42E-02	4,63E-02	4,97E-02	4,97E-02	4,69E-02
mean	3,68E-02	**3,38E-02**	3,51E-02	3,72E-02	3,66E-02	3,95E-02	3,95E-02	3,75E-02
std.	**4,61E-03**	5,02E-03	5,13E-03	6,08E-03	5,82E-03	5,55E-03	5,55E-03	5,99E-03
Dermatology								
best	1,38E-02	1,09E-02	1,48E-02	1,12E-02	1,72E-02	**9,96E-03**	1,69E-02	1,42E-02
worst	4,69E-02	4,18E-02	**3,82E-02**	4,27E-02	5,29E-02	4,36E-02	4,93E-02	5,71E-02
mean	3,33E-02	**2,53E-02**	2,67E-02	2,99E-02	3,38E-02	3,02E-02	3,39E-02	3,96E-02
std.	9,10E-03	8,58E-03	**7,69E-03**	8,42E-03	9,29E-03	8,53E-03	7,87E-03	9,40E-03
DNA								
best	5,30E-01	5,07E-01	5,32E-01	4,87E-01	5,31E-01	**4,74E-01**	5,06E-01	5,12E-01
worst	**5,80E-01**	5,81E-01	6,06E-01	5,84E-01	6,02E-01	5,87E-01	**5,80E-01**	**5,80E-01**
mean	5,53E-01	5,54E-01	5,59E-01	5,48E-01	5,62E-01	**5,46E-01**	5,51E-01	5,51E-01
std.	**1,55E-02**	2,01E-02	1,74E-02	2,15E-02	1,71E-02	2,61E-02	2,12E-02	1,97E-02
Exactly								
best	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**
worst	3,22E-01	**6,16E-02**	1,85E-01	7,15E-02	3,21E-01	3,21E-01	3,12E-01	3,11E-01
mean	1,26E-01	**1,34E-02**	2,94E-02	2,05E-02	1,55E-01	1,75E-01	1,91E-01	2,36E-01
std.	1,30E-01	**1,51E-02**	5,28E-02	2,48E-02	1,32E-01	1,36E-01	1,23E-01	1,15E-01
Exactly2								
best	2,19E-01	2,19E-01	**2,09E-01**	2,21E-01	2,21E-01	2,13E-01	2,19E-01	**2,09E-01**
worst	2,99E-01	2,83E-01	2,99E-01	2,91E-01	2,83E-01	3,01E-01	**2,70E-01**	2,86E-01
mean	2,48E-01	**2,38E-01**	2,45E-01	2,48E-01	2,65E-01	2,50E-01	2,44E-01	2,41E-01
std.	2,18E-02	1,56E-02	2,41E-02	1,98E-02	**1,42E-02**	2,91E-02	3,64E-02	4,18E-02
HeartEW								
best	1,56E-01	**1,20E-01**	1,42E-01	1,58E-01	1,92E-01	1,34E-01	1,49E-01	1,78E-01
worst	2,37E-01	**2,16E-01**	2,20E-01	**2,16E-01**	2,36E-01	2,44E-01	2,40E-01	2,22E-01
mean	1,89E-01	**1,74E-01**	1,85E-01	1,87E-01	2,11E-01	1,90E-01	1,95E-01	2,02E-01
std.	2,18E-02	2,52E-02	2,14E-02	**1,54E-02**	1,84E-02	2,74E-02	2,45E-02	1,78E-02
Lung-Cancer								
best	**1,79E-04**	1,07E-03	1,43E-03	1,25E-03	3,57E-04	3,57E-04	3,57E-04	7,14E-04
worst	1,26E-01	1,26E-01	1,87E-01	1,26E-01	**1,24E-01**	**1,24E-01**	1,86E-01	**1,24E-01**
mean	7,19E-02	**4,20E-02**	5,79E-02	5,47E-02	4,86E-02	4,86E-02	7,48E-02	5,57E-02
std.	4,62E-02	**3,62E-02**	5,60E-02	4,58E-02	4,11E-02	4,11E-02	4,75E-02	3,71E-02
Lymphography								
best	7,10E-02	9,75E-02	8,30E-02	**7,02E-02**	1,23E-01	9,70E-02	1,23E-01	1,40E-01
worst	2,03E-01	**1,90E-01**	1,92E-01	2,19E-01	2,29E-01	2,06E-01	2,31E-01	2,30E-01
mean	1,53E-01	**1,37E-01**	1,42E-01	1,42E-01	1,71E-01	1,44E-01	1,66E-01	1,88E-01
std.	3,37E-02	2,57E-02	3,04E-02	3,36E-02	2,79E-02	3,27E-02	3,01E-02	**2,23E-02**
M-of-n								
best	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**
worst	1,61E-01	**2,79E-02**	5,57E-02	3,39E-02	1,60E-01	1,72E-01	1,70E-01	1,78E-01
mean	6,64E-02	**8,21E-03**	1,51E-02	8,31E-03	7,94E-02	9,17E-02	6,41E-02	1,03E-01
std.	6,31E-02	7,90E-03	1,65E-02	**6,98E-03**	6,70E-02	6,71E-02	6,52E-02	6,34E-02
Mushroom								
best	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**
worst	**2,31E-03**	2,73E-03	2,73E-03	2,73E-03	2,52E-03	2,52E-03	2,55E-03	3,18E-03
mean	2,02E-03	**1,96E-03**	2,18E-03	2,22E-03	2,36E-03	1,99E-03	2,02E-03	2,14E-03
std.	2,34E-04	**2,29E-04**	2,38E-04	2,78E-04	2,38E-04	2,47E-04	2,55E-04	4,20E-04
Primary-tumor								
best	4,12E-01	5,34E-01	3,48E-01	**2,98E-01**	5,66E-01	3,82E-01	5,60E-01	6,16E-01
worst	6,46E-01	**6,37E-01**	6,79E-01	6,60E-01	6,66E-01	6,68E-01	6,77E-01	6,53E-01
mean	**5,90E-01**	6,00E-01	6,01E-01	6,04E-01	6,21E-01	6,20E-01	6,31E-01	6,34E-01
std.	6,01E-02	2,50E-02	6,45E-02	7,53E-02	**2,41E-02**	5,93E-02	3,04E-02	2,62E-02
Soybean-small								
best	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**
worst	**5,71E-04**	1,43E-03	1,43E-03	1,14E-03	**5,71E-04**	8,57E-04	**5,71E-04**	**5,71E-04**
mean	**5,71E-04**	9,86E-04	9,86E-04	9,00E-04	**5,71E-04**	5,86E-04	**5,71E-04**	**5,71E-04**
std.	**1,11E-19**	2,85E-04	2,17E-04	1,68E-04	**1,11E-19**	6,39E-05	**1,11E-19**	**1,11E-19**

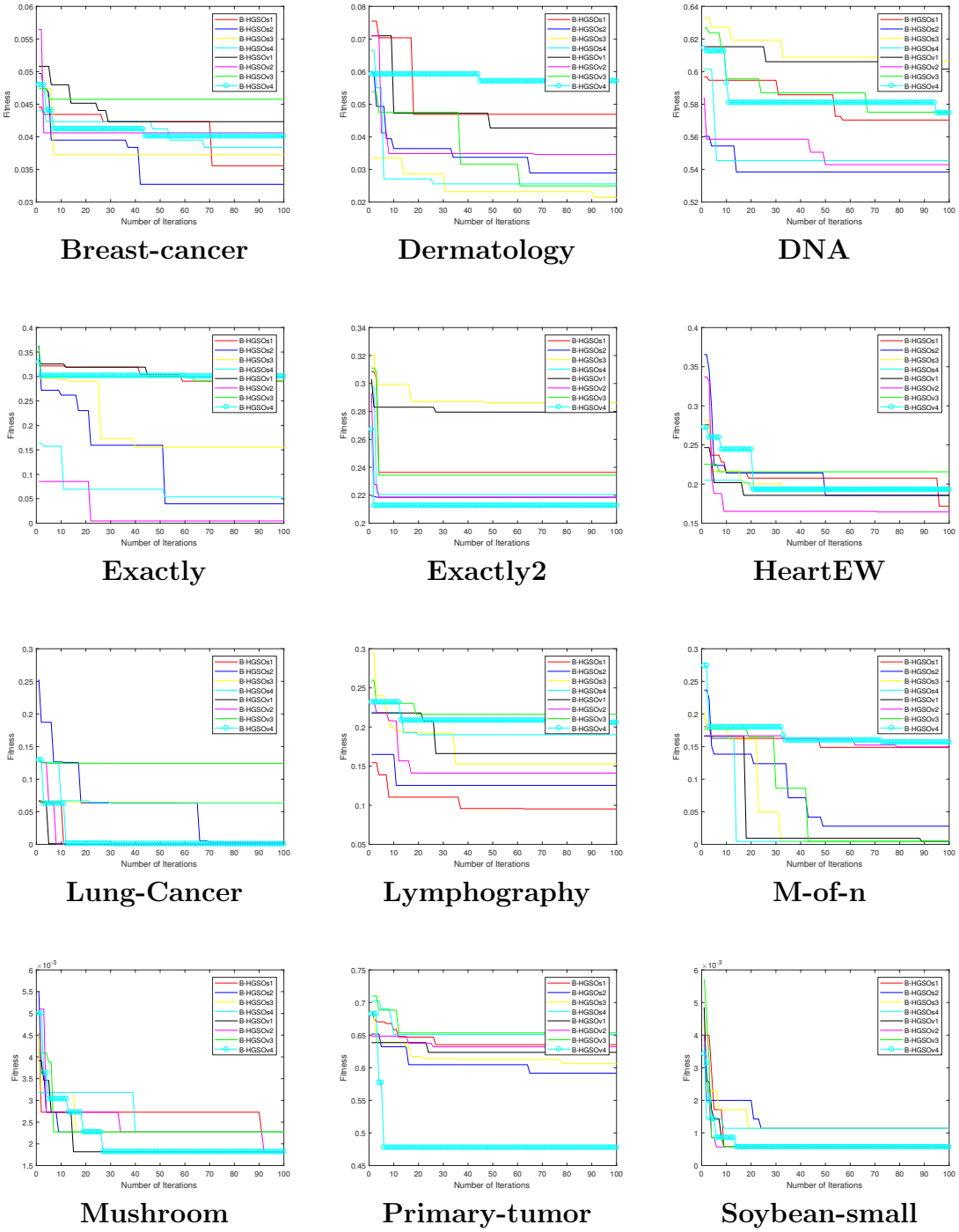


Figure 4.4: Convergence curves of the eight versions of B-HGSO for the benchmark datasets.

In summary, the B-HGSO wrapper feature selection method can be effective in selecting optimal subsets, and its effectiveness in solving feature selection remains an active area of research. It still needs further research to establish its effectiveness in solving feature selection problems due to its sensitivity to adjusting parameters that can help to achieve optimal performance for the B-HGSO algorithm.

4.3.4 Comparison with other metaheuristics

In this subsection, a comparison among the best-performing methods (B-HGSOs1, B-HGSOs2, and B-HGSOv4) with four population-based metaheuristic algorithms that are based on swarm intelligence is presented, these algorithms are commonly used in the literature as benchmarking techniques in order to achieve a meaningful comparison with new algorithms that are:

- The GWO algorithm is inspired by the hunting behavior of grey wolves, and it is known for its efficient and effective optimization abilities. For wrapper feature selection problems, B-GWO is a promising method thanks to using three main operators: searching, encircling, and attacking;
- The B-PSO wrapper feature selection method inherits the PSO algorithm's concepts to search for the optimal subset of features that maximizes the performance of a model;
- The MRFO algorithm is inspired by the foraging behavior of manta rays, which are known for their efficient search for food in the ocean. B-MRFO algorithm has several advantages for wrapper feature selection, it can handle discrete variables and has been shown to perform well by applying three main operators: search for prey, evaluate prey, and update foraging behavior;
- Whale Optimization Algorithm (WOA) is a metaheuristic algorithm inspired by the hunting behavior of humpback whales, which is a promising method for wrapper feature selection by applying exploration, exploitation, and selection operators.

The parameters play a crucial role in balancing exploration and exploitation in optimization algorithms. In our study, the parameters used by competitor algorithms were adopted as per their developers' descriptions. The settings for the parameters are presented in Table 4.8. To ensure fairness in comparison, the optimizers are performed under the same conditions, we set the number of iterations to 100, the population size to 10, and k in k-NN to 5, $\alpha = 0,5$ and $\beta = 0,5$ parameters in the fitness function. The results are based on 20 independent runs across twelve different datasets.

Table 4.8: Parameters setting for experiments

Algorithm	**Parameters**
B-GWO	Number of wolves: $N = 10$ Maximum number of iterations:100 Dimension corresponds to the number of features. a decreases linearly from 2 to 0
B-PSO	Population size: $N = 10$ Maximum number of iterations:100 Dimension corresponds to the number of features $Wmax = 0.9$, $Wmin = 0.4$ and $Vmax = 0.6$ $c1 = c2 = 2$
B-MRFO	Population size: $N = 10$ Maximum number of iterations:100 Dimension corresponds to the number of features $S = 2$
B-WOA	Population size: $N = 10$ Maximum number of iterations:100 Dimension corresponds to the number of features $b = 1$

This work presents a comparison of the performance of various metaheuristic optimizers on selected datasets. The findings for the classification accuracy are shown in Table 4.9, expressed as percentages with their standard deviations obtained from the optimizers. The best results are highlighted in bold text, and the results indicate that the suggested B-HGSOS1 outperforms the other optimizers in all datasets except for Primary-tumor data, where it ranked second with highly competitive results. From this table, we can clearly see that the B-PSO performs better in five datasets. It is interesting to note that the suggested B-HGSOs2 is slightly superior, ranking second or third in most datasets with respect to other optimizers.

Table 4.9: Classification accuracy values obtained from the competitor optimizers

Dataset	B-GWO	B-PSO	B-MRFO	B-WOA	B-HGSOs1	B-HGSOs2	B-HGSOv4
Breast-cancer							
mean (%)	95,43	97,19	96,00	94,57	**97,43**	96,29	91,73
std.	6,10E-03	6,24E-03	**5,93E-03**	6,84E-03	2,47E-02	7,59E-03	5,56E-02
Dermatology							
mean (%)	96,72	**98,36**	96,17	95,08	**98,36**	97,27	97,27
std.	7,58E-03	**4,00E-03**	7,62E-03	1,01E-02	8,81E-02	2,48E-02	1,68E-01
DNA							
mean (%)	44,03	47,20	40,88	37,74	**49,06**	42,14	33,43
std.	2,30E-02	**2,01E-02**	2,70E-02	2,91E-02	3,80E-02	3,81E-02	5,48E-02
Exactly							
mean (%)	96,03	**100,00**	78,80	78,98	**100,00**	82,00	62,50
std.	1,00E-01	**0,00E+00**	6,60E-02	1,16E-01	**0,00E+00**	5,94E-02	1,10E-01
Exactly2							
mean (%)	74,13	71,80	75,15	72,63	**78,00**	75,40	66,13
std.	2,23E-02	1,85E-02	**1,58E-02**	3,27E-02	4,93E-02	3,99E-02	1,23E-01
HeartEW							
mean (%)	76,30	79,26	80,37	77,48	**84,44**	73,33	79,26
std.	2,32E-02	**2,05E-02**	2,71E-02	3,59E-02	7,75E-02	6,30E-02	1,53E-01
Lung-Cancer							
mean (%)	87,50	93,75	94,37	89,69	**100,00**	87,50	71,53
std.	3,99E-02	3,19E-02	5,33E-02	6,18E-02	**0,00E+00**	9,64E-02	2,40E-01
Lymphography							
mean (%)	81,08	88,26	85,10	80,85	**93,06**	81,08	70,22
std.	3,12E-02	**2,85E-02**	3,10E-02	4,06E-02	4,79E-02	5,75E-02	8,64E-02
M-of-n							
mean (%)	98,60	**100,00**	97,94	90,97	**100,00**	99,00	70,40
std.	3,07E-03	**0,00E+00**	4,07E-02	6,53E-02	**0,00E+00**	8,77E-02	1,03E-01
Mushroom							
mean (%)	99,90	**100,00**	**100,00**	99,88	**100,00**	99,33	82,42
std.	3,04E-04	**0,00E+00**	**0,00E+00**	3,13E-04	**0,00E+00**	1,32E-02	1,65E-01
Primary-tumor							
mean (%)	**64,15**	41,34	40,31	36,43	58,49	36,47	34,34
std.	6,10E-02	**2,52E-02**	9,59E-02	3,80E-02	8,57E-02	4,89E-02	8,71E-02
Soybean-small							
mean (%)	**100,00**	**100,00**	**100,00**	**100,00**	**100,00**	92,08	79,37
std.	**0,00E+00**	**0,00E+00**	**0,00E+00**	**0,00E+00**	**0,00E+00**	9,36E-02	2,36E-01

Table 4.10 presents the average number of selected features by the different optimizers. Among these methods, B-HGSOv4 achieved the lowest number of features across the eleven selected datasets. In addition, in Figures 4.5, 4.6 and 4.7 we can see the bar graph for the mean accuracy and the average number of selected features. From these figures, it can be observed the superior performance of the B-HGSOs1 and B-HGSOv4 compared to all other optimizers in terms of classification accuracy and reducing the dimensionality of the data, respectively.

The fitness function is designed by considering both the classification accuracy and the length of the subset, Table 4.11 outlines the fitness values obtained over the 20 runs. The comparison of the different optimizers reveals that B-HGSOs2 produces significant results in terms of mean fitness on six benchmark datasets and is able of providing the minimum results on best, worst, and Std in some cases. However, the results also demonstrate that B-GWO and B-PSO are able of providing very competitive results.

The convergence graph is an essential visualization tool to evaluate the performance of the optimizers by depicting the values of the fitness function versus the number of iterations during the optimization process. Figure 4.8 shows the convergence curves of the optimizers on various selected datasets, it is worth mentioning that 100 iterations were enough for the optimizers to converge toward the optimal solution. The behavior

Table 4.10: Average Of Selected Features Comparison between the competitor optimizers

Dataset	B-GWO	B-PSO	B-MRFO	B-WOA	B-HGSOs1	B-HGSOs2	B-HGSOv4
Breast-cancer	5,38	5,16	5,35	5,38	4,50	6,39	**2,61**
Dermatology	**4,12**	5,07	5,40	5,03	6,47	7,94	5,29
DNA	2,74	4,65	4,49	3,50	2,02	5,67	**1,27**
Exactly	5,00	4,61	5,64	4,90	5,04	6,08	**3,04**
Exactly2	3,19	2,92	1,91	2,54	**1,65**	4,80	2,05
HeartEW	6,15	5,38	3,61	**2,52**	3,84	7,69	3,08
Lung-Cancer	1,47	3,48	2,31	1,45	0,69	5,77	**0,50**
Lymphography	4,11	4,25	4,32	4,33	2,83	2,83	**2,22**
M-of-n	5,50	4,61	5,59	5,47	5,27	6,23	**3,19**
Mushroom	2,43	2,16	2,84	2,44	1,93	5,66	**1,34**
Primary-tumor	7,64	7,64	5,66	5,38	7,22	8,33	**3,88**
Soybean-small	1,27	1,67	1,82	1,26	0,54	5,49	**0,53**

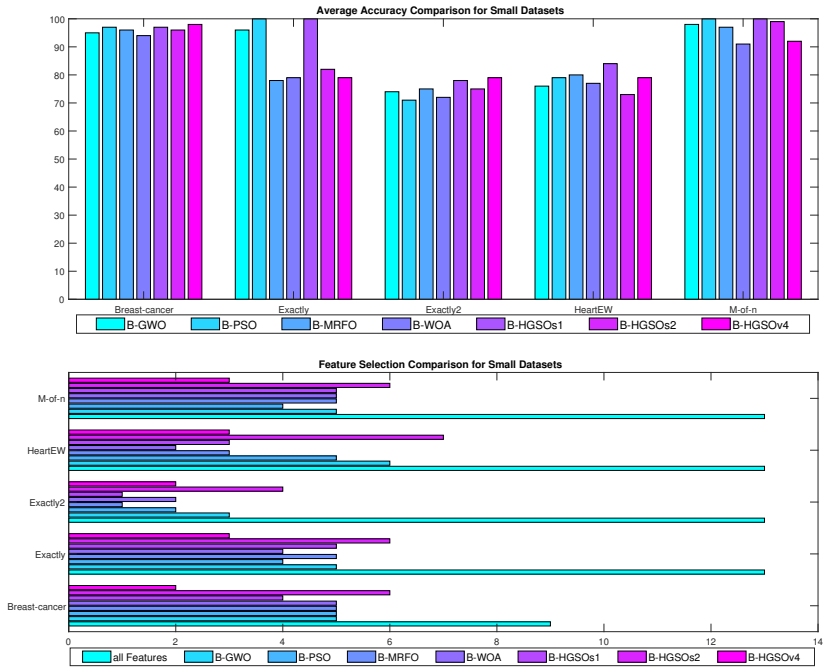


Figure 4.5: Small datasets comparison in term of average accuracy and number of selected features for the competitor algorithms.

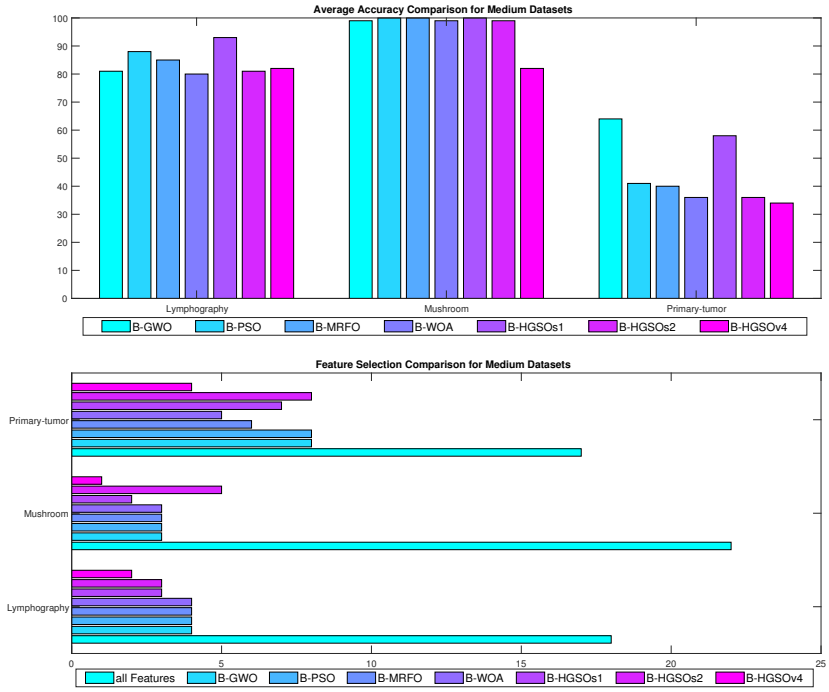


Figure 4.6: Medium datasets comparison in term of average accuracy and number of selected features for the competitor algorithms.

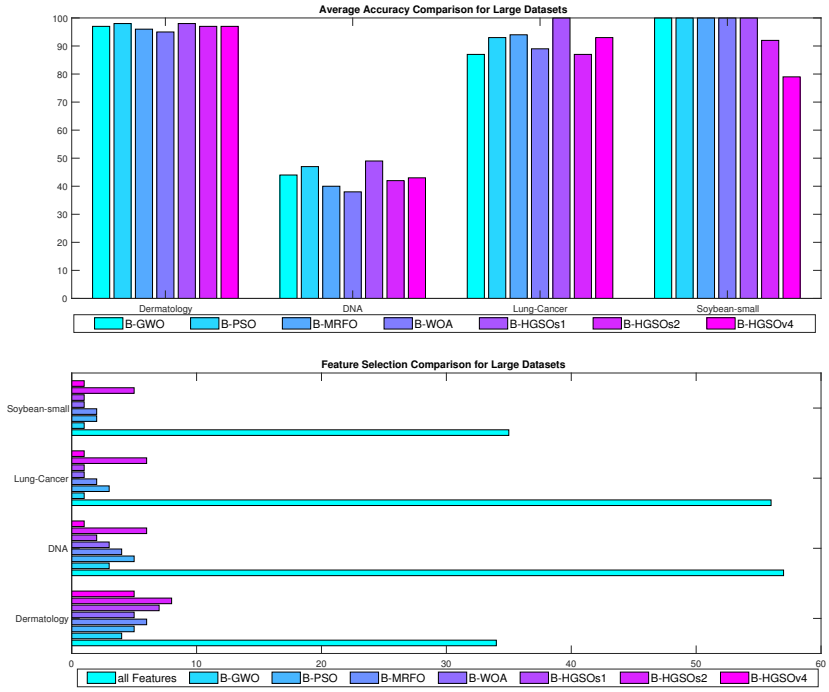


Figure 4.7: Large datasets comparison in term of average accuracy and number of selected features for the competitor algorithms.

of the optimizer is influenced by the specific problem being solved and the characteristics of the dataset. The decreasing trend in the fitness values of the competitor optimizers suggests that these optimizers are making progress to find the optimal feature subset. The convergence curves of the suggested methods indicate a fast convergence in most datasets, but a slow convergence in others, depending on the characteristics of each dataset.

Table 4.11: The obtained fitness values from the competitor optimizers.

Dataset	B-GWO	B-PSO	B-MRFO	B-WOA	B-HGSOs1	B-HGSOs2	B-HGSOv4
Breast-cancer							
best	**2,31E-02**	3,36E-02	2,65E-02	3,21E-02	2,88E-02	2,54E-02	2,42E-02
worst	4,52E-02	4,52E-02	4,52E-02	5,93E-02	4,45E-02	**4,40E-02**	4,69E-02
mean	3,57E-02	4,30E-02	3,66E-02	4,27E-02	3,68E-02	**3,38E-02**	3,75E-02
std.	5,82E-03	5,73E-03	5,48E-03	6,18E-03	**4,61E-03**	5,02E-03	5,99E-03
Dermatology							
best	**8,44E-03**	1,06E-02	1,60E-02	2,05E-02	1,38E-02	1,09E-02	1,42E-02
worst	3,64E-02	**2,86E-02**	4,30E-02	5,32E-02	4,69E-02	4,18E-02	5,71E-02
mean	2,63E-02	2,75E-02	2,77E-02	3,57E-02	3,33E-02	**2,53E-02**	3,96E-02
std.	7,76E-03	**3,85E-03**	7,46E-03	1,01E-02	9,10E-03	8,58E-03	9,40E-03
DNA							
best	**4,45E-01**	4,98E-01	4,85E-01	5,14E-01	5,30E-01	5,07E-01	5,12E-01
worst	**5,44E-01**	5,59E-01	5,91E-01	6,25E-01	5,80E-01	5,81E-01	5,80E-01
mean	**4,83E-01**	5,27E-01	5,31E-01	5,56E-01	5,53E-01	5,54E-01	5,51E-01
std.	2,25E-02	1,99E-02	2,66E-02	2,94E-02	**1,55E-02**	2,01E-02	1,97E-02
Exactly							
best	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**
worst	3,40E-01	**4,62E-03**	2,18E-01	3,36E-01	3,22E-01	6,16E-02	3,11E-01
mean	3,68E-02	**4,62E-03**	5,37E-02	2,13E-01	1,26E-01	1,34E-02	2,36E-01
std.	9,90E-02	**1,78E-18**	6,62E-02	1,14E-01	1,30E-01	1,51E-02	1,15E-01
Exactly2							
best	2,23E-01	2,17E-01	2,15E-01	2,23E-01	2,19E-01	2,19E-01	**2,09E-01**
worst	2,82E-01	2,83E-01	**2,71E-01**	3,31E-01	2,99E-01	2,83E-01	2,86E-01
mean	2,53E-01	2,53E-01	2,47E-01	2,74E-01	2,48E-01	**2,38E-01**	2,41E-01
std.	1,78E-02	2,01E-02	**1,61E-02**	3,43E-02	2,18E-02	1,56E-02	4,18E-02
HeartEW							
best	1,35E-01	1,37E-01	1,44E-01	1,64E-01	1,56E-01	**1,20E-01**	1,78E-01
worst	2,25E-01	**2,08E-01**	2,44E-01	2,80E-01	2,37E-01	2,16E-01	2,22E-01
mean	1,75E-01	1,77E-01	1,98E-01	2,25E-01	1,89E-01	**1,74E-01**	2,02E-01
std.	2,23E-02	2,02E-02	2,65E-02	3,55E-02	2,18E-02	2,52E-02	**1,78E-02**
Lung-Cancer							
best	7,14E-04	2,86E-03	5,36E-04	5,36E-04	**1,79E-04**	1,07E-03	7,14E-04
worst	1,25E-01	**6,58E-02**	1,88E-01	1,88E-01	1,26E-01	1,26E-01	1,24E-01
mean	4,75E-02	**3,13E-02**	5,76E-02	1,03E-01	7,19E-02	4,20E-02	5,57E-02
std.	3,95E-02	**3,15E-02**	5,24E-02	6,10E-02	4,62E-02	3,62E-02	3,71E-02
Lymphography							
best	7,02E-02	**6,00E-02**	9,64E-02	1,25E-01	7,10E-02	9,75E-02	1,40E-01
worst	1,93E-01	1,96E-01	2,06E-01	2,98E-01	2,03E-01	**1,90E-01**	2,30E-01
mean	1,72E-01	1,50E-01	1,52E-01	1,94E-01	1,53E-01	**1,37E-01**	1,88E-01
std.	3,07E-02	2,80E-02	3,09E-02	3,96E-02	3,37E-02	2,57E-02	**2,23E-02**
M-of-n							
best	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**	**4,62E-03**
worst	**4,62E-03**	**4,62E-03**	1,47E-01	1,73E-01	1,61E-01	2,79E-02	1,78E-01
mean	**4,62E-03**	**4,62E-03**	2,59E-02	9,53E-02	6,64E-02	8,21E-03	1,03E-01
std.	**1,78E-18**	**1,78E-18**	4,13E-02	6,48E-02	6,31E-02	7,90E-03	6,34E-02
Mushroom							
best	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**	**1,82E-03**
worst	**2,27E-03**	**2,27E-03**	3,18E-03	3,64E-03	2,31E-03	2,73E-03	3,18E-03
mean	2,05E-03	2,16E-03	2,23E-03	2,85E-03	2,02E-03	**1,96E-03**	2,14E-03
std.	2,28E-04	**2,02E-04**	3,58E-04	5,40E-04	2,34E-04	2,29E-04	4,20E-04
Primary-tumor							
best	3,39E-01	5,52E-01	**2,13E-01**	5,76E-01	4,12E-01	5,34E-01	6,16E-01
worst	6,43E-01	6,59E-01	6,79E-01	7,10E-01	6,46E-01	**6,37E-01**	6,53E-01
mean	**5,84E-01**	5,86E-01	5,97E-01	6,35E-01	5,90E-01	6,00E-01	6,34E-01
std.	6,33E-02	2,53E-02	9,60E-02	3,76E-02	6,01E-02	**2,50E-02**	2,62E-02
Soybean-small							
best	**5,71E-04**	1,14E-03	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**	**5,71E-04**
worst	1,14E-03	2,57E-02	2,57E-03	2,86E-03	**5,71E-04**	1,43E-03	**5,71E-04**
mean	7,00E-04	1,67E-03	8,71E-04	1,13E-03	**5,71E-04**	9,86E-04	**5,71E-04**
std.	1,73E-04	2,82E-04	4,68E-04	5,82E-04	**1,11E-19**	2,85E-04	**1,11E-19**

Table 4.12 shows that B-WOA is the fastest method for the majority of benchmark datasets, outperforming all other algorithms in six out of twelve datasets. Our algorithm B-HGSOv4 ranks second with the best time in five datasets and competitive execution

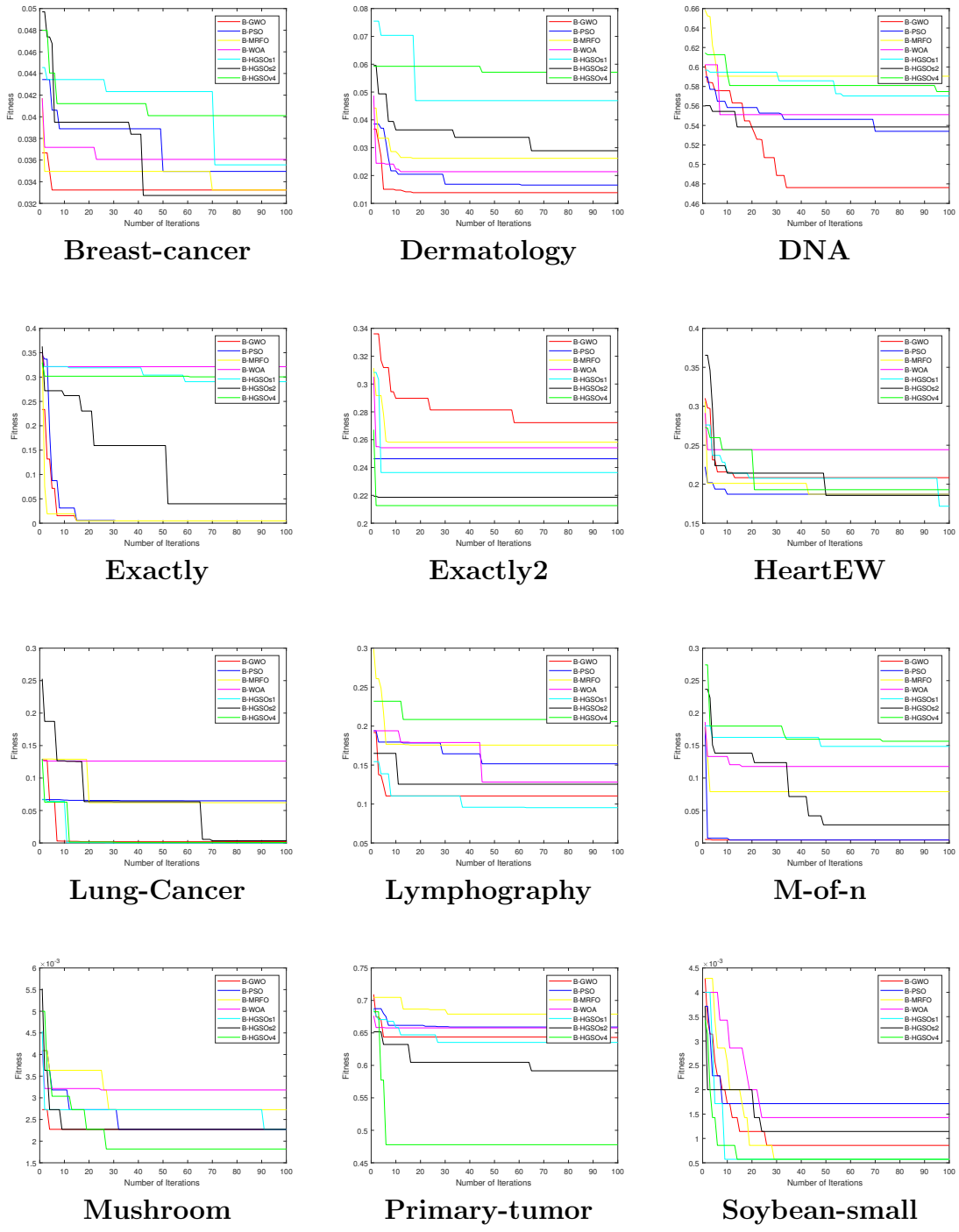


Figure 4.8: Convergence curves of the competitor algorithms for the datasets.

Table 4.12: Comparison between the competitor optimizers in term of computational time

Dataset	B-GWO	B-PSO	B-MRFO	B-WOA	B-HGSOs1	B-HGSOs2	B-HGSOv4
Breast-cancer	10,29	5,99	7,46	**4,83**	8,28	7,58	8,65
Dermatology	6,55	5,86	8,84	6,09	3,16	3,82	**3,15**
DNA	10,10	14,28	10,04	**6,85**	17,95	16,58	15,20
Exactly	6,73	6,81	10,21	6,41	6,42	6,09	**4,90**
Exactly2	8,93	6,53	7,78	5,30	5,87	5,32	**4,44**
HeartEW	4,50	4,97	10,07	4,56	1,65	1,73	**1,43**
Lung-Cancer	8,09	7,53	8,93	**6,96**	12,12	11,06	11,19
Lymphography	9,08	5,85	7,4	**4,83**	6,64	5,80	5,43
M-of-n	8,34	7,79	10,41	6,54	5,84	5,86	**5,48**
Mushroom	24,42	22,47	27,47	**19,35**	20,40	26,95	23,68
Primary-tumor	6,05	6,50	9,38	6,25	**1,46**	2,05	5,36
Soybean-small	7,88	8,68	7,48	**6,28**	8,23	10,35	9,76

times. Note that, B-HGSOv4 achieved the lowest number of features across the eleven selected datasets, leading to increased speed in the execution of this method. Furthermore, our suggested B-HGSOs1 excels in one dataset (Primary-tumor) and ranks second in four others. B-HGSOs2, also proposed in this study, ranks second in one dataset (Primary-tumor) and achieves highly competitive results, ranking third in five datasets.

In this study, the Wilcoxon nonparametric test[36] is employed to evaluate the efficacy of the best-performing methods (B-HGSOs1, B-HGSOs2, and B-HGSOv4) and determine if they yield significant improvements over the selected algorithms. In our study, we utilize the Wilcoxon test at a 95% confidence level i.e, the two algorithms have similar performance if the p-value is greater than 0,05. Otherwise, they have a significant difference.

As mentioned above, B-HGSOs1 demonstrated superior classification accuracy, while B-HGSOv4 achieved the best results in reducing data dimensionality. To check the accuracy and dimensionality reduction capabilities of both algorithms, we conducted pairwise comparisons using the Wilcoxon test across 20 independent runs. The results of these comparisons are outlined in Tables 4.13 and 4.14, respectively. These findings indicate that B-HGSOs1 outperforms its competitors in the majority of the selected datasets. Similarly, B-HGSOv4 excels in reducing the dimensionality of all selected datasets, as demonstrated by its significant performance improvements versus its competitors, except for one dataset (Exactly2) which indicates similar performance for the algorithms.

Also, the effectiveness of the proposed B-HGSOs2 algorithm and its strength to provide a balanced measure of the model's performance is checked by using the Wilcoxon test. Findings are provided in Table4.15, we can observe that our approach achieves significantly better results in 22 cases.

Actually, the performance of an algorithm consists on the specific problem being addressed and the dataset properties, therefore, adjusting the parameter values is necessary to achieve optimal performance. The HGSO algorithm includes several parameters, which

lead to long computation times and make the algorithm impractical for some applications. The specific controlling parameters to use would depend on the particular dataset and the problem being solved.

Table 4.13: The p values of Wilcoxon test for Mean accuracy

B-HGSOs1 vs				
	B-GWO	B-PSO	B-MRFO	B-WOA
Breast-cancer	3,87E-01	**3,63E-04**	**1,28E-02**	6,44E-01
Dermatology	**4,18E-07**	**8,42E-08**	**6,74E-05**	**1,07E-02**
DNA	**9,90E-08**	**1,35E-06**	**6,08E-06**	**6,05E-03**
Exactly	**5,74E-03**	5,52E-01	**1,65E-03**	**9,84E-07**
Exactly2	5,16E-01	3,78E-01	2,32E-01	8,25E-01
HeartEW	1,21E-01	**6,17E-03**	4,03E-01	9,34E-01
Lung-Cancer	**7,31E-05**	**4,11E-06**	**3,51E-04**	5,35E-02
Lymphography	**9,92E-04**	**1,95E-05**	**7,47E-03**	7,21E-01
M-of-n	**8,08E-05**	1,75E-01	**5,71E-05**	**1,72E-07**
Mushroom	**5,69E-04**	**2,01E-02**	**2,01E-02**	**1,57E-03**
Primary-tumor	1,03E-01	**2,11E-03**	3,00E-01	7,82E-01
Soybean-small	1,04E-01	1,04E-01	1,04E-01	1,04E-01

Table 4.14: The p values of Wilcoxon test for Average Of Selected Features

B-HGSOv4 vs				
	B-GWO	B-PSO	B-MRFO	B-WOA
Breast-cancer	**1,48E-06**	**8,48E-07**	**3,45E-07**	**2,09E-06**
Dermatology	**5,47E-05**	**1,13E-06**	**1,01E-06**	**2,31E-05**
DNA	**5,20E-07**	**3,24E-08**	**6,09E-08**	**1,94E-07**
Exactly	**7,36E-05**	**1,74E-05**	**2,03E-05**	**4,75E-03**
Exactly2	1,44E-01	5,00E-01	6,59E-01	4,82E-01
HeartEW	**1,03E-03**	**5,26E-04**	**1,05E-03**	**1,53E-02**
Lung-Cancer	**6,42E-05**	**1,11E-05**	**7,30E-05**	**7,25E-04**
Lymphography	**2,99E-06**	**3,01E-06**	**1,41E-05**	**6,28E-05**
M-of-n	**5,51E-07**	**5,63E-06**	**2,38E-06**	**9,93E-05**
Mushroom	**2,04E-06**	**1,33E-06**	**1,59E-06**	**1,52E-05**
Primary-tumor	**1,66E-02**	**1,61E-02**	**2,98E-02**	**2,97E-02**
Soybean-small	**3,64E-08**	**1,83E-08**	**1,03E-07**	**3,21E-06**

The statistical results obtained in this study demonstrate the efficiency of the proposed b-HGSO algorithm. This effectiveness is attributed to three key factors:

- The adoption of a multi-population concept in the original HGSO algorithm enhances its capability to explore the feature space, increasing the chance of identifying high-quality solutions.
- The algorithm dynamically updates important parameters, such as $S_{(i,j)}$, γ, and F, to maintain a balance between exploration and exploitation.

Table 4.15: The p values of Wilcoxon test for the fitness

B-HGSOs2 vs				
	B-GWO	B-PSO	B-MRFO	B-WOA
Breast-cancer	1,31E-01	**4,38E-02**	**4,94E-02**	**2,89E-05**
Dermatology	9,99E-01	9,98E-01	1,79E-01	**1,46E-03**
DNA	1,00E+00	1,00E+00	9,98E-01	4,52E-01
Exactly	9,91E-01	1,00E+00	**9,77E-03**	**9,04E-05**
Exactly2	**2,40E-02**	**3,59E-02**	6,81E-02	**1,51E-03**
HeartEW	8,35E-01	7,38E-01	**2,19E-02**	**1,97E-04**
Lung-Cancer	9,85E-01	4,25E-01	4,84E-01	**4,93E-02**
Lymphography	**8,38E-03**	**9,86E-03**	**1,72E-03**	**4,89E-05**
M-of-n	1,00E+00	1,00E+00	7,36E-02	**1,53E-06**
Mushroom	1,00E+00	**4,96E-02**	**4,56E-02**	**1,37E-03**
Primary-tumor	7,91E-01	9,83E-01	9,71E-02	**2,35E-03**
Soybean-small	9,99E-01	**1,38E-07**	9,72E-01	2,85E-01

- The HGSO algorithm retains multiple solutions as the optimal feature subset, ensuring that all useful and relevant features are included in the final subset.

These characteristics contribute significantly to the efficiency and effectiveness of the b-HGSO algorithm in addressing feature selection tasks. Although B-HGSO has shown to be effective in prior discussions, it also possesses certain limitations. The algorithm requires a significant number of function evaluations to attain convergence towards an optimal solution, which may result in high costs as the model's complexity and the number of features increase. Furthermore, B-HGSO includes several controlling parameters, which may restrict its applicability in comparison to other similar approaches.

The study concludes that S-shaped transfer functions are generally more effective for improving classification accuracy, while V-shaped transfer functions demonstrate advantages in reducing computational complexity and the number of selected features. The balance between exploration and exploitation is necessary, and the choice of transfer function significantly impacts the convergence behavior and overall effectiveness of the B-HGSO algorithm. In summary, while S-shaped functions enhance accuracy, V-shaped functions improve efficiency, and the best performance is context-dependent based on the specific objective of the feature selection task.

Contrary to the findings of several studies employing different binary optimization algorithms, our research indicates that S-shaped transfer functions generally outperform V-shaped transfer functions in feature selection tasks. Ghosh et al. [53] investigated the impact of S-shaped and V-shaped transfer functions on a binary version of Manta Ray Foraging Optimization (MRFO) for feature selection. Their findings revealed that binary MRFO with V-shaped transfer functions outperformed existing approaches in terms of classification accuracy and feature selection efficiency. In [93],the obtained results showed that the V-shaped transfer functions outperformed the S-shaped ones in terms of con-

vergence speed and solution quality in the comparison between S-shaped and V-shaped transfer functions for binary Particle Swarm Optimization (BPSO). Mafarja et al. [84] examined the behavior of binary Ant Lion Optimizer (BALO) with S-shaped and V-shaped transfer functions for feature selection. Their findings indicated that BALO with V-shaped transfer functions effectively improved performance in avoiding local minima and achieved superior accuracy compared to the original ALO.

4.4 Conclusion

The experimental results consistently demonstrate that B-HGSO remains highly competitive in the field of feature selection, achieving interesting results across various datasets in comparison to other similar algorithms. While B-HGSO has shown promising results, there is still potential for enhancing its performance by incorporating additional mechanisms and more efficient techniques and exploring its performance in feature selection problems and also in conjunction with more efficient and powerful classifiers such as support vector machine (SVM).

To further investigate the impact of transfer functions on the proposed binary HGSO algorithm, we conducted a comprehensive analysis using multi-objective optimization in the next chapter. Given the conflicting results obtained in the single-objective setting, we hypothesized that a multi-objective approach might provide a more nuanced understanding of the influence of transfer functions.

In the next chapter, we plan to evaluate the proposed algorithm's performance by investigating the use of multi-objective optimization for generating diverse and high-performing feature subsets. This approach extends the B-HGSO approach to a multi-objective framework, termed the B-MOHGSO algorithm.

CHAPTER 5

Binary Multi-objective Henry Gas Solubility Optimization Algorithm in Feature Selection problem

5.1 Introduction

Eight binary versions of the multi-objective Henry Gas Solubility Optimization (MOHGSO) called B-MOHGSO for wrapper multi-objective feature selection task are proposed. The MOHGSO is a physical-based, global optimization approach originally designed for continuous optimization problems. This work focuses on evaluating the impact of the S-shaped and V-shaped transfer functions on the performance of the MOHGSO algorithm that optimizes two objectives: maximizing the accuracy and minimizing the number of selected features. The effectiveness of the proposed algorithms was assessed on twelve UCI Machine Learning Repository benchmark datasets.

5.2 B-MOHGSO implementation for feature selection

In order explore the use of eight transfer functions divided into S-shaped and V-shaped families and evaluate their performance in binary multi-objective Henry Gas Solubility Optimization (B-MOHGSO) algorithm to meet the specific requirements of feature selection, the following aspects have been adapted:

5.2.1 Individual representation

Feature selection is addressed as a discrete problem, thus the binary representation can be adopted, i.e., each individual's position that represents a subset of features is assigned as "1" or "0", which indicates whether the feature is chosen into the feature subset or not for evaluation, and the length of the individual's dimension corresponds to the total number of features in the dataset. Thus, in B-MOHGSO the $i-th$ individual's position is an n-dimensional vector that would be coded as an n-bit string:
$X_i = (x_1, x_2, ..., x_j, ..., x_n)$
In discrete binary search space, the solution is restricted to the binary $\{0, 1\}$ values. Accordingly, the transfer function is used to convert the positions of solutions from the continuous domain to the binary domain 4.2.1.1.

5.2.2 Objective Functions

The two objective functions are defined and formulated to treat the multi-objective feature selection problem as follows:

$$\begin{cases} \text{Maximize } f_1(x) = \dfrac{TP + TN}{TP + TN + FP + FN} \\ \text{Minimize } f_2(x) = \dfrac{M}{N} \\ \text{Subject to } x_i \in \{0, 1\} \end{cases} \rightarrow \begin{cases} \text{Maximize } f_1(x) \\ \text{Maximize } 1 - f_2(x) \\ \text{Subject to } x_i \in \{0, 1\} \end{cases} \tag{5.2.1}$$

where M is the chosen features and N represents the entire features of a dataset. TP and TN correspond to true positives and true negatives, while FP and FN denote false positives and false negatives, respectively.

- (f_1) measures the classification performance (the accuracy) of the selected feature subset using a learning classifier;
- (f_2) identifies Pareto-optimal feature subsets that result in a minimum of redundancy and maximum relevance of features to reduce the dimensionality of the dataset.

5.2.3 Wrapper Procedure

In the wrapper procedure, the performance of different feature subsets is evaluated using a chosen classification model, which has to be taken into account as the second objective function to assess the quality of the selected feature subset. In this work, the K-Nearest Neighbor (k-NN) algorithm is employed as a learning classifier to maximize classification accuracy. A higher classification accuracy suggests that the selected features are relevant, non-redundant, informative and directly related to the classification task.

B-MOHGSO is used to select the optimal feature subsets. The B-MOHGSO algorithm starts by generating randomly an initial population of n solutions, where each feature subset is encoded by a binary sequence. The clustering step is applied to divide the population into multiple clusters. After that, the objective values are computed for each solution. Then, an $N+1$ external-archives are used to preserve the best feature subsets found so far. Once the fixed number of iterations is defined, the process of updating solutions is started. An efficient archiving strategy based on the crowding distance is adopted to guarantee the exploration and exploitation properties. Each solution is assigned two leaders, chosen from the least populated archive by using the crowding distance, which enables to guide the solutions to a well-distributed Pareto front and update the position of the current population. Then, binarize the updated positions, this process is repeated until the maximum number of iterations is met. In fact, the use of Multi-archives ensures the effective collection of the obtained best feature subsets, and the use of Multi-leaders from the Multi-archives improves the convergence and diversity abilities of the suggested algorithm and enhances the search capability of different kinds of solutions. We propose the following mathematical model for the B-MOHGSO algorithm:

- **Step1:** Initial population is generated, the n gases are distributed uniformly in the entire search space;
- **Step2:** The population is segmented into $(k = 5)$ clusters, with n/N gases randomly allocated to each cluster;
- **Step3:** $N+1$ archives are generated to maintain the elitism concepts during optimization and save the non-dominated feature subsets for the next iteration;

The N local-archives are utilized to keep the non-dominated feature subsets discovered within their respective K clusters, with each local-archive restricted to a maximum size of T_{lmax}. In contrast, the global-archive, which has a maximum size of T_{gmax}, is employed to store the best non-dominated feature subsets (optimal feature subsets) obtained from the entire population.

- **Step4:** Update both Henry's coefficient H_j and solubility $S_{(i.j)}$. For each group, H_j is updated using the formula given in 3.2.3. Then, the solubility $S_{(i.j)}$ of i^{th} gas in the j^{th} cluster is updated as in 3.2.4;
- **Step5:** The position $X_{i,j}$ in iteration $t+1$ is updated as in 3.2.8 :
- **Step6:** Binarize the updated positions using S-shaped functions 4.2.2 and V-shaped functions 4.2.3;
- **Step7:** The updated local-archives and the current global-archive are merged into a single set, from which the global non-dominated feature subsets are then identified, when $|global - archive| > T_{gmax}$, the outsized solutions are deleted from the most crowded segments

Algorithm 6 outlines the pseudocode of the suggested approach B-MOHGSO

Algorithm 6: Pseudo-code of B-MOHGSO algorithm.

load the dataset
Random gases population $X_i(i = 1, 2, ..., n)$
Divide the population into equal ($k = 5$) clusters
Evaluate each individual in X_i (increasing accuracy and reducing dimensionality)
Initialize Local-Archives and the Global-Archive
while $t < Max\_It$ **do**
    Sort the archives in decreasing crowding distance values
    for *each search agent* **do**
        Update Henry's coefficient for each cluster by equation3.2.3
        Update solubility of each gas by equation 3.2.4
        $X_{(j.leader)}$ = Select a local-leader from the local-archive
        $X_{Gleader}$ = Select a global-leader from the global-archive
        Update the position of the current individual by equation 3.2.8
    end
    Binarize the updated positions of X_i using transfer function
    Evaluate each individual in the updated X_i (increasing accuracy and reducing dimensionality)
    Update the local-archives
    Global-archive = $\bigcup_{i=1}^{N+1} External - Archives$
    Update the Global-archive
    t = t+1.
end
Global-archive: the optimal feature subsets.

5.2.4 Complexity of B-MOHGSO algorithm

B-MOHGSO complexity depends on several factors, including the number of features D, number of objectives M, population size n, fitness evaluation, and number of iterations (Max_It). An approximate time complexity of the proposed B-MOHGSO for feature selection is calculated considering these main operations:

- Random initialization has a complexity of $O(D \times n)$, each candidate solution is represented by a binary vector(features selected or not selected) of size D;
- Fitness Evaluation (Wrapper evaluation) has a complexity of $O(M \times n)$;
- Updating archive (update global archive) has a complexity of $O(M \times n \times (n-1))$.

Hence, for a fixed number of iterations, the B-MOHGSO's time complexity would be:

$$O(Max\_It \times (D \times n + M \times n + M \times n \times (n-1))).$$

5.3 Experimental results and discussion

In this section, the impact of the two types of transfer functions is investigated, and the eight binary MOHGSO algorithms are compared with each other to select the best one. The eight binary versions of MOHGSO are implemented in Matlab and compared their performance. The algorithms were evaluated over 10 independent runs per dataset, with a maximum of 100 iterations, a population size of 100 gases, an archive-local size of 50, and an archive-global size of 100, and we reported the statistical results obtained. In this study, the parameter values were used as recommended by the original HGSO algorithm authors. The parameters of our algorithm are listed in Table 5.1.

Table 5.1: Parameters values of B-MOHGSO algorithm.

Algorithm	**Parameters**
B-MOHGSO	Gases Number: $n = 100$ Maximum number of iterations:100 Number the run: = 10 Dimension corresponds to the number of features $\alpha = 1, \beta = 1$ and $K = 1$ Cluster number: $n = 5$ $l_1 = 5E-03$, $l_2 = 1E+02$, and $l_3 = 1E-02$ $M_1 = 0.1$ and $M_2 = 0.2$ $\epsilon = 0.05$ and $F = \pm 1$ T_{lmax}=50: the size of local-archive. T_{gmax}=100: the size of global-archive.

The effectiveness of the eight versions of B-MOHGSO was assessed on twelve datasets from the UCI repository and compared against each other to determine the superior algorithm and validate the performance of the selected best transfer function. Furthermore, the performance of the algorithms is checked on a real-world COVID-19 dataset. Table 5.2 summarizes the characteristics of the datasets used for benchmarking in terms of the number of features, the number of instances ...etc.

The experiment focuses on evaluating the impact of different transfer functions in MOHGSO when used with a k-NN classifier (k = 5). The evaluation metrics are accuracy, the number of selected features, and the computational time, whereas higher accuracy and fewer features with shorter time indicate higher performance.

During the search process, eight binary versions of Multi-Objective Henry Gas Solubility Optimization (B-MOHGSO) simultaneously optimize two objectives: maximizing

the K-NN classifier's accuracy and minimizing the number of selected features. The classifier is evaluated using 5-fold cross-validation on all datasets. The k-fold cross-validation method is utilized to demonstrate the reliability of an algorithm, allowing the exploration of feature subsets that increase the model's accuracy and reduce the total number of features in a reasonable time.

In the context of using the K-fold cross-validation method to assess our proposed algorithm reliability, K-fold cross-validation involves partitioning the data into (K =10) equal-sized folds. K-1 folds are used for training, while the remaining fold serves as the test set once. This process is repeated k times for training and validation, ensuring a more robust evaluation of the algorithm's performance. The performance metrics are averaged across all iterations to assess the overall performance of the K-NN model on the selected datasets.

Table 5.2: Benchmark Datasets used

Dataset	Instances	No. Features	Data area
Breast-cancer	699	9	Biology
Dermatology	366	34	Biology
DNA	318	57	Biology
Exactly	1000	13	Biology
Exactly2	1000	13	Biology
HeartEW	270	13	Biology
Lung-Cancer	32	56	Biology
Lymphography	148	18	Biology
M-of-n	1000	13	Biology
Mushroom	8124	22	Biology
Primary-tumor	339	17	Biology
Soybean-small	47	35	Biology

5.3.1 Results on UCI' datasets

The results of the classification accuracies and the average of selected features obtained by the eight algorithms are outlined in Tables 5.3 and 5.4, respectively, including the minimum, maximum, and mean accuracy expressed as percentages, along with their standard deviations, where the best results are shown in boldface, and the classification accuracies obtained using all features are displayed in parentheses. This study aims to investigate the impact of the eight different transfer functions on the performance of the MOHGSO algorithm for wrapper feature selection using the k-NN classifier.

Table 5.3: The impact of S-shaped and V-shaped transfer functions in MOHGSO over classification accuracy metric.

Algorithm:	B-MOHGSOs1	B-MOHGSOs2	B-MOHGSOs3	B-MOHGSOs4	B-MOHGSOv1	B-MOHGSOv2	B-MOHGSOv3	B-MOHGSOv4
Breast-cancer	(94,57%)							
min(%)	96.86	97.14	96.86	97.43	97.71	97.71	98.00	98.00
max(%)	97.71	97.71	97.71	98.00	97.71	97.71	98.00	98.00
mean(%)	97.12	97.43	97.46	97.55	97.71	97.71	**98.00**	**98.00**
Std.	3.25E-03	2.86E-03	1.85E-03	1.62E-03	5.00E-15	2.23E-15	6.70E-16	1.00E-15
Dermatology	(92,35%)							
min(%)	88.52	83.06	93.44	89.07	99.45	98.36	98.91	98.36
max(%)	98.91	98.91	98.91	98.91	99.45	98.36	99.45	98.91
mean(%)	95.49	91.26	96.75	95.81	**99.45**	98.36	98.91	98.76
Std.	3.14E-02	6.38E-02	2.02E-02	3.35E-02	3.42E-16	3.12E-15	3.78E-15	2.41E-03
DNA	(33,96%)							
min(%)	55.19	95.63	90.16	95.08	98.36	98.91	98.91	98.91
max(%)	98.36	99.45	98.91	98.91	98.91	98.91	98.91	98.91
mean(%)	90.91	98.65	96.75	96.86	98.40	**98.91**	**98.91**	**98.91**
Std.	1.11E-01	1.02E-02	2.80E-02	1.37E-02	1.37E-03	1.11E-15	1.14E-16	3.78E-15
Exactly	(62,00%)							
min(%)	68.80	68.80	72.80	69.20	74.00	78.80	76.20	74.40
max(%)	70.00	74.00	74.80	81.80	81.80	81.80	90.20	81.80
mean(%)	72.81	70.91	73.96	75.80	76.05	79.57	**83.66**	80.34
Std.	3.27E-02	2.03E-02	5.81E-03	5.43E-02	3.21E-02	1.21E-02	6.42E-02	1.90E-02
Exactly2	(72,40%)							
min(%)	76.40	76.40	76.40	76.40	78.20	79.60	78.40	78.20
max(%)	78.20	78.00	78.00	78.40	78.40	79.60	78.40	78.80
mean(%)	76.69	76.70	76.73	76.80	78.20	**79.60**	78.40	78.60
Std.	5.96E-03	5.71E-03	5.53E-03	6.99E-03	3.10E-04	5.60E-16	0.00E+00	2.85E-03
HeartEW	(60,74%)							
min(%)	77.78	83.70	75.56	77.78	85.19	85.93	85.93	85.19
max(%)	85.93	84.44	87.41	84.44	85.93	85.93	85.93	85.19
mean(%)	81.98	84.28	81.38	82.89	85.43	**85.93**	**85.93**	85.19
Std.	2.45E-02	3.12E-03	3.29E-02	1.97E-02	3.51E-03	1.00E-15	3.35E-16	3.65E-16
Lung-Cancer		(75,00 %)						
min(%)	75.00	75.00	75.00	75.00	75.00	75.00	75.00	94.44
max(%)	75.00	75.00	75.00	75.00	75.00	75.00	75.00	94.44
mean(%)	75.00	75.00	75.00	75.00	75.00	75.00	75.00	**94.44**
Std.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.22E-16
Lymphography	(64,86%)							
min(%)	75.68	71.62	37.84	74.32	82.43	85.14	83.78	83.78
max(%)	85.14	83.78	83.78	85.14	83.78	85.14	83.78	83.78
mean(%)	78.74	78.52	71.41	81.43	82.46	**85.14**	83.78	83.78
Std.	2.08E-02	3.15E-02	1.70E-01	3.13E-02	1.91E-03	1.68E-15	0.00E+00	1.17E-16
M-of-n	(66,40%)							
min(%)	77.60	85.00	86.80	85.60	91.60	92.20	91.60	90.00
max(%)	92.40	98.40	89.90	93.00	91.60	98.00	96.40	93.80
mean(%)	89.40	90.01	87.21	90.20	91.60	94.12	**94.13**	92.00
Std.	4.59E-02	5.11E-02	8.41E-03	2.71E-02	5.62E-16	2.74E-02	2.41E-02	1.85E-02
Mushroom	(98,99%)							
min(%)	51.62	98.92	98.84	99.24	82.43	100.00	100.00	100.00
max(%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
mean(%)	95.25	99.88	99.93	99.84	87.80	**100.00**	**100.00**	**100.00**
Std.	1.43E-01	3.08E-03	2.85E-03	2.01E-03	8.06E-02	0.00E+00	0.00E+00	0.00E+00
Primary-tumor	(32,94%)							
min(%)	25.88	27.06	28.82	35.29	42.94	41.76	44.71	45.88
max(%)	46.47	42.94	44.12	43.53	43.53	43.53	44.71	45.88
mean(%)	36.82	39.48	36.59	40.40	43.06	42.00	44.71	**45.88**
Std.	7.84E-02	4.56E-02	5.85E-02	2.22E-02	2.39E-03	5.92E-03	2.24E-16	3.35E-16
Soybean-small		(70,83%)						
min(%)	54.17	75.00	75.00	62.50	100.00	100.00	100.00	100.00
max(%)	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
mean(%)	85.67	86.99	90.49	90.54	**100.00**	**100.00**	**100.00**	**100.00**
Std.	1.36E-01	8.14E-02	1.05E-01	1.04E-01	0.00E+00	0.00E+00	0.00E+00	0.00E+00

Based on these results, it is evident that the B-MOHGSOv (which utilizes V-shaped transfer functions) achieved higher maximum, minimum, and average accuracy classification, and achieved lower standard deviation values compared to the B-MOHGSOs (which utilizes S-shaped transfer functions) over the twelve benchmark datasets.

Also, the B-MOHGSOv selected fewer average features over all benchmark datasets, indicating that these methods chose the minimum feature in most cases. Among the B-

MOHGSOv1 to v4, the results show that the B-MOHGSOv3 outperformed the others by obtaining a higher average accuracy in most datasets and ranked second on three other datasets. B-MOHGSOv4 is best on five datasets.

It is clearly seen the superior performance of B-MOHGSOv3 in most statistical results on seven datasets (Breast-cancer, DNA, Exactly1, HeartEW, M-of-n, Mushroom, and Soybean-small), while B-MOHGSOv4 provides the best performance on sex datasets (Breast-cancer, Lung-Cancer, Mushroom, Primary-tumor, and Soybean-small). Note that, the results for B-MOHGSOv3 and B-MOHGSOv4 are almost similar in most cases.

For S-shaped transfer functions, the B-MOHGSOs4 outperformed B-MOHGSOs1, B-MOHGSOs2, and B-MOHGSOs3 in terms of accuracy in most datasets, while B-MOHGSOs3 minimized features effectively compared to other S-shaped methods in seven datasets. Here, we take into account that the B-MOHGSOs1 has achieved an accuracy lower than that obtained using all features in terms of minimum accuracy in four datasets.

For Mushroom and Soybean-small datasets, the B-MOHGSOv achieved higher accuracy values and reached 100% which impacts the performance of the machine-learning model.

Regarding the stability of the algorithms under study, we can see similar results of the B-MOHGSOv algorithms and there are no significant differences in their performance in most datasets. That suggests the behavior of these algorithms is stable and their performance is robust in different datasets.

Considering the simulation results, we can conclude that the V-shaped transfer functions enabled the MOHGSO algorithm to explore the binary search space more effectively and identify minimal feature subsets that maximize accuracy compared to S-shaped functions.

Table 5.4: The impact of S-shaped and V-shaped transfer functions in MOHGSO over the average of selected features.

Algorithm:	B-MOHGSOs1	B-MOHGSOs2	B-MOHGSOs3	B-MOHGSOs4	B-MOHGSOv1	B-MOHGSOv2	B-MOHGSOv3	B-MOHGSOv4
Breast-cancer								
min(%)	44.44	55.56	44.44	66.67	88.89	88.89	88.89	88.89
max(%)	88.89	88.89	88.89	88.89	88.89	88.89	88.89	88.89
mean(%)	81.00	78.24	71.54	84.02	**88.89**	**88.89**	**88.89**	**88.89**
Std.	1.30E-01	1.40E-01	1.41E-01	6.31E-02	5.34E-15	2.45E-15	2.12E-15	1.45E-15
Dermatology								
min(%)	45.45	45.45	51.52	57.58	96.97	51.52	96.97	51.52
max (%)	81.82	84.85	81.82	84.85	96.97	96.97	96.97	96.97
mean(%)	61.24	74.08	67.79	74.02	**96.97**	94.40	**96.97**	92.27
Std.	1.12E-01	1.20E-01	1.19E-01	8.88E-02	5.70E-16	1.05E-01	7.79E-16	1.39E-01
DNA								
min(%)	45.45	48.48	57.58	57.58	57.58	96.97	96.97	96.97
max(%)	81.82	78.79	81.82	84.85	96.97	96.97	96.97	96.97
mean(%)	64.18	60.94	69.87	72.42	94.34	**96.97**	**96.97**	**96.97**
Std.	1.20E-01	8.04E-02	9.14E-02	9.79E-02	9.85E-02	5.68E-16	7.79E-16	7.80E-16
Exactly								
min(%)	46.15	53.85	61.54	15.38	15.38	15.38	53.85	15.38
max(%)	92.31	92.31	92.31	92.31	92.31	92.31	92.31	92.31
mean (%)	68.04	**81.01**	73.23	65.20	77.01	73.40	70.46	47.79
Std.	2.03E-01	1.33E-01	9.44E-02	3.29E-01	2.88E-01	3.10E-01	1.69E-01	3.23E-01
Exactly2								
min(%)	30.77	7.69	53.85	30.77	30.77	92.31	92.31	15.38
max(%)	92.31	92.31	92.31	92.31	92.31	92.31	92.31	92.31
mean(%)	84.96	84.17	84.88	84.45	90.80	**92.31**	**92.31**	41.03
Std.	1.74E-01	2.10E-01	1.29E-01	1.44E-01	9.54E-02	1.12E-15	1.15E-16	3.65E-01
HeartEW								
min(%)	7.69	76.92	30.77	61.54	84.62	92.31	92.31	92.31
max(%)	92.31	92.31	92.31	92.31	92.31	92.31	92.31	92.31
mean(%)	70.30	80.27	81.73	73.16	89.72	**92.31**	**92.31**	**92.31**
Std.	2.13E-01	6.49E-02	1.18E-01	1.01E-01	3.65E-02	2.01E-15	1.13E-16	2.01E-15
Lung-Cancer								
min(%)	71.43	73.21	82.14	80.36	98.21	98.21	98.21	98.21
max(%)	71.43	73.21	82.14	80.36	98.21	98.21	98.21	98.21
mean(%)	71.43	73.21	82.14	80.36	**98.21**	**98.21**	**98.21**	**98.21**
Std.	1.12E-16	3.40E-16	8.93E-16	1.15E-16	2.12E-14	2.12E-14	2.12E-14	2.12E-14
Lymphography								
min(%)	50.00	27.78	50.00	55.56	55.56	94.44	94.44	94.44
max(%)	83.33	88.89	94.44	94.44	94.44	94.44	94.44	94.44
mean(%)	71.13	73.17	76.18	74.11	93.66	**94.44**	**94.44**	**94.44**
Std.	7.40E-02	1.40E-01	1.65E-01	9.52E-02	5.49E-02	1.23E-15	1.22E-16	0.00E+00
M-of-n								
min(%)	53.85	53.85	53.85	46.15	92.31	53.85	46.15	46.15
max(%)	92.31	92.31	92.31	92.31	92.31	92.31	92.31	92.31
mean(%)	68.11	70.74	90.33	64.90	**92.31**	79.58	67.99	69.48
Std.	1.26E-01	1.26E-01	5.45E-02	1.76E-01	5.62E-16	1.82E-01	2.32E-01	2.13E-01
Mushroom								
min(%)	63.64	59.09	86.36	54.55	50.00	95.45	95.45	95.45
max(%)	86.36	86.36	90.91	90.91	94.44	95.45	95.45	95.45
mean(%)	73.30	67.98	86.65	76.53	87.92	**95.45**	**95.45**	**95.45**
Std.	7.64E-02	9.18E-02	1.12E-02	9.17E-02	1.16E-01	1.22E-15	5.57E-16	1.00E-15
Primary-tumor								
min(%)	29.41	35.29	35.29	41.18	41.18	41.18	94.12	94.12
max(%)	88.24	94.12	94.12	94.12	94.12	94.12	94.12	94.12
mean(%)	69.69	61.46	71.81	76.29	83.24	83.24	**94.12**	**94.12**
Std.	2.06E-01	2.30E-01	2.37E-01	1.58E-01	2.15E-01	2.15E-01	1.46E-15	2.12E-15
Soybean-small								
min(%)	60.00	65.71	65.71	65.71	97.14	97.14	97.14	97.14
max(%)	77.14	77.14	85.71	85.71	97.14	97.14	97.14	97.14
mean(%)	69.30	74.77	74.98	72.86	**97.14**	**97.14**	**97.14**	**97.14**
Std.	4.90E-02	2.37E-02	8.45E-02	6.73E-02	6.89E-15	4.12E-15	2.45E-15	6.56E-15

The obtained Pareto front of the eight algorithms for each dataset are presented in Figure 5.1, to show how well each algorithm balances the trade-off solutions (higher accuracy with fewer features).

B-MOHGSOv3 has the best coverage of the search space on most datasets, as well as producing non-dominated solutions distributed along the entire Pareto front. It obtains solutions with high accuracy using approximately 50 to 60% of the features in most cases.

The Pareto fronts obtained from the B-MOHGSOv3 and B-MOHGSOv4 dominate those obtained by other algorithms, especially for Breast-cancer, Dermatology, DNA, Exactly, HeartEW, Lymphography, M-of-n, and Mushroom datasets. Specifically, B-MOHGSOv3 achieves 98% accuracy using only 20% of the features in the Breast-cancer

dataset.

B-MOHGSOv2 achieves good diversity with high accuracy, while B-MOHGSOv1 achieves solutions with higher accuracy with higher percentages of features.

The Pareto fronts of B-MOHGSOs1 and B-MOHGSOs2 are clustered in the left corner, indicating low accuracy solutions using higher features. B-MOHGSOs3 and B-MOHGSOs4 exhibit better diversity than B-MOHGSOs1 and B-MOHGSOs2 but are still dominated by the B-MOHGSO with V-shaped transfer functions.

In summary, the Pareto fronts obtained indicate that the algorithms with V-shaped transfer functions can find solutions distributed along the top right corner, achieving solutions with higher accuracy using fewer features. B-MOHGSOv3 exhibits better exploration of the search space, resulting in diverse non-dominated feature subset balancing accuracy and features.

The graph illustrates a representation of how well each optimizer balances the trade-off solutions between higher accuracy and fewer features. It confirms that algorithms with V-shaped transfer functions outperform exploring the binary feature space compared to algorithms with S-shaped functions, which often exhibit limitations in terms of diversity.

Table 5.5: Average computational time (in minutes)

Algorithm:	B-MOHGSOs1	B-MOHGSOs2	B-MOHGSOs3	B-MOHGSOs4	B-MOHGSOv1	B-MOHGSOv2	B-MOHGSOv3	B-MOHGSOv4
Breast-cancer	5.80	5.86	5.82	6.96	5.25	5.23	5.21	**5.15**
Dermatology	4.92	5.18	4.90	4.90	**4.87**	4.89	4.88	5.39
DNA	5.38	5.36	5.37	5.36	5.34	**5.28**	5.30	5.29
Exactly	5.81	5.79	5.77	5.84	5.68	5.40	**5.38**	5.40
Exactly2	5.41	**5.38**	5.39	5.40	**5.38**	**5.38**	5.42	5.41
HeartEW	5.02	5.04	5.02	5.02	5.52	5.03	5.10	**5.01**
Lung-Cancer	4.79	**4.76**	4.78	5.18	6.15	6.12	6.11	6.24
Lymphography	10.02	10.26	8.63	5.35	5.38	**5.34**	7.25	7.08
M-of-n	7.70	6.18	5.71	5.71	5.82	5.73	5.71	**5.70**
Mushroom	13.78	14.65	16.36	16.17	**13.43**	14.29	14.87	13.69
Primary-tumor	6.98	8.15	6.76	6.79	**6.73**	6.79	6.77	6.85
Soybean-small	5.08	4.83	4.82	4.84	**4.78**	4.88	4.79	4.80

Table 5.5 outlines the average computational time (in minutes) for the eight versions of the B-MOHGSO algorithm on twelve benchmark datasets. The findings show that the computational time changes depending on the algorithm and the category of the data used. For example, for the small datasets(0 – 15 features), the lowest average running time ranges from 5,01 minutes in the HeartEW dataset and 5,70 minutes in the M-of-n dataset. Similarly, for the medium datasets(16 - 25 features), the lowest average running time ranges from 5,34 minutes in the Lymphography dataset and 13,43 minutes in the Mushroom dataset. For the large datasets (>30 features), the lowest average running time ranges from 4,76 minutes in the Lung-Cancer dataset and 13,43 minutes in the DNA dataset.

On the other hand, the computational time for the B-MOHGSO algorithms with V-shaped transfer functions is generally almost similar to that for algorithms with S-shaped transfer functions on the chosen dataset. In conclusion, the time varies depending on the differences between the B-MOHGSO versions over all datasets; however, there was no clear winner among the eight algorithms.

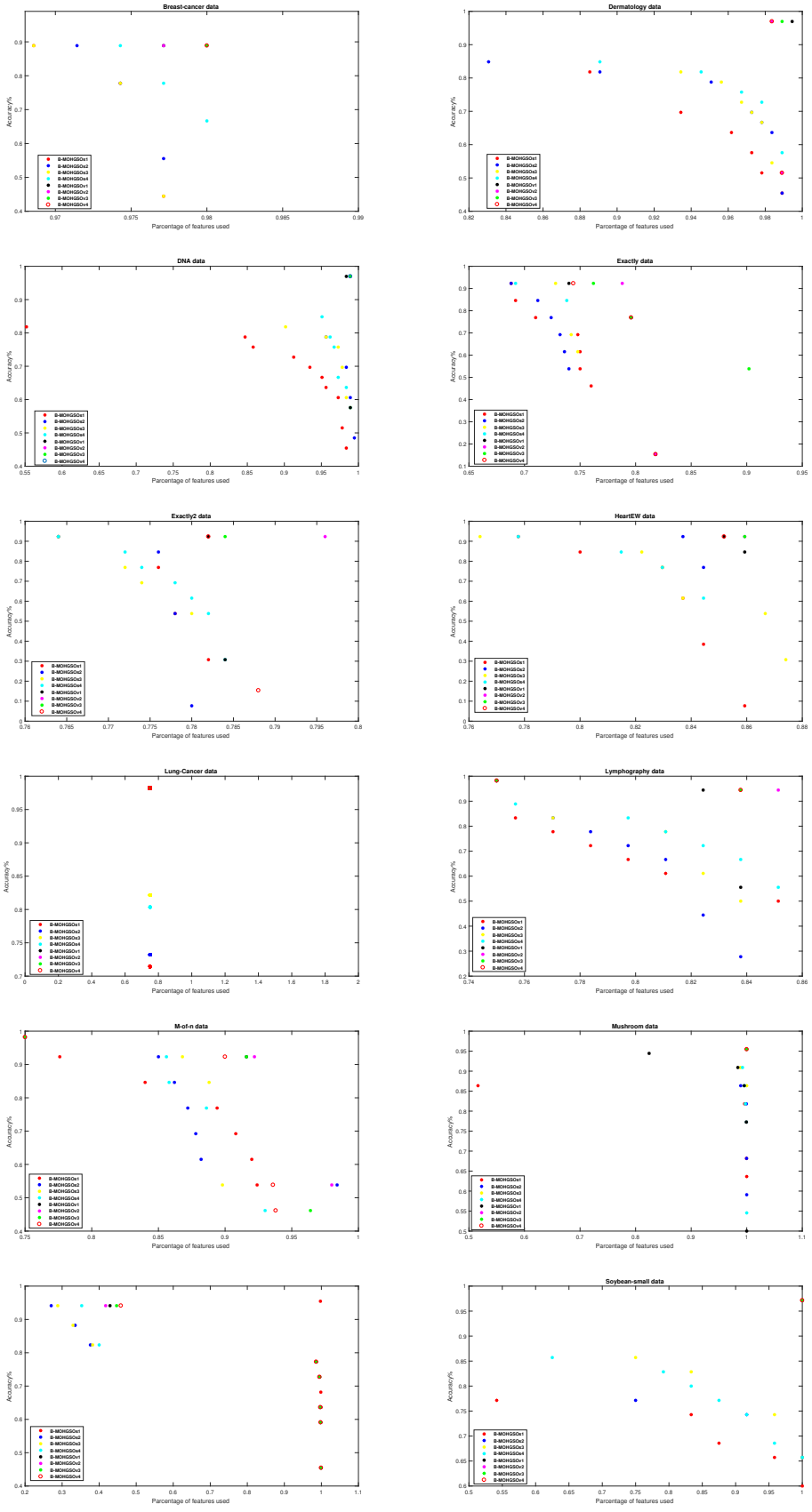


Figure 5.1: Comparison between the optimizers for twelve UCI' datasets.

Overall, the typical runtime for different versions of the B-MOHGSO algorithm depends on the specific dataset (varying sizes and characteristics) and the sensitivity of the algorithm's parameters. Although the differences between the versions of the B-MOHGSO algorithm, the total runtime of the proposed algorithms for 100 iterations with a population size of 100 and the same parameters that are used to ensure fairness in comparison indicates the proposed MOHGSO algorithm's efficiency.

Our analysis of B-MOHGSO algorithms using V-shaped transfer functions reveals several significant observations into their multi-objective feature selection performance:

Enhanced Classification Accuracy: These algorithms achieved higher minimum, maximum, and mean accuracies across the datasets compared to those using S-shaped transfer functions.
Effective Dimensionality Reduction: The V-shaped transfer functions enabled the MOHGSO algorithm to identify minimal feature subsets while maintaining high accuracy.
Stability: The performance of the B-MOHGSO algorithms using V-shaped transfer functions demonstrated stability across different datasets, suggesting their ability to various feature selection challenges.
Improved Search Space Exploration: The V-shaped transfer functions allowed the B-MOHGSO algorithms to explore the binary search space more effectively, leading to a better balance between exploration and exploitation during the optimization process. This capability is essential for avoiding local optima and discovering diverse non-dominated feature subsets.

The effectiveness of the proposed algorithms was assessed on twelve benchmark datasets from the UCI Machine Learning Repository. The results indicate that the B-MOHGSO algorithms, particularly those utilizing V-shaped transfer functions outperform those using S-shaped transfer functions in terms of classification accuracy and feature selection efficiency. In summary, the V-shaped transfer functions significantly enhanced the B-MOHGSO algorithm's ability to perform multi-objective feature selection, making them a preferred choice for binary optimization problems in various applications.

5.4 Conclusion

In this chapter, we extend the B-HGSO approach to a multi-objective framework, this work focuses on the development and evaluation of the Binary Multi-objective Henry Gas Solubility Optimization (B-MOHGSO) algorithm for feature selection tasks. The B-MOHGSO algorithm is designed to optimize two objectives: maximizing classification accuracy while minimizing the number of selected features. By exploring the trade-offs between performance and efficiency of K-NN machine learning model. The B-MOHGSO algorithm incorporates a multi-archive strategy to maintain a diverse set of non-dominated

feature subsets, enabling effective exploration of the solution space. The use of clustering and crowding distance enhances the algorithm's ability to balance between exploration and exploitation, which is necessary for avoiding local optima.

The findings confirm the importance of transfer functions in the optimization process and suggest that both S-shaped and V-shaped functions have their roles in improving classification accuracy and reducing dimensionality. The proposed algorithm allows for an exploration of the trade-off between model simplicity (fewer features) and performance (higher accuracy), resulting in a Pareto front of solutions (feature subsets), each representing a different balance between these two objectives.

Overall, this work contributes to advancing multi-objective optimization techniques in the field of machine learning and data mining. Our proposed algorithm is particularly relevant in the context of machine learning, where achieving high accuracy with a simpler model (fewer features) is often desired. The demonstrated effectiveness of the B-MOHGSO algorithm in feature selection motivated us to apply it to high-dimensional dataset.

CHAPTER 6

Deep Learning and Feature Selection for COVID-19 Classification

6.1 Introduction

The rapid spread of COVID-19 has necessitated the development of reliable and efficient diagnostic tools to assist in managing the pandemic. Machine learning has emerged as a crucial technology in this regard, offering potential solutions for early detection and classification of COVID-19 cases. However, the high dimensionality of medical datasets often hampers the performance of these models. Feature selection, which involves identifying the most relevant features from a dataset, plays a vital role in enhancing the accuracy of machine learning classifiers. This chapter outlines the application of the best-performing B-HGSOs1 algorithm on a real Covid-19 chest X-ray dataset as an effective tool for feature selection, leveraging the power of GoogleNet and ResNet deep learning models for feature extraction. The proposed method was compared to B-PSO and B-GWO. Then, eight versions of Multi-Objective HGSO algorithm were combined to classify this dataset, which consists of 3 classes of Covid-19, pneumonia, and normal lung X-ray images. To validate the performance of the proposed algorithms and determine the superior algorithm, the metrics used were accuracy, computational runtime, and average selected features over deep learning models with eight transfer functions using the same k-NN classifier. Additionally, the k-fold cross-validation method was employed to ensure the reliability of the results.

6.1.1 Deep learning models

Deep learning algorithms play a crucial role in improving diagnostic tools for COVID-19, by providing models to extract valuable insights from complex medical data. However, their effectiveness and performance depend on the quality and relevance of the features

used for training. The application of feature selection optimization allows us to balance competing objectives, such as maximizing classification accuracy while minimizing the number of features used. This dual is particularly important in the context of medical diagnostics, as a simpler model with fewer features can result in faster processing times and easier interpretation, both of which are essential in clinical settings.

In this work, four deep-learning models were employed for extracting features from enhanced medical images consisting of three classes: Covid-19, pneumonia, and normal lung X-ray images. Each model provided 1000 features, which were then selected using the feature selection of the proposed algorithm and classified using K-Nearest Neighbor (K-NN).

AlexNet is a deep CNN architecture that consists of eight layers: five convolutional layers followed by three fully connected layers. The architecture employs techniques such as ReLU (Rectified Linear Unit) activations, dropout for regularization, and data augmentation to improve performance. The work in [73] represents the evolution of deep learning for image classification, by demonstrating the effectiveness of deep convolutional networks on a large-scale dataset, the success of AlexNet has led to an increased interest in deep learning and CNNs, Which lead to many developments in this field. It set the way for subsequent structures and techniques in computer vision.

GoogleNet, introduced in 2015, focused on the importance of going deeper within neural networks and treated the challenge of increasing neural network depth by employing inception modules. These modules combine diverse convolution and pooling layers, prompting feature extraction without excessive parameter growth, leading to improved performance in visual recognition tasks. The Inception architecture has since influenced many subsequent developments in deep learning and computer vision [127].

ResNet, introduced in 2015, revolutionized deep learning with its residual blocks, treating vanishing gradients and allowing the construction of deeper networks. This makes it easier to reduce the number of parameters calculated compared to traditional deep-learning models. It has become a widely adopted architecture, as a foundational for many subsequent models and applications. ResNet models represent significant advancement in deep learning architectures for image recognition [63].

VGGNet is a series of convolutional neural network model with varying depths (up to 19 layers) presented in [120], in order to explore the impact of increasing the depth of convolutional networks on image classification tasks. VGGNet has become a popular deep learning architecture for image recognition, its design has influenced many later models and serves as a benchmark for evaluating new techniques for various tasks, making it versatile for different applications in computer vision.

6.2 B-HGSO for COVID-19 X-ray Classification: A Feature Selection Approach

This section outlines the application of the B-HGSOs1 algorithm to a real COVID-19 dataset in which the K-Nearest Neighbor (k-NN) is employed as a learning classifier to maximize classification accuracy. In [17], a novel approach termed MH-COVIDNet was introduced by combining deep learning and metaheuristic algorithms for the early diagnosis of COVID-19 disease using chest X-ray images, two metaheuristic approaches for feature selection. Firstly, the author created a dataset containing three classes of X-ray images (COVID-19, normal, and pneumonia lung X-ray images); each class contains 364 images. Image contrast enhancement was applied as a preprocessing step to obtain a new dataset. Then, this dataset is used to train four deep learning models, including AlexNet, VGG19, GoogleNet, and ResNet, for feature extraction. Next, two metaheuristic algorithms: binary particle swarm optimization (B-PSO) and binary gray wolf optimization (B-GWO), are employed to select the most relevant features. These selected features are then classified using a Support Vector Machine (SVM).

In this work, we use the enhancement dataset described in the mentioned paper, which refers to a preprocessed version of the original dataset in order to improve the quality of the images by the application of an Image Contrast Enhancement Algorithm (ICEA). Two deep learning models including GoogleNet and ResNet extract features from the preprocessed images. These features are then input into the B-HGSOs1 algorithm to select the most relevant ones, which helps reduce dimensionality while retaining the most informative features. Finally, the selected features are used to train a K-NN classifier to differentiate between COVID-19, pneumonia, and normal cases.

Note that, we use several performance metrics that are very similar to those used in the previously mentioned paper in order to evaluate the classification models. We used k-fold cross-validation (k=5) to ensure the robustness and reliability of the obtained results. We also compared the performance of the proposed B-HGSOs1 algorithm with Binary Particle Swarm Optimization (BPSO) and Binary Gray Wolf Optimization (BGWO) mentioned in the paper.

For a fair comparison, experimental studies assigned 30% of the enhancement data for testing and 70% for training at each stage. Performance evaluation utilized metrics calculated from the confusion matrix (see Figure 6.1) are given as follows:

1. **Accuracy (Acc)**: The ratio of correctly predicted instances (true positives and true negatives) to the total number of cases examined.

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

2. **Sensitivity (Se)**: Also known as recall, it is the ratio of true positive predictions to the total number of actual positive cases.

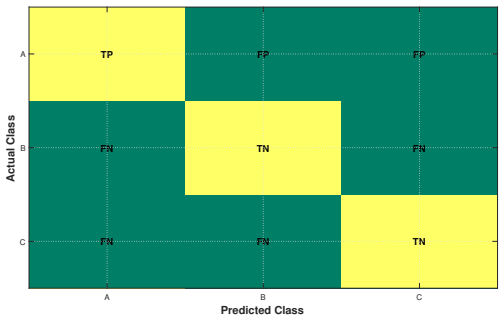


Figure 6.1: Multiclass classification confusion matrix.

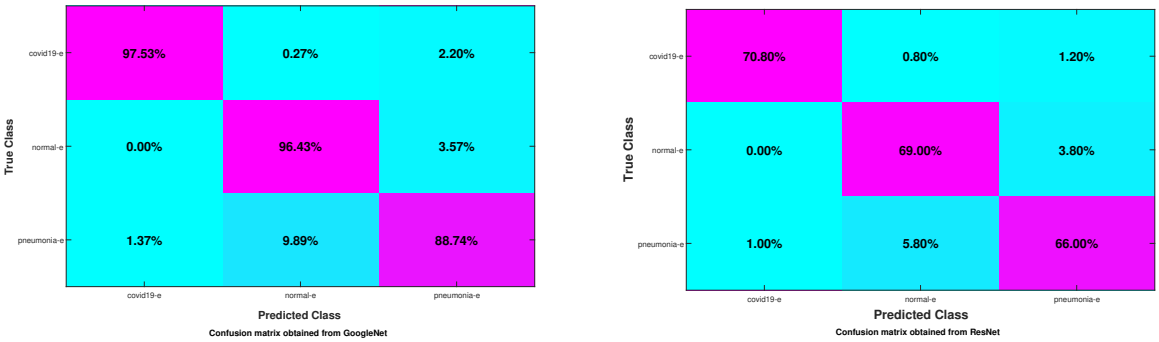


Figure 6.2: Confusion matrices obtained using the B-HGSOs1 method for GoogleNet and ResNet models.

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

3. **Precision (Pre)**: The ratio of true positive predictive values to the total number of positive predictions for each class.

$$\text{Precision} = \frac{TP}{TP+FP}$$

4. **F-score (F-Scr)**: It calculates the harmonic mean of precision and sensitivity (recall) metrics, providing a balanced measure of a model's accuracy by combining precision and recall into one metric.

$$F\text{-score} = 2 \times \frac{\text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

here, TP represents the count of true positives, FN indicates the number of false negatives, TN refers to the count of true negatives, and FP denotes the number of false positives.

Figure 6.2 presents two confusion matrices comparing the performance of the proposed B-HGSOs1 that is applied to GoogleNet and ResNet models to classify the data into three

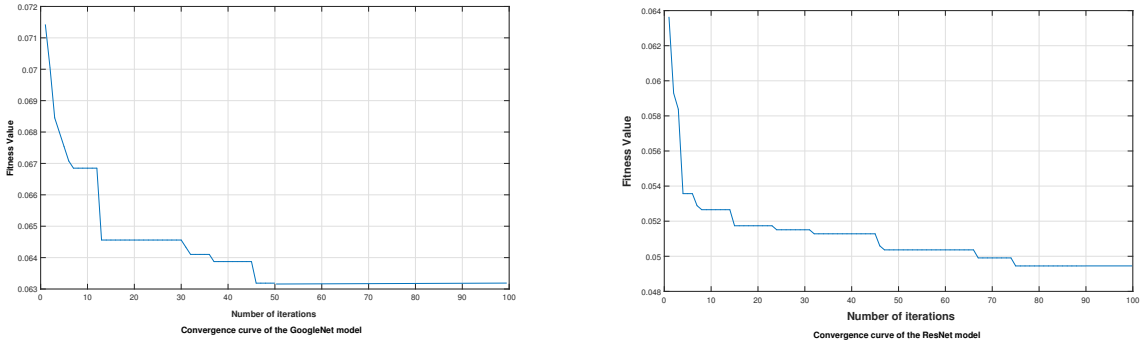


Figure 6.3: Convergence curves obtained using the B-HGSOs1 method for GoogleNet and ResNet models.

classes (covid-19, normal, and pneumonia), in which the higher values on the diagonal indicate a higher number of correct classifications.

GoogleNet model demonstrated superior performance in classifying COVID-19 cases and achieved an accuracy of 97.53%, with a relatively low false negative rate. It correctly classified 96.43% of normal cases and 88.74% of pneumonia cases, it exhibited a false negative rate of 9.89% in classifying pneumonia cases as normal.

70, 80% of COVID-19 cases were correctly classified, according to the ResNet model's confusion matrix. 29, 20% of the cases were false negatives, with 1.00% being diagnosed as pneumonia and 0, 80% as normal. Furthermore, 69, 00% of the normal cases were accurately classified. False negative and false positive rates for normal cases were 5, 80% and 1, 20%, respectively, whereas the true positive rate for pneumonia cases was 66, 00%. The proposed B-HGSOs1 is applied to GoogleNet and ResNet models to classify the data into three classes (covid-19, normal, and pneumonia). Figure 6.2 shows two confusion matrices comparing the performance of these models. The higher values on the diagonal indicate a higher number of correct classifications.

The confusion matrix is used in order to evaluate the classification model's performance by comparing predicted classes against true classes. GoogleNet model performs well with high rates for the correct classifications. However, there is a need for improvement, especially for Class 3 (pneumonia), which has the lowest correct classification rate. While, the ResNet model showed a moderate performance in all three categories (COVID-19, normal, and pneumonia).

Figure 6.3 reveals the fitness values obtained using the proposed B-HGSOs1 algorithm for GoogleNet and ResNet models. Lower fitness values indicate better performance, which represents a decrease in error. The fitness value starts high and decreases rapidly in the first 10 iterations, indicating a fast improvement in the performance of the GoogleNet model. After about 50 iterations, the final fitness value is very low and stable, indicating good overall performance. ResNet model demonstrates a rapid decrease in fitness value in the first few iterations, indicating its rapid improvement. In addition, the lower final fitness value indicates better performance, meaning the ResNet model with a stable value

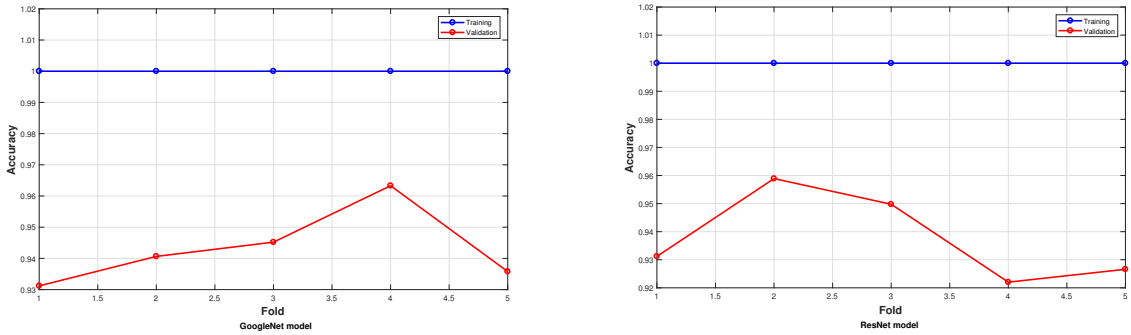


Figure 6.4: Training and Validation Accuracy per Fold of the models.

around $0,049$ is considered better at minimizing the optimization objective compared to the GoogleNet model.

In fact, both models show improvement over iterations, indicating effective training. The stability in the fitness values of the GoogleNet model suggests it has more stable learning. On the other hand, the ResNet achieves a lower final error, indicating a better fit according to the fitness metric used.

Training and Validation accuracy of GoogleNet and ResNet models across five folds of Covid-19 data are presented in Figure6.4. The training accuracy for both models remains constant at 100% for all folds, indicating that the model is achieving perfect accuracy on the training data.

The validation accuracy of the GoogleNet model varies across folds, and ranges from 93% to 97% approximately, indicating that the model's performance on unseen data is a little inconsistent across different subsets of the data. ResNet provides more fluctuations in validation accuracy that range from 92% to 96% across folds and exhibits its maximum validation performance earlier (k= 2) than GoogleNet (k= 4), with a significant drop after the third fold (k= 3).

In summary, the obtained graphs illustrate the models' performance across different cross-validation folds, highlighting the consistent perfect training accuracy and the varying validation accuracy. This variation in validation accuracy suggested the need to evaluate model stability and performance on the unseen data.

Table6.1 presents a comparison of GoogleNet and ResNet models that are applied to classify enhanced medical image data. The models were evaluated using the selected B-HGSOs1 algorithm for feature selection using diverse metrics like accuracy, sensitivity, and precision.

The application of the B-HGSOs1 algorithm is effective for feature selection, as it allows both GoogleNet and ResNet models to achieve acceptable performance with a reduced feature set. GoogleNet uses 270 features from enhancement data and exceeds at identifying COVID-19 cases, with very high F-scores and precision. Results indicate that the ResNet model has a more balanced performance across the three classes, using the

selected 511 features from the same enhancement data.

The similar overall accuracy suggests that both GoogleNet and ResNet models are effective in classifying Covid-19, pneumonia, and normal cases. Despite the significant difference in the number of features used, the results demonstrated the strong performance and suitability of the selected models in the classification task.

Table 6.1: Metric values obtained using the B-HGSOs1 algorithm.

Model	Data Type	Classes	Number of Features	F-Scr (%)	Se. (%)	Pre. (%)	Acc. (%)	Overall Acc. (%)
GoogleNet	Enhancement Data	COVID-19		98.07	97.53	98.62	93.58	
		Pneumonia	270	91.22	88.72	93.92	95.43	94.23
		Normal		93.36	96.43	90.52	95.43	
ResNet	Enhancement Data	COVID-19		97.92	97.25	98.62	94.50	
		Pneumonia	511	91.81	90.66	93.05	94.52	94.22
		Normal		92.98	94.76	91.34	95.89	

The most important part of the feature selection procedure in this work is played by the suggested B-HGSOs1 method, which chooses the most relevant features from the enhanced data following feature extraction using GoogleNet and ResNet models. As a result, the model performs better and the complexity of the data is reduced. Higher accuracy and efficiency can be attained by the models by concentrating on the most informative features.

Table 6.2: Comparison of the proposed B-HGSOs1 with the existing MH-CovidNet method

Algorithms	Model	Feature Size	Accuracy. (%)
B-PSO	GoogleNet	488	95.71
	ResNet	477	96.94
B-GWO	GoogleNet	662	96.33
	ResNet	572	96.94
B-HGSOs1	GoogleNet	511	94.23
	ResNet	270	94.22

Table 6.2 shows a comparison between the performance of the proposed algorithm and the results reported in the cited paper, which were obtained by using both B-GWO and B-PSO algorithms for feature selection in classifying COVID-19 cases, employing both GoogleNet and ResNet models; in the experiments, a K-NN classifier was used for B-HGSOs1, while an SVM classifier was used for B-PSO and B-GWO.

The proposed B-HGSOs1 approach achieves significant feature reduction, especially for the ResNet model with 270 features compared to 477 for B-PSO and 572 for the B-GWO algorithm, even though all algorithms achieve comparable accuracy. While BGWO and BPSO algorithms employ less features than B-HGSOs1, they attain the maximum accuracy ($96,94\%$) with ResNet, indicating that our suggested method strikes a reasonable balance between feature reduction and performance.

In comparison to the B-HGSO algorithm, which uses the K-NN classifier and maintains competitive accuracy ($94,23\%$) using significantly fewer features, the B-GWO and B-PSO

algorithms use the ResNet model to achieve the highest accuracy (up to **96, 94%**). Both algorithms use the Support Vector Machine (SVM) classifier, which is more suited to high-dimensional data. This shows how important it is to balance accuracy and feature efficiency when choosing the model.

Machine learning algorithms play a crucial role in improving diagnostic tools for COVID-19, by providing models to extract valuable insights from complex medical data. However, their effectiveness and performance depend on the quality and relevance of the features used for training. Here, the B-HGSO feature selection process becomes critical, as it aims to identify the most informative features from the enhancement dataset, enhancing model performance. In this approach, the fitness function is designed to optimize the selection process within the specific framework, balancing the trade-offs between reduced dimensionality and improved accuracy.

This study highlights the trade-offs between feature size and classification accuracy across different feature selection algorithms and classifiers. The choice of classifier can significantly impact the model's performance and feature selection method. While the proposed algorithm can effectively select features, the high dimensionality of the COVID-19 dataset favors SVM for better classification performance. This is due to SVM's ability to handle high dimensionality efficiently and avoid overfitting.

This study presents a novel approach to feature selection using the binary Henry Gas Solubility Optimization (b-HGSO) algorithm, specifically applied to COVID-19 classification tasks. Our results indicate that b-HGSO outperforms existing algorithms in terms of accuracy and efficiency, selecting fewer features while maintaining high classification accuracy. These findings suggest that b-HGSO can be a valuable addition to the set of tools of machine learning methods used in medical diagnostics, particularly for COVID-19.

6.3 Binary Multi-Objective HGSO for Accurate COVID-19 Diagnosis

This section outlines the application of the eight binary MOHGSO algorithms on a real Covid-19 dataset, four deep learning models and eight versions of multi-objective metaheuristic algorithms were combined to classify this dataset that consists of 3 classes of COVID-19, pneumonia, and normal lung X-ray images. To determine which algorithm performs best for the selected dataset under study, the metrics are accuracy, computational runtime, and selected features over deep learning models with eight transfer functions using the same k-NN classifier. Also, the k-fold cross-validation method is used to prove the reliability of the proposed algorithms.

Table 6.3 outlines the obtained minimum, maximum, mean, and standard deviation of the classification accuracies. As mentioned above, the classification accuracies obtained using all features are displayed in parentheses, while the best "mean" accuracy achieved by each algorithm is marked in boldface. Knowing that higher accuracy values with

Table 6.3: The impact of S-shaped and V-shaped transfer functions in MOHGSO over classification accuracy metric.

Algorithm:	B-MOHGSOs1	B-MOHGSOs2	B-MOHGSOs3	B-MOHGSOs4	B-MOHGSOv1	B-MOHGSOv2	B-MOHGSOv3	B-MOHGSOv4
AlexNet	(93,68%)							
min(%)	93.68	93.77	93.86	94.05	94.05	94.05	94.32	94.51
max(%)	96.25	96.25	96.15	96.34	96.25	96.25	96.15	96.25
mean(%)	95.43	95.07	95.21	95.37	**95.49**	95.28	95.35	95.40
Std.	6.57E-03	6.60E-03	6.45E-03	5.92E-03	5.75E-03	4.80E-03	4.46E-03	4.21E-03
GoogleNet	(93,22%)							
min(%)	92.67	92.49	92.22	92.49	92.67	92.86	92.31	92.31
max(%)	94.69	94.51	94.60	94.41	94.69	94.60	94.69	94.60
mean(%)	**93.81**	93.54	93.63	93.55	93.60	93.55	93.53	93.57
Std.	5.33E-03	5.54E-03	5.84E-03	6.40E-03	5.65E-03	5.74E-03	5.62E-03	6.44E-03
ResNet	(93,41%)							
min(%)	93.68	93.68	93.86	93.13	93.59	98.08	98.26	93.22
max(%)	95.79	95.70	95.70	95.88	95.70	98.90	98.90	95.79
mean(%)	94.88	94.57	94.86	94.51	94.66	98.52	**98.54**	94.62
Std.	5.07E-03	6.57E-03	4.46E-03	7.34E-03	5.68E-03	2.01E-03	1.65E-03	5.78E-03
VGG19	(98,44%)							
min(%)	98.26	97.99	97.99	98.17	98.17	98.08	98.26	98.08
max(%)	98.90	98.99	98.99	98.90	98.99	98.90	98.90	98.90
mean(%)	98.56	98.50	98.57	98.54	**98.60**	98.54	98.56	98.53
Std.	1.85E-03	2.67E-03	2.58E-03	1.82E-03	1.71E-03	1.92E-03	1.85E-03	2.34E-03

lower average numbers of selected features and shorter average computation times indicate better performance.

Generally, the classification accuracies obtained using the B-MOHGSO algorithm with feature selection were higher than the classification accuracy obtained using all features in all versions and deep learning models. For the AlexNet model, the mean classification accuracy extended from 95.07% as a minimum mean to 95.49% as a maximum mean, while the mean classification accuracy for the GoogleNet model extended from 93.53% to 93.81%.

Also, the results show that the B-MOHGSOv which utilizes V-shaped transfer functions outperformed the B-MOHGSOs with S-shaped transfer functions in terms of classification accuracy, with higher minimum, maximum, and mean accuracies and lower standard deviations.

The experimental results indicate that B-MOHGSO achieves interesting results on a high-dimensional COVID-19 dataset.

According to Table 6.4, most algorithms over the four deep learning models show a similar performance. B-MOHGSO algorithms with V-shaped transfer functions achieved fewer average features in most models, indicating they were effective in minimizing features. However, the eight B-MOHGSO algorithms demonstrated the best performance in exploring the binary search space and identifying the minimal feature subsets, this indicates they are promising algorithms for multi-objective feature selection.

Considering the findings, we can say that the B-MOHGSO can explore different regions of the solution space and impact the diversity and quality of the selected feature subsets. B-MOHGSO's capability (successful) impacts finding feature subsets that contribute positively to classification accuracy. Also, the transfer functions contribute to the efficiency and effectiveness of the binary multi-objective HGSO algorithm in feature

Table 6.4: The impact of S-shaped and V-shaped transfer functions in MOHGSO based over the average of selected features.

Algorithm:	B-MOHGSOs1	B-MOHGSOs2	B-MOHGSOs3	B-MOHGSOs4	B-MOHGSOv1	B-MOHGSOv2	B-MOHGSOv3	B-MOHGSOv4
AlexNet								
min(%)	46.30	44.40	45.40	45.30	45.00	44.90	44.90	44.90
max(%)	51.10	50.90	49.90	50.60	50.90	51.70	51.00	51.70
mean(%)	47.77	46.44	46.86	47.56	47.34	47.42	47.26	**47.95**
Std.	1.47E-02	1.47E-02	1.16E-02	1.27E-02	1.88E-02	1.55E-02	1.47E-02	1.64E-02
GoogleNet								
min(%)	45.70	45.20	44.10	45.00	44.60	46.00	43.90	46.00
max(%)	50.00	51.60	50.50	49.90	51.60	51.10	52.00	50.80
mean(%)	47.07	**47.75**	47.47	47.21	47.14	47.30	47.39	47.11
Std.	8.96E-03	1.59E-02	1.55E-02	1.03E-02	1.47E-02	1.16E-02	1.68E-02	1.35E-02
ResNet								
min(%)	44.90	44.80	44.90	45.40	44.70	44.50	45.70	44.60
max(%)	51.80	51.10	49.90	51.50	50.50	52.10	51.30	50.40
mean(%)	47.71	46.82	47.02	47.07	47.07	**47.83**	47.45	47.16
Std.	1.98E-02	1.28E-02	1.41E-02	1.11E-02	1.37E-02	1.70E-02	1.43E-02	1.10E-02
VGG19								
min(%)	44.90	45.40	45.00	45.70	44.90	45.30	44.90	45.40
max(%)	51.50	50.40	50.60	50.50	49.80	51.50	51.50	49.80
mean(%)	47.21	46.97	46.28	47.27	47.04	47.41	47.21	**47.45**
Std.	1.41E-02	9.81E-03	1.21E-02	1.44E-02	1.26E-02	1.31E-02	1.41E-02	1.00E-02

selection tasks.

The obtained Pareto fronts of the eight algorithms for each deep learning model are presented in Figure 6.5. B-MOHGSO was able to generate a diverse set of non-dominated feature subsets for each model, with each point representing a non-dominated feature subset.

For AlexNet, B-MOHGSOv1 achieves higher accuracy (approximately $95,5\%$) overall with less than 54% of features, it exhibits the most diverse set of non-dominated feature subsets, while the Pareto fronts for algorithms with S-shaped are clustered in the left corner, indicating less variety of solutions. Also, B-MOHGSOv2 has a good diversity with lower percentages of features.

For GoogleNet, the graph shows that B-MOHGSOv3 dominates other algorithms by generating a wide range of non-dominated solutions. Also, B-MOHGSOv1 shows good diversity, while B-MOHGSOv4 performs poorly. B-MOHGSO with S-shaped transfer functions has a slightly better front with approximately 60% of the features.

For ResNet, the fronts generated by both B-MOHGSOv2 and B-MOHGSOv3 completely dominate the other fronts. They achieve over $98,5\%$ accuracy using only 50% of features. The rest of the algorithms have decent diversity, including those that use the S-shaped transfer function with lower accuracy.

For VGG19, B-MOHGSOv1 achieves higher accuracy ($98,6\%$) with fewer features selected, while B-MOHGSOv2 and B-MOHGSOv3 achieve the most diverse solutions showing better distribution and accuracy. The fronts for the algorithms with V-shaped transfer functions are slightly better than those with S-shaped.

Considering the analysis of the obtained Pareto front, the diversity of feature subsets was influenced by two components: i) multi-archives contribute to the effective collection of the best non-dominated feature subsets, ii) multi-leaders ensure diversity among these

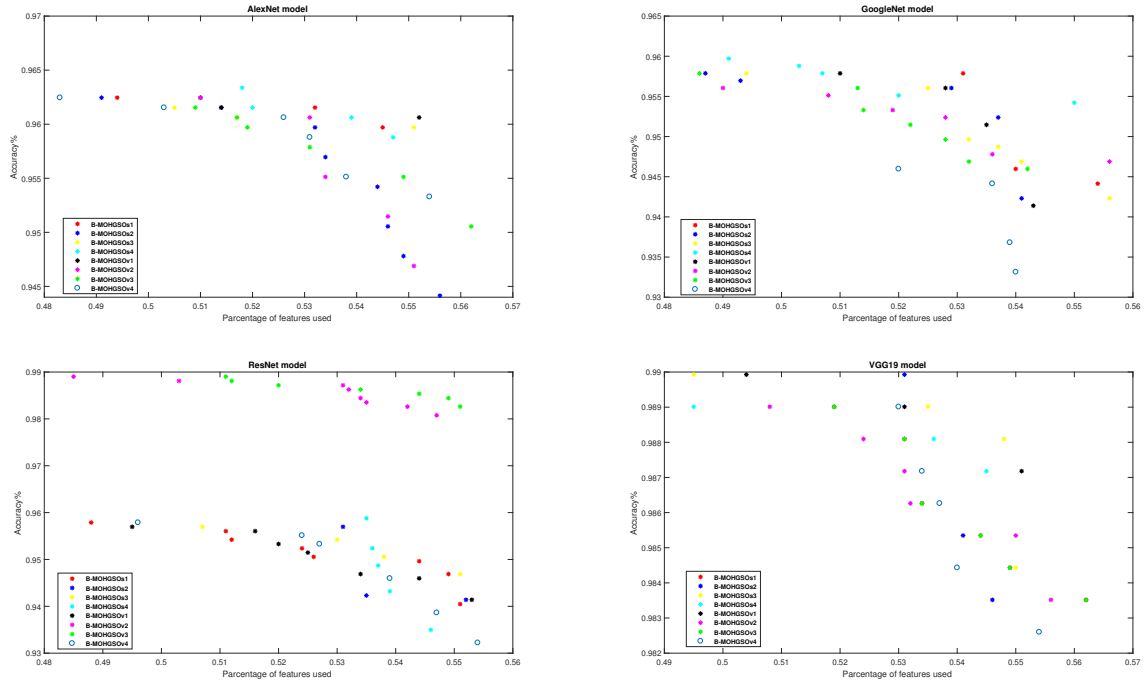


Figure 6.5: Comparison between the optimizers for AlexNet, GoogleNet, ResNet, and VGG19 models.

subsets.

Table 6.5: Average computational time (in minutes)

Algorithm:	B-MOHGSOs1	B-MOHGSOs2	B-MOHGSOs3	B-MOHGSOs4	B-MOHGSOv1	B-MOHGSOv2	B-MOHGSOv3	B-MOHGSOv4
AlexNet	15,08	9,28	**8,9**	15,08	14,67	15,5	14,9	35,67
GoogleNet	120	120.9	30.6	30.47	33.33	32.77	30.6	**30.37**
ResNet	66.33	30.03	133.3	30.3	**29.43**	117.6	29.97	29.67
VGG19	31.67	31.83	31.83	36.33	15.91	16.11	15.96	**14.88**

Table 6.5 shows that the computational time varies depending on the deep learning model and the version of the algorithm used. Both the complexity and the structure of the deep learning model, and the selected algorithm significantly impact the runtime. The results show that the computational runtime varies for each combination of the deep learning model and B-MOHGSO algorithm version selected.

For the AlexNet model, the computational runtime ranges from 8,9 to more than 35 minutes which were achieved by each of the MOHGSOs3 and B-MOHGSOv4, respectively. While the computational runtime for the ResNet model ranges from 29,43 to 133,3 minutes which were achieved by each of the B-MOHGSOv1 and the B-MOHGSOs3, respectively. The higher computational runtime for the B-MOHGSO with V-shaped transfer functions in some cases may be worth it to achieve better performance in terms of accuracy and feature selection as explained by Tables 7 and 8. For the GoogleNet and the VGG19 models, the computational runtime for the B-MOHGSO algorithm with S-shaped transfer functions is generally higher than that for models with V-shaped transfer functions.

However, it's significant to note that the runtime of an algorithm is attached and dependent on the deep learning model architecture (complexity and structure), the specific transfer function, and dataset characteristics used for feature selection.

In fact, with an appropriate transfer function, the MOHGSO algorithm can explore and examine the search space more efficiently and search the binary feature space that represents the most effective features to optimize the model's performance (increasing accuracy, reducing dimensionality).

Although these eight binary versions of MOHGSO have exhibited a strong capacity for solving multi-objective feature selection, consistent with the No-Free-Lunch (NFL) theorem in optimization, underscoring the inherent limitations of any optimization method, meaning there is no optimization algorithm can guarantee the same performance for all types of problems [140].

In conclusion, the analysis reveals certain limitations of the B-MOHGSO algorithm's performance in generating diverse non-dominated feature subsets including the identified mechanisms within the algorithm such as crowding distance used in the sorting of the archive, and sensitivity of the algorithm's parameters, tuning these parameters is often necessary to fit the type of transfer function selected. Obviously, more effective techniques can be combined and added to the suggested algorithm that need to be studied to identify transfer functions that enhance the algorithm's performance for multi-objective feature selection.

6.4 Comparative Analysis of Classifier Impact on B-MOHGSO Feature Selection

This section explores the application of the MOHGSO algorithm for multi-objective feature selection in the context of COVID-19 diagnosis from medical images containing 1,000 features extracted from the VGG19 deep-learning model. By leveraging these advanced techniques, we aim to identify the most informative features that maximize diagnostic accuracy while minimizing the number of selected features. This study proposed two binary versions of the Multi-Objective HGSO algorithm, termed B-MOHGSO1 and B-MOHGSO2, which are Pareto-based algorithms. B-MOHGSO1 and B-MOHGSO2 employ Grid Mechanism and Crowding Distance as a density operator, respectively, to find and identify the most and least populated segments of the Pareto front. To discover the best combination of features that suit a predetermined classifier, three factors are considered during the optimization process: K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) as evaluators, classification accuracy as an evaluation criterion for the selected feature subset quality, and the ability of B-MOHGSO as a searching method to achieve a significant balance between the two opposing objectives.

This work investigates how different classifiers (such as K-NN and SVM) influence the feature subsets selected by B-MOHGSO, potentially affecting the algorithm's perfor-

mance. Here's an explanation of how the two versions of the multi-objective approach called the Multi-Objective Henry Gas Solubility Optimization (MOHGSO) work:

- **Initialization:** The process begins with a random population of candidate feature subsets. Each subset represents a potential solution to the feature selection problem.
- **Evaluation:** Each candidate subset is evaluated using a classifier, such as K-nearest neighbor (K-NN) or support vector machine (SVM). The accuracy of these classifiers on the data subset serves as a primary objective function, indicating the quality of the feature subset.
- **Incorporating Diversity:** B-MOHGSO considers the diversity of the candidate feature subsets to avoid premature convergence to suboptimal solutions and encourages exploration of the feature space. The diversity is measured in two ways:
    1. Crowding distance quantifies how close a particular feature subset is to its neighbors in the search space. Feature subsets with larger crowding distances are preferred as they represent more unique combinations of features.
    2. Grid mechanism ensures that the selected features are well-distributed across the entire dataset, preventing overconcentration in specific areas.
- The evaluation process is repeated for a predetermined number of iterations until a final subset of features is selected, achieving a balance between maximizing the classification accuracy and maintaining diversity, this enhances the robustness of the classification model.
- Selected features are classified using two powerful machine-learning algorithms k-nearest neighbors (k-NN) and support vector machines (SVM);
- Evaluate the impact of algorithms on classifier types like k-nearest neighbors (k-NN) and support vector machines (SVM).

Note that for the discrete binary search space, the solution is restricted to binary $\{0, 1\}$ values according to this chosen transfer function 6.4.1:

$$T(x) = |tanh(x)| \tag{6.4.1}$$

In this work, the K-Nearest Neighbor (k-NN) and Support Vector Machine (SVM) algorithms are employed as learning classifiers to maximize classification accuracy. A higher classification accuracy suggests that the selected features are relevant, non-redundant, informative, and directly related to the classification task.

1. K-NN algorithm (with k = 5) plays a crucial role when many features are irrelevant and redundant for the performance task, this algorithm is based on feature space similarity measurement, which classifies a new sample instance based on the majority of the K-nearest neighbor category.

Table 6.6: The algorithm's impact on the k-NN and SVM classifiers over classification accuracy metric.

Algorithm	min	max	mean	Std.
VGG19	(98,44%)			
B-MOHGSO1 (K-NN)	33,42%	98,35%	88,95%	1,53E-01
B-MOHGSO1 (SVM)	33,42%	98,08%	85,51%	2,02E-01
B-MOHGSO2 (K-NN)	98,17%	98,99%	98,60%	1,71E-03
B-MOHGSO2 (SVM)	98,87%	99,81%	**99,75%**	1,36-03

2. Support Vector Machine (SVM) is a powerful and strong supervised machine learning algorithm applied for classification (e.g., normal vs. abnormal X-ray) and regression. The motives for adopting it in this work are its capability to address high-dimensional data and can be adapted and extended for multi-class problems as well, making it a popular choice for many machine learning tasks.

6.4.1 Experimental results

The simulation study is carried out in MATLAB to study the potentiality of the B-MOHGSO for the feature selection task and the algorithm's impact on two classifiers k-NN (k=5) and SVM. Two binary MOHGSO algorithms are compared with each other in order to select the best one. In our experiments, we set 100 initial particles and the maximum number of iterations to 100, the local-archive size to 50, and the global-archive size to 100 particles. We executed 10 independent runs and reported the statistical results obtained. The comparisons are carried out using tabular, and graphical presentations.

Two B-MOHGSO algorithms simultaneously optimize two objectives during the search process: minimizing the number of selected features and maximizing the accuracy of the classifier (either K-NN or SVM) that will be evaluated using 10-fold cross-validation on the dataset, providing a comprehensive assessment of the model's stability across different subsets of the Covid-19 data.

In order to compare the use of K-Nearest Neighbors (K-NN) and Support Vector Machines (SVM) in conjunction with both B-MOHGSO algorithms for feature selection, classification accuracy and dimensionality reduction, should be considered. The comparison focuses on how well each algorithm performs in terms of accurately classifying data and reducing the number of features, which are two metrics for evaluating the effectiveness of feature selection methods combined with the selected classifiers.

Table 6.6 outlines the obtained classification accuracies expressed as percentages with standard deviations from the two binary multi-objective HGSO methods proposed on both k-NN and SVM clas- sifiers. The classification accuracies obtained using all features are displayed in parentheses, while the best "mean" accuracy achieved by each algorithm is marked in boldface.

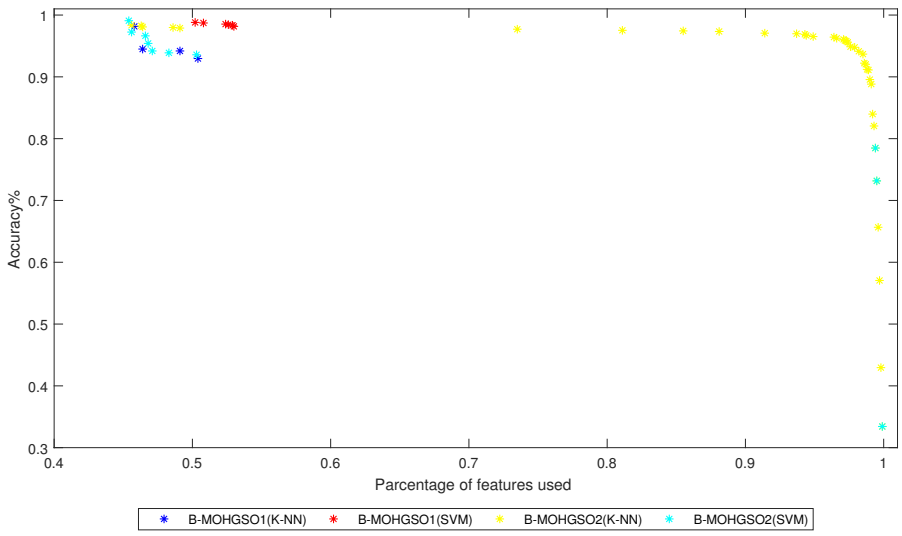


Figure 6.6: The obtained Pareto of the optimizers for VGG19 model.

The finding shows the performance of B-MOHGSO2 using the SVM classifier, with high accuracy ranging from 98,87% to 99,81% and a minimum standard deviation meaning its stability. These results indicates the performance of B-MOHGSO2 using the K-NN classifier, with accuracy ranging from 98,17% to 98,99% with a slightly higher standard deviation compared to the SVM classifier. The obtained standard deviation from B-MOHGSO1 using the SVM classifier show a variability in performance, an accuracy ranging from 33,42% to 98,08%.

Table 6.6 highlights the varying impact of different algorithms (B-MOHGSO1 and B-MOHGSO2) on the k-NN and SVM classifiers in terms of classification accuracy, with B-MOHGSO2, which uses the Crowding Distance mechanism, generally outperforming B-MOHGSO1 across both classifiers.

The obtained Pareto front from the competitor algorithms presented in Figure 6.6 shows the non-dominated feature subsets that balance accuracy and dimensionality. The shapes represent the obtained Pareto fronts from both algorithms in conjunction with two machine learning models used. It can be noted that the B-MOHGSO2 method, which utilized crowding distance with the SVM classifier, achieves high performance (close to 100% accuracy) with a small number of features. Additionally, some models may perform well with a limited selection of features, but their performance may stabilize or slightly decrease as more features are added. This is observed in the proposed B-MOHGSO1 method, which used the grid mechanism.

Table 6.7 shows the minimum, maximum, mean, and standard deviation for the percentage of selected features for each algorithm-classifier combination. B-MOHGSO1 with K-NN shows a wide range (45,60% to 99,90%) with a high mean of 89,38% and a relatively

Table 6.7: The algorithm's impact on the k-NN and SVM classifiers over the average of selected features.

Algorithm	min	max	mean	Std.
VGG19				
B-MOHGSO1 (K-NN)	45,60%	99,90%	89,38%	1,75E-01
B-MOHGSO1 (SVM)	45,40%	99,90%	62,89%	2,54E-01
B-MOHGSO2 (K-NN)	50,20%	53,00%	51,98%	1,18E-02
B-MOHGSO2 (SVM)	45,80%	50,40%	**47,93%**	2,19E-02

high standard deviation, while its performance with SVM has a similar range (45,40% to 99,90%) but a lower mean (62,89%) and higher standard deviation.

The performance of B-MOHGSO2 varies depending on the classifier used. With K-NN, it shows a narrow accuracy range of 50,20% to 53%, with a mean accuracy of 51,98% and a very low standard deviation. Using SVM, the accuracy range is also narrow, from 45,80% to 50,40%, but with a lower mean accuracy of 47,93% and a standard deviation of $\mathbf{2,19E-02}$, highlighting less variation in performance.

B-MOHGSO1 algorithm seems to select a higher percentage of features on average, especially with K-NN, and the B-MOHGSO2 algorithm is more consistent in its feature selection, with much lower variability across runs. The SVM classifier generally leads to fewer selected features comparing K-NN for both algorithms.

The proposed B-MOHGSO2 algorithm appears to be more effective at minimizing the number of selected features, especially when paired with SVM, while the proposed B-MOHGSO1 shows higher variability, which might indicate it's more sensitive to initial conditions or dataset characteristics.

To identify the most informative features for a machine learning model, the proposed B-MOHGSO2 uses the Crowding Distance mechanism in combination with a random selection of individuals (feature subsets). This approach maintains diverse solutions in non-dominated regions of the objective space, ensuring the exploration of different areas of the search space. This strategy helps prevent premature convergence on local optima and avoids getting stuck in regions with similar features. This results in a better balance between minimizing the number of selected features and maximizing classification accuracy. When combined with SVM classifiers, B-MOHGSO2 demonstrated high performance, achieving near-perfect accuracy with a small number of features.

Grid Mechanism employed in B-MOHGSO1, and divides the objective space into hypercubes, allowing for a structured exploration. This method helps identify the most and least populated segments of the Pareto front. The grid mechanism facilitates a comprehensive search, leading to well-distributed Pareto solutions. Although the B-MOHGSO1 algorithm using this approach showed stable and slightly decreased performance as more features were added, it still proved effective in balancing the dual objectives.

Both B-MOHGSO (Binary Multi-Objective Henry Gas Solubility Optimization) algo-

rithms can effectively select relevant features for both K-NN and SVM classifiers, optimizing the feature subset to improve classification performance. The choice of classifier (K-NN or SVM) influences the feature selection process and the resulting subset of features selected by the B-MOHGSO algorithms. SVM demonstrates its ability to handle the classification task and provides higher accuracy compared to K-NN in the high-dimensional Covid-19 dataset. While K-NN is a simple algorithm, it suffers from the curse of dimensionality in the Covid-19 dataset, impacting its accuracy. K-NN is a lazy learning algorithm that requires storing all training data and computing distances at test time, which can be computationally expensive for large datasets. In contrast, SVM is computationally efficient, scales well to high-dimensional data, tends to generalize well to unseen data, and is less prone to overfitting, making it suitable for classification task with high-dimensional data. K-NN may suffer from overfitting, especially in high-dimensional spaces, and requires careful selection of the "k" parameter (number of neighbors) to avoid bias in the classification results.

While both B-MOHGSO algorithms can effectively select features, the high dimensionality of the COVID-19 dataset favors SVM for better classification performance. This is due to SVM's ability to handle high dimensionality efficiently and avoid overfitting.

6.5 Conclusion

This work explores the potential of physics-inspired optimization techniques in machine learning by introducing the Binary Henry Gas Solubility Optimization (B-HGSO) algorithm, based on gas solubility principles for feature selection in classification tasks. The best-performing B-HGSO algorithm was applied to a COVID-19 chest X-ray dataset as an effective tool for feature selection, leveraging the power of GoogleNet and ResNet deep learning models for the feature extraction process. In this context, it was compared to B-PSO and B-GWO. This study highlights the effectiveness of B-HGSO in feature selection, particularly for medical image classification, offering a valuable tool for improving classification accuracy while reducing dimensionality.

Our research focuses on the development and application of binary versions of the Multi-Objective Henry Gas Solubility Optimization (B-MOHGSO) algorithm for feature selection, merged with state-of-the-art deep learning models and machine learning classifiers. This new approach aims to enhance the accuracy and efficiency of COVID-19 diagnosis while addressing the complexities inherent in high-dimensional medical imaging data. Experimental results demonstrate the ability of the proposed algorithms to achieve interesting results in terms of improving classification accuracy and reducing dimensionality.

In the second part of this chapter, we present the comparative analysis of both K-NN and SVM classifiers' impact on two versions of the B-MOHGSO algorithm. The study demonstrates the effectiveness of B-MOHGSO algorithms in feature selection for high-

dimensional datasets, specifically in the context of COVID-19 classification using features extracted from a VGG19 model. Firstly, the binary search space restriction using the Tanh transfer function ensures the algorithm's applicability to discrete feature selection problems. The choice of classifier (K-NN or SVM) significantly impacts the feature selection process and subsequent classification performance. Finally, SVM outperforms K-NN in this high-dimensional COVID-19 dataset, likely due to its better handling of the curse of dimensionality and resistance to overfitting.

Conclusion

During this work, we demonstrated that implementing a new metaheuristic algorithm named MOHGSO for multi-objective problems can significantly enhance the efficiency and effectiveness of solving complex optimization tasks. Extending this algorithm to feature selection leverages its strengths to identify optimal feature subsets, leading to improved performance in various applications such as machine learning and data mining. These results suggest that MOHGSO-based feature selection is a promising approach for supervised classification tasks in data mining.

The focal point of this thesis is the development and application of a new physical-based algorithm, the Henry Gas Solubility Optimization (HGSO) algorithm, originally designed for continuous optimization problems. This thesis presents significant contributions to multi-objective optimization and feature selection in classification tasks through the exploration of physics-inspired optimization techniques in machine learning by introducing the binary version of HGSO based on gas solubility principles.

Initially, we developed a population-based metaheuristic that simulates Henry's law for solving multi-objective optimization problems (MOPs) before applying it to specific data mining challenges. The performance of our approach relies on two critical factors: managing multi-external archives when new non-dominated solutions are found and selecting guiding individuals (multi-leaders) to direct their teammates toward promising regions, which is essential for balancing convergence (finding good solutions) and diversity (exploring different search areas).

The first part of this work focuses on exploring strategies for archive management and leader selection to enhance the performance and robustness of our proposed algorithm. Through the development of MOHGSO, we introduced a novel multi-objective optimization algorithm based on Henry's law that demonstrates effectiveness in continuous optimization problems. The performance heavily depended on two crucial elements:

1. Management of multi-external archives for new non-dominated solutions;

2. Selection of guiding individuals (multi-leaders) to direct teammates toward promising regions.

Comparative analyses within MOHGSO for MOPs—crowding distance with random selection versus grid mechanism with roulette wheel—revealed distinct strengths for each approach. For example, in ZDT functions, both strategies performed well, with crowding distance showing an advantage in complex variants like ZDT6, while the grid mechanism was more efficient for simpler ones like ZDT1. Similarly, DTLZ functions showed crowding distance's benefits for highly multimodal problems such as DTLZ3, while the grid mechanism suited less complex variants like DTLZ1. The UF suite highlighted crowding distance's robustness across varied complexities.
A comparative analysis evaluated MOHGSO against MOGWO and MSSA algorithms across four engineering design problems: Four-bar truss design, Speed reducer design, Disk brake design, and Welded beam design. Findings indicate our proposed algorithm outperforms its competitors across all problems.

The crowding distance with random selection strategy maintained diversity effectively in complex, high-dimensional problems but occasionally sacrificed convergence speed. Conversely, the grid mechanism with roulette wheel excelled in low-dimensional problems but struggled with high-dimensional variants due to increased computational costs. Comparative analysis of two strategies within MOHGSO revealed distinct strengths:

1. Crowding distance with random selection:
    - Demonstrated strong performance in maintaining diversity;
    - Excelled in complex, high-dimensional, or highly multimodal problems;
    - Showed flexibility across various problem types;
    - Occasionally sacrificed convergence speed for diversity.
2. Grid mechanism with roulette wheel:
    - Excelled in ensuring even distribution of solutions in low-dimensional problems;
    - Balanced diversity and convergence effectively for simpler MOPs;
    - Showed decreased efficiency as objectives increased due to computational demands.

To address discrete search space challenges, we adapted the HGSO algorithm for discrete optimization and further optimized the b-HGSO algorithm for complex datasets in the medical field. We developed a Binary-Multi-Objective HGSO (B-MOHGSO), specifically designed for feature selection applied to COVID-19 classification tasks.

The binary HGSO utilizes principles of gas solubility to tackle discrete optimization challenges effectively by balancing exploration and exploitation to achieve high classification accuracy through optimized feature subsets. Eight binary versions of HGSO were

developed to evaluate their performance in selecting optimal feature subsets that enhance classification accuracy while minimizing features. The binary versions explored different transfer functions:

1. S-shaped transfer functions:

- Provided smooth, gradual transitions between exploration and exploitation;
- Facilitated a gradual shift from exploration to exploitation, leading to stable convergence;
- Resulted in incremental changes to feature subsets.

2. V-shaped transfer functions:

- Offered abrupt transitions with sharp change points between exploration and exploitation;
- Promoted more exploration at first, followed by rapid exploitation;
- Led to faster convergence but with potentially higher instability or variability in feature subset changes.

The rapid spread of COVID-19 necessitated efficient diagnostic tools in healthcare. Machine learning has emerged as a crucial technology for early detection and classification of COVID-19 cases. However, high dimensionality often hampers model performance. Feature selection plays a vital role in enhancing classifier accuracy by identifying relevant features from datasets.

Our proposed method aims to improve classification accuracy while minimizing features used in COVID-19 datasets. Experiments demonstrate that the best-performing b-HGSO algorithm outperforms existing algorithms regarding efficiency and selecting fewer features while maintaining high classification accuracy.

The second part of this work focuses on extending the Multi-Objective HGSO (MO-HGSO) algorithm into a novel Binary-Multi-Objective HGSO (B-MOHGSO), designed specifically for discrete optimization problems like feature selection. This adaptation aims to improve B-MOHGSO's ability to explore promising regions within search spaces while reducing population size and identifying diverse, high-quality feature subsets.

B-MOHGSO incorporates eight different binary versions to investigate various transfer functions' impact on performance in wrapper feature selection. A comprehensive study on S-shaped and V-shaped transfer functions reveals their effects on navigating discrete search spaces effectively, merged with state-of-the-art deep learning models and machine learning classifiers. This novel approach aims to enhance the accuracy and efficiency of COVID-19 diagnosis while addressing the complexities inherent in high-dimensional medical imaging data. The primary objectives of the second part of this study are:

- Develop and compare eight binary versions of the MOHGSO algorithm, exploring both S-shaped and V-shaped transfer functions for multi-objective feature selection;
- Evaluate the performance of these algorithms when combined with different deep learning models (AlexNet, GoogleNet, ResNet, and VGG19) for feature extraction from chest X-ray images;
- Assess the impact of different classifiers, specifically K-Nearest Neighbors (KNN) and Support Vector Machines (SVM), on the feature selection process and overall classification accuracy;
- Determine the most effective combination of feature selection method, deep learning model, and classifier for accurate COVID-19 diagnosis.

In the context of COVID-19 diagnosis, our research yielded several significant findings:

1. V-shaped transfer functions unexpectedly outperformed S-shaped functions:
    - Achieved better exploration;
    - Attained higher classification accuracy with fewer features;
    - This finding contradicted general assumptions about S-shaped functions.
2. Key factors influencing performance:
    - Dataset characteristics: the COVID-19 dataset may have unique properties that favor well with V-shaped functions' behavior.
    - Feature correlation: a high correlation among features could make the sharp transitions of V-shaped functions more effective at identifying the most informative features.
    - Parameter settings: the specific parameter settings of B-MOHGSO might have been particularly favorable for V-shaped functions in this context.

Finally, this work explores how different classifiers, such as K-Nearest Neighbors (KNN) and Support Vector Machines (SVM), influence the feature subsets selected by the Binary Multi-Objective Henry Gas Solubility Optimization (B-MOHGSO) algorithm, potentially affecting our algorithm's performance. These classifiers were chosen due to their established efficacy in handling classification tasks, particularly in medical imaging contexts. The KNN classifier, known for its simplicity and effectiveness in high-dimensional spaces, performed well with the selected feature subsets. However, the SVM classifier outperformed KNN in this context, achieving higher accuracy and better generalization on unseen data. This highlights the importance of classifier selection in the feature selection process, as different classifiers may respond differently to the features identified by the MOHGSO algorithm.

The dual focus on using features extracted from the VGG19 model and V-shaped transfer function within the B-MOHGSO algorithm highlight the significant influence of classifier choice on feature selection process and overall algorithm performance. The results of the evaluation demonstrated that the proposed method significantly enhances classification performance for COVID-19 detection. Specifically, the study demonstrates that when the B-MOHGSO algorithm is paired with the SVM classifier, it outperforms the K-NN classifier in terms of classification accuracy and efficiency. SVM is better suited for this application compared to K-NN with high-dimensional data and noise present in medical imaging. Important and main findings from our research include:

- The superiority of B-MOHGSO algorithms, particularly those employing V-shaped transfer functions, in selecting relevant features while maintaining high classification accuracy;
- The V-shaped transfer function is particularly effective in guiding the optimization process, allowing for a more decisive transition between feature inclusion and exclusion;
- The exceptional performance of the VGG19 model when combined with B-MOHGSO and the SVM classifier, achieving a remarkable accuracy while significantly reducing the feature set.
- The critical impact of choosing appropriate feature selection methods, transfer functions, and classifiers when dealing with high-dimensional medical imaging data;
- The crucial role of feature extraction, optimization feature selection algorithm, and classifier choice in enhancing diagnostic accuracy in healthcare;
- Emphasizing the need for specialized optimization strategies that take into account the nature of the data and classifiers used.

Finally, we acknowledge that there is extent for further research and improvement. Future research directions include:

1. Exploring MOHGSO feature selection with other machine learning algorithms beyond K-NN and SVM;
2. Comparing B-MOHGSO with other state-of-the-art feature selection algorithms;
3. Investigating parameter settings within MOHGSO for performance optimization;
4. Extending the approach to other medical imaging modalities and diseases;
5. Studying the relationship between dataset characteristics and transfer function selection.

This work makes significant contributions to machine learning and optimization research, particularly in developing effective feature selection techniques for improved data mining applications. The proposed B-MOHGSO algorithm aims to identify relevant features efficiently, addressing high-dimensional medical imaging data's complexities while enhancing diagnostic accuracy and efficiency in clinical settings. The successful application to COVID-19 diagnosis demonstrates its potential impact on healthcare systems worldwide, offering promising tools for managing future health crises.

Bibliography

[1] Cme307/msande311: Optimization, 2024.

[2] Safinaz AbdEl-Fattah Sayed, Emad Nabil, and Amr Badr. A binary clonal flower pollination algorithm for feature selection. *Pattern Recognition Letters*, 77:21–27, 2016.

[3] B. Abdollahzadeh and F. S. Gharehchopogh. A multi-objective optimization algorithm for feature selection problems. *Engineering with Computers*, 2021.

[4] G. Adel, M. Abdelouahab, and Z. Djaafar. A guided population archive whale optimization algorithm for solving multi-objective optimization problems. *Expert Syst Appl*, 141:112972, 2020.

[5] C.C. Aggarwal. *Data classification: algorithms and applications*. CRC press, 2014.

[6] R.K. Agrawal and N.G. Bawane. Multiobjective pso based adaption of neural network topology for pixel classification in satellite imagery. *Applied Soft Computing*, 28:217–225, 2015.

[7] Q. Al-Tashi, S.J. Abdulkadir, H.M. Rais, S. Mirjalili, H. Alhussian, M.G. Ragab, and A. Alqushaibi. Binary multi-objective grey wolf optimizer for feature selection in classification. *IEEE Access*, 8:106247–106263, 2020.

[8] Praveen Anand and Sanjeev Arora. A novel chaotic selfish herd optimizer for global optimization and feature selection. *Artificial Intelligence Review*, 53:1441–1486, 2020.

[9] Shaeela Ayesha, Muhammad Kashif Hanif, and Ramzan Talib. Overview and comparative study of dimensionality reduction techniques for high dimensional data. *INF. FUSION*, 2020.

[10] Ricardo Landa Becerra and Carlos A. Coello Coello. Solving hard multiobjective optimization problems using ε-constraint with cultured differential evolution. In T.P. Runarsson, H.-G. Beyer, E. Burke, J.J. Merelo-Guervós, L.D. Whitley, and X. Yao, editors, *Parallel Problem Solving from Nature - PPSN IX*, volume 4193 of *Lecture Notes in Computer Science*, Berlin, Heidelberg, 2006. Springer.

[11] C. L. Blake and CJ Merz. Uci repository of machine learning databases, 1998. Department of Information and Computer Science, University of California, Irvine, CA. [Online]. Available http://www. ics. uci. edu/mlearn/MLRepository. html.

[12] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35:268–308, 2003.

[13] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, 1992.

[14] A. Bouraoui, S. Jamoussi, and Y. BenAyed. A multi-objective genetic algorithm for simultaneous model and feature selection for support vector machines. *Artificial Intelligence Review*, 50(2):261–281, 2018.

[15] Hyeran Byun and Seong-Whan Lee. Applications of support vector machines for pattern recognition: A survey. *International Workshop on Support Vector Machines*, pages 213–236, 2002.

[16] L.C. Cagnina, S.C. Esquivel, and C.A. Coello. Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Informatica*, 32:319–326, 2008.

[17] Murat Canayaz. Mh-covidnet: Diagnosis of covid-19 using deep neural networks and meta-heuristic-based feature selection on x-ray images. *Biomedical Signal Processing and Control*, 64:102257, 2021.

[18] G. Chandrashekar and F. Sahin. A survey on feature selection methods. *Computers and Electrical Engineering*, 40(1):16–28, 2014.

[19] Hamouda K. Chantar, Majdi Mafarja, Hamad Alsawalqah, Amir A. Heidari, Ibrahim Aljarah, and Hossam Faris. Feature selection using binary grey wolf optimizer with elite-based crossover for arabic text classification. *Neural Computing and Applications*, 2019.

[20] S.-N. Chegini, A. Bagheri, and F. Najafi. Psoscalf: A new hybrid pso based on sine cosine algorithm and levy flight for solving optimization problems. *Appl Soft Comput*, 73:697–726, 2018.

[21] Y. Chen and Y. Hao. A feature weighted support vector machine and k-nearest neighbor algorithm for stock market indices prediction. *Expert Systems with Applications*, 80:340–355, 2017.

[22] K. Chomboon, P. Chujai, P. Teerarassamee, K. Kerdprasop, and N. Kerdprasop. An empirical study of distance metrics for k-nearest neighbor algorithm. In *Proceedings of the 3rd International Conference on Industrial Application Engineering*, 2015.

[23] S.-C. Chu, P.-W. Tsai, and J.-S. Pan. Cat swarm optimization. *Lecture Notes in Computer Science*, 4099:854–858, 2006.

[24] L-Y. Chuang, H-W. Chang, C-J. Tu, and C-H. Yang. Improved binary pso for feature selection using gene expression data. *Comput Biol Chem*, 32(1):29–38, 2008.

[25] L.Y. Chuang, S.W. Tsai, and C.H. Yang. Improved binary particle swarm optimization using catfish effect for feature selection. *Expert Systems with Applications*, 38(10):12699–12707, 2011.

[26] H. Chun-lin, Y. Zhang, and B. Wu. Multi-objective feature selection based on artificial bee colony for hyperspectral images. In *International Conference on Bio-Inspired Computing: Theories and Applications*. Springer, 2019.

[27] C.A.C. Coello, G.T. Pulido, and M.S. Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):256–279, 2004.

[28] C.A. Coello Coello and M.S. Lechuga. Mopso: a proposal for multiple objective particle swarm optimization. In *Proceeding of the 2002 Congress on Evolutionary Computation*, 2002.

[29] D. Corne, M. Dorigo, and F. Glover. *New Ideas in Optimization*. McGraw-Hill, London, UK, 1999.

[30] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[31] Y. Cui, Z. Geng, Q. Zhu, and Y. Han. Review: Multi-objective optimization methods and application in energy saving. *Energy*, 2017.

[32] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(1-4):131–156, 1997.

[33] K. Deb, A. Pratap, S. Agarwal, and T. Mayarivan. A fast and elitist multi-objective algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[34] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 825–830. IEEE, 2002.

[35] Kalyanmoy Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley and Son, New York, 2001.

[36] J. Derrac, S. García, D. Molina, and F. Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[37] G. Dhiman and V. Kumar. Multi-objective spotted hyena optimizer: a multi-objective optimization algorithm for engineering problems. *Knowledge-Based Systems*, 150:175–197, 2018.

[38] G. Dhiman and V. Kumar. Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*, 165:169–196, 2019.

[39] G. Dhiman, K.-K. Singh, A. Slowik, V. Chang, A.-R. Yildiz, A. Kaur, and M. Garg. Emosoa: a new evolutionary multi-objective seagull optimization algorithm for global optimization. *Machine Learning and Cybernetics*, 2020. early access.

[40] G. Dhiman, K.-K. Singh, M. Soni, A. Nagar, M. Dehghani, A. Slowik, A. Kaur, A. Sharma, E.-H. Houssein, and K. Cengiz. Mosoa: A new multi-objective seagull optimization algorithm. *Expert Systems with Applications*, 167:114150, 2020.

[41] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Machine Learning Proceedings*, pages 194–202. Elsevier, 1995.

[42] Nicolas Dupin. Integer linear programming reformulations for the linear ordering problem, 2022.

[43] Matthias Ehrgott, Xavier Gandibleux, and Arkadiusz Przybylski. *Exact Methods for Multi-Objective Combinatorial Optimisation*, volume 233 of *International Series in Operations Research and Management Science*. Springer, New York, NY, 2016.

[44] E. Emary, W. Yamany, A.E. Hassanien, and V. Snasel. Multi-objective grey wolf optimization for attribute reduction. In *International Conference on Communications, Management, and Information Technology (ICCMIT'2015)*.

[45] Eid Emary, Hossam M. Zawbaa, and Aboul Ella Hassanien. Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, 172:371–381, 2016.

[46] J. Fakanda. *Contribution aux Methodes d'Optimisation Combinatoire Multi-Objectif*. Edition universitaire européennes, 2016.

[47] M.P. Fourman. Compaction of symbolic layout using genetic algorithms. In *Proceeding of the first international conference on genetic algorithms*, pages 141–153, 1985.

[48] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. 2017.

[49] Narayanan Ganesh and S. Balamurugan. *Metaheuristics for Machine Learning: Algorithms and Applications*. Wiley, 2023.

[50] Manizheh Ghaemi and Mohammad Reza Feizi-Derakhshi. Feature selection using forest optimization algorithm. *Pattern Recognition*, 60:121–129, 2016.

[51] C. Gherardo, M. Matteo, M. Daniel, P. Nicolas, O.T. Erkki, T.W. Lars, et al. A meta-analysis and review of the literature on the k-nearest neighbors technique for forestry applications that use remotely sensed data. *Remote Sensing of Environment*, 176:282–294, 2016.

[52] Iffat A. Gheyas and Leslie S. Smith. Feature subset selection in large dimensionality domains. *Pattern Recognition*, 43(1):5–13, 2010.

[53] Kaushik Kanti Ghosh, Rajdeep Guha, Suman Kumar Bera, et al. S-shaped versus v-shaped transfer functions for binary manta ray foraging optimization in feature selection problem. *Neural Computing and Applications*, 33:11027–11041, 2021.

[54] M. et al. Ghosh. Feature selection using histogram-based multi-objective ga for handwritten devanagari numeral recognition. *Intelligent Engineering Informatics*, page 471–479, 2018.

[55] R. Goswami, D. K. Bhattacharyya, and M. Dutta. Multiobjective differential evolution algorithm using binary encoded data in selecting views for materializing in data warehouse. In *Swarm, Evolutionary, and Memetic Computing, Lecture Notes in Computer Science*, volume 8298, pages 95–106. Springer, 2013.

[56] Jiawei Han and Micheline Kamber. *Data mining: concepts and techniques*. Morgan Kaufmann, 2006.

[57] E. Hancer. A new multi-objective differential evolution approach for simultaneous clustering and feature selection. *Engineering Applications of Artificial Intelligence*, 87:103307, 2020.

[58] E. Hancer, B. Xue, and M. Zhang. Differential evolution for filter feature selection based on information theory and feature ranking. *Knowledge-Based Systems*, 140:103–119, 2018.

[59] E. Hancer, B. Xue, M. Zhang, D. Karaboga, and B. Akay. A multi-objective artificial bee colony approach to feature selection using fuzzy mutual information. In *Proceedings of IEEE Congress on the Evolutionary Computation (CEC)*, 2015.

[60] Emrah Hancer, Bing Xue, Dervis Karaboga, and Mengjie Zhang. A binary abc algorithm based on advanced similarity scheme for feature selection. *Applied Soft Computing Journal*, 36:334–348, 2015.

[61] M. P. Hansen. Tabu search for multiobjective optimization: Mots. Technical report, Institute of Mathematical Modeling, Technical University of Denmark, 1997.

[62] F.A. Hashim, E.H. Houssein, M.S. Mabrouk, W. Al-Atabany, and S. Mirjalili. Henry gas solubility optimization: A novel physics-based algorithm. *Future Gener Comput Syst*, 101:646–667, 2019.

[63] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

[64] Ying Hu, Yong Zhang, and Dunwei Gong. Multiobjective particle swarm optimization for feature selection with fuzzy cost. *IEEE Transactions on Cybernetics*, 51(2):874–888, 2021.

[65] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[66] Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, and Jong Wook Kim. Q-learning algorithms: A comprehensive classification and applications. *IEEE ACCESS*, 2019.

[67] P. Jangir and N. Jangir. A new non-dominated sorting grey wolf optimizer (ns-gwo) algorithm: Development and application to solve engineering designs and economic constrained emission dispatch problem with integration of wind power. *Engineering Applications of Artificial Intelligence*, 72:449–467, 2018.

[68] B. Ji, X. Lu, G. Sun, et al. Bio-inspired feature selection: An improved binary particle swarm optimization approach. *IEEE Access*, 8:85989–86002, 2020.

[69] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. 1996.

[70] A. Kamalova, S. Navruzov, D. Qian, and S.G. Lee. Multi-robot exploration based on multi-objective grey wolf optimizer. *Applied Sciences*, 9(14):2931, 2019.

[71] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[72] D. Klein and E. Hannen. An algorithm for multiple objective integer linear programming problem. *European Journal of Operational Research*, 9:378–385, 1982.

[73] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[74] P. Krömer, J. Platoš, J. Nowaková, and V. Snášel. Optimal column subset selection for image classification by genetic algorithms. *Ann Oper Res*, 265:205–222, 2018.

[75] A. Kurpati, S. Azarm, and J. Wu. Constraint handling improvements for multiobjective genetic algorithms. *Struct Multidiscip Optim*, 23:204–213, 2002.

[76] M. Labani, P. Moradi, and M. Jalili. A multi-objective genetic algorithm for text feature selection using the relative discriminative criterion. *Expert Systems with Applications*, 149:113276, 2020.

[77] Y.J. Lai and C.L. Hwang. *Fuzzy Mathematical Programming: methods and applications*, volume 394 of *Lecture in Economics and Mathematical Systems*. Springer-Verlag, Berlin, 1992.

[78] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation*, 10(3):263–282, 2002.

[79] L. Leifsson and S. Koziel. Multi-fidelity design optimization of transonic airfoils using physics-based surrogate modeling and shape-preserving response prediction. *J Comput Sci*, 1:98–106, 2010.

[80] M. Li, H. Xu, X. Liu, and S. Lu. Emotion recognition from multichannel eeg signals using k-nearest neighbor classification. In *6th International Conference on Biomedical Engineering and Biotechnology (ICBEB2017)*, pages 509–519. IOS Press, 2017.

[81] J. G. Lin. On min-norm and min-max methods of multi-objective optimization. *Mathematical Programming*, 103(1):1–33, 2005.

[82] M. Mafarja, I. Aljarah, A.A. Heidari, H. Faris, P. Fournier-Viger, X. Li, and S. Mirjalili. Binary dragonfly optimization for feature selection using time-varying transfer functions. *Knowledge-Based Systems*, 161:185–204, 2018.

[83] Majdi Mafarja, Ibrahim Aljarah, Hossam Faris, Ahmad I Hammouri, Ahmad M Al-Zoubi, and Seyedali Mirjalili. Three improvement algorithms based on whale optimization algorithm. *Soft Computing*, 22(19):6501–6529, 2018.

[84] Majdi Mafarja, Derar Eleyan, Salwani Abdullah, and Seyedali Mirjalili. S-shaped vs. v-shaped transfer functions for ant lion optimization algorithm in feature selection problem. *Networks and Distributed Systems*, 2017(6):6, 2017.

[85] Majdi M. Mafarja and Seyedali Mirjalili. Hybrid whale optimization algorithm with simulated annealing for feature selection. *Applied Soft Computing*, 62:817–827, 2018.

[86] Xavier Meyer, Bastien Chopard, and Paul Albuquerque. A branch-and-bound algorithm using multiple gpu-based lp solvers. pages 129–138, 12 2013.

[87] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, and S.M. Mirjalili. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, 114:163–191, 2017.

[88] S. Mirjalili, P. Jangir, and S. Saremi. Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems. *Applied Intelligence*, 46:79–95, 2017.

[89] S. Mirjalili and A. Lewis. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9(4):1–14, 2013.

[90] S. Mirjalili and A. Lewis. The whale optimization algorithm. *Advances in Engineering Software*, 95:51–67, 2016.

[91] S. Mirjalili, S.M. Mirjalili, and A. Lewis. Grey wolf optimizer. *Advances in Engineering Software*, 69:46–61, 2014.

[92] S.A. Mirjalili, S. Sarem, S.M. Mirjalili, and L.S. Coelho. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Systems With Applications*, 47:106–119, 2016.

[93] Seyed Mohammad Mirjalili and Andrew Lewis. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm Evol. Comput.*, 9:1–14, 2013.

[94] P. Moradi and M. Gholampour. A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. *Applied Soft Computing*, 43:117–130, 2016.

[95] M.A. Munoz, L. Villanova, D. Baatar, and K. Smith-Miles. Instance spaces for machine learning classification. *Machine Learning*, 107(1):109–147, 2018.

[96] Kevin P. Murphy. *Machine learning: A probabilistic perspective*. MIT press, 2012.

[97] I.S. Oh, J.S. Lee, and B.R. Moon. Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1424–1437, 2004.

[98] S. Park, H. Yoo, and S. Park. Multi-style license plate recognition system using k-nearest neighbors. *KSII Transactions on Internet and Information Systems*, 13(5):2509–2528, 2019.

[99] João H. R. D. Vasconcelos João A. Parreiras, Roberta O. Maciel. The a posteriori decision in multiobjective optimization problems with smarts, promethee ii, and a fuzzy algorithm. *IEEE Transactions on Magnetics*, 42(4):1139–1139, 2006.

[100] S. Paul and S. Das. Simultaneous feature selection and weighting – an evolutionary multi-objective optimization approach. *Pattern Recognition Letters*, 65:51–59, 2015.

[101] Huijun Peng, Chun Ying, Shuhua Tan, Bing Hu, and Zhixin Sun. An improved feature selection algorithm based on ant colony optimization. *IEEE Access*, 6:69203–69209, 2018.

[102] Suruchi Pimple. Application of support vector machine for diagnosis of diabetes: A systematic review. *IOSR Journal of Computer Engineering (IOSRJCE)*, 2016.

[103] P.M. Pradhan and G. Panda. Solving multi objective problems using cat swarm optimization. *Expert Systems with Applications*, 39(3):2956–2964, 2012.

[104] M. Premkumar, P. Jangir, and R. Sowmya. Mogbo: A new multiobjective gradient-based optimizer for real-world structural optimization problems. *Knowledge-Based Systems*, 218:106856, 2021.

[105] M. Premkumar, P. Jangir, R. Sowmya, H.-H. Alhelou, A.-A. Heidari, and H. Chen. Mosma: Multi-objective slime mould algorithm based on elitist non-dominated sorting. *IEEE Access*, 2021.

[106] Z. Ramdani. Cours de programmation linéaire. 2017.

[107] S. Rathee and S. Ratnoo. Feature selection using multi-objective chc genetic algorithm. *Procedia Computer Science*, 167:1656–1664, 2020.

[108] T. Ray and K.M. Liew. A swarm metaphor for multi-objective design optimization. *Eng Optim*, 34:141–153, 2002.

[109] M. Reyes-Sierra and C.A.C. Coello. Multi-objective particle swarm optimizers: a survey of the state-of-the-art. *Int J Comput Intell Res*, 2(3):287–308, 2006.

[110] Mehrdad Rostami, Saman Forouzandeh, Kamal Berahmand, and Mehdi Soltani. Integration of multiobjective pso based feature selection and node centrality for medical datasets. *Genomics*, 112(6):4370–4384, 2020.

[111] A. Sahoo and S. Chandra. Multi-objective grey wolf optimizer for improved cervix lesion classification. *Applied Soft Computing*, 52:64–80, 2017.

[112] B. Schneier. *Cryptographie appliquée: Protocoles, algorithmes et codes sources en C*. International Thomson Publishing, Paris, 2 edition, 1997.

[113] J.R. Schott. Fault tolerant design using single and multicriteria genetic algorithm optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, 1995.

[114] B. Schölkopf, A.J. Smola, F. Bach, et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[115] Shokri Z. Selim and Mohamed A. Ismail. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on pattern analysis and machine intelligence*, (1):81–87, 1984.

[116] V. Sharma, S. Kumar. A comprehensive review on multi-objective optimization techniques: Past, present and future. *Archives of Computational Methods in Engineering*, 29(7):5605–5633, 2022.

[117] X. Shi and D. Kong. A multi objective ant colony optimization algorithm based on elitist selection strategy. 2015.

[118] P. Shunmugapriya and S. Kanmani. A hybrid algorithm using ant and bee colony optimization for feature selection and classification (ac-abc hybrid). *Swarm and Evolutionary Computation*, 36(4):27–36, 2017.

[119] M.R. Sierra and C.A.C. Coello. Improving pso-based multi-objective optimization using crowding, mutation and ϵ-dominance. In *Proceedings of the Evolutionary Multi-Criterion Optimization*, 2005.

[120] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[121] Narinder Singh and S. B. Singh. Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance. *Journal of Applied Mathematics*, 2017:1–15, 2017.

[122] Mohsen K. Sohrabi and Amir Tajik. Multi-objective feature selection for warfarin dose prediction. *Computational Biology and Chemistry*, 69:126–133, 2017.

[123] Gholamreza Fazel Soleimanian, Norbik Basri Isa, and Zahra Azmi Dizaji. Chaotic vortex search algorithm: metaheuristic algorithm for feature selection. *Evolving Intelligence*, 14:1777–1808, 2021.

[124] Ghada M. A. Soliman, Tarek H. M. Abou-El-Enien, E. Emary, and Motaz M. H. Khorshid. A novel multi-objective moth-flame optimization algorithm for feature selection. *Indian Journal of Science and Technology*, 11(38), 2018.

[125] N. Srinivas and K. Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evol Comput*, 2:221–248, 1995.

[126] Nasser H. Sweilam, AA Tharwat, and NK Abdel Moniem. Support vector machine for diagnosis cancer disease: A comparative study. *Egyptian Informatics Journal*, 11(2):81–92, 2010.

[127] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, Boston, MA, 2015.

[128] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to data mining*. Addison-Wesley, 2006.

[129] J. Teghem. *Recherche operationnelle, volume 1: Methodes d'optimisation*. Ellipses, 2012.

[130] J. Teghem. *Recherche opérationnelle : volume 1 : Méthodes d'optimisation*. Ellipses, 2012.

[131] T. Thaher, A.A. Heidari, M. Mafarja, J.S. Dong, and S. Mirjalili. Binary harris hawks optimizer for high-dimensional, low sample size feature selection. In *Evolutionary Machine Learning Techniques*, pages 251–272. Springer, 2020.

[132] J. Too and A.R. Abdullah. A new and fast rival genetic algorithm for feature selection. *J Supercomput*, pages 1–31, 2020.

[133] Qiang Tu, Xuechen Chen, and Xingcheng Liu. Hierarchy strengthened grey wolf optimizer for numerical optimization and feature selection. *IEEE Access*, 7:78012–78028, 2019.

[134] B.E.L. Ulungo. *Optimisation combinatoire multicritère : génération des solutions efficaces et méthodes itératives*. PhD thesis, Université de Mons, 1993.

[135] E. Ulungo and J. Tegham. Multi-objective combinatorial optimization problems: A survey. *Journal of Multi-Criteria Decision Analysis*, 3:83–104, 1994.

[136] B.E.L. Ulungu and J. Teghem. The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of computing and decision sciences*, 20(2):149–165, 1994.

[137] Youchuan Wan, Mingwei Wang, Zhiwei Ye, and Xudong Lai. A feature selection method based on modified binary coded ant colony optimization algorithm. *Applied Soft Computing*, 49:248–258, 2016.

[138] X.-H. Wang et al. Multi-objective feature selection based on artificial bee colony: an acceleration approach with variable sample size. *Applied Soft Computing*, 88:106041, 2020.

[139] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

[140] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1:67–82, 1997.

[141] B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang. Multi-objective evolutionary algorithms for filter based feature selection in classification. *International Journal on Artificial Intelligence Tools*, 22:1350024, 2013.

[142] B. Xue, W. Fu, and M. Zhang. Multi-objective feature selection in classification: A differential evolution approach. In *Lecture Notes in Computer Science*, pages 516–528. Springer International Publishing, 2014.

[143] B. Xue, M. Zhang, W. N. Browne, and X. Yao. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, 20(4):606–626, 2016.

[144] B. Xue, M. Zhang, and W.N. Browne. Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. *Applied Soft Computing*, 18:261–276, 2014.

[145] Bing Xue, Mengjie Zhang, and Wesley N. Browne. Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE Transactions on Cybernetics*, 43(6):1656–1671, 2013.

[146] Y. Xue, T. Tang, W. Pang, and A. X. Liu. Self-adaptive parameter and strategy based particle swarm optimization for large-scale feature selection problems with multiple classifiers. *Applied Soft Computing*, 88, 2020.

[147] Chaokun Yan, Jingjing Ma, Huimin Luo, and Ashutosh Patel. Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets. *Chemometrics and Intelligent Laboratory Systems*, 15:1777–1808, 2022.

[148] X.-S. Yang, M. Karamanoglu, and X. He. Multi-objective flower algorithm for optimization. *Procedia Computer Science*, 18:861–868, 2013.

[149] X.S. Yang. Multi-objective firefly algorithm for continuous optimization. *Engineering with Computers*, 29(2):175–184, 2013.

[150] Z. Yong, G. Dun-Wei, and Z. Wan-Qiu. Feature selection of unreliable data using an improved multi-objective pso algorithm. *Neurocomputing*, 171:1281–1290, 2016.

[151] M. Zamalloa, G. Bordel, L.J. Rodríguez, and M. Penagarikano. Feature selection based on genetic algorithms for speaker recognition. In *IEEE Odyssey 2006: Workshop on Speaker and Language Recognition*, pages 1–8. IEEE, 2006.

[152] Hossam M. Zawbaa, E. Emary, Crina Grosan, and Vaclav Snasel. Large-dimensionality small-instance set feature selection: A hybrid bio-inspired heuristic approach. *Swarm and Evolutionary Computation*, 42:29–42, 2018.

[153] Q. Zhang, A. Zhou, S. Zhao, P.N. Suganthan, W. Liu, and S. Tiwari. Multiobjective optimization test instances for the cec 2009 special session and competition. Technical Report 264, University of Essex and Nanyang Technological University, 2008. Special session on performance assessment of multi-objective optimization algorithms.

[154] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[155] Xiaojie Zhang, Yuxin Xu, Chen Yu, and et al. Gaussian mutational chaotic fruit fly-built optimization and feature selection. *Expert Systems with Applications*, 141, 2020.

[156] Y. Zhang et al. Cost-sensitive feature selection using two archive multi-objective artificial bee colony algorithm. *Expert Systems with Applications*, 137:46–58, 2019.

[157] Y. Zhang, D.-w. Gong, X.-z. Gao, T. Tian, and X.-y. Sun. Binary differential evolution with self-learning for multi-objective feature selection. *Information Sciences*, 507:67–85, 2020.

[158] Y. Zhang, D.W. Gong, X.Y. Sun, and Y.N. Guo. A pso-based multi-objective multilabel feature selection method in classification. *Scientific Reports*, 7, 2017.

[159] Yanan Zhang, Renjing Liu, Xin Wang, and et al. Boosted binary harris hawks optimizer and feature selection. *Engineering with Computers*, 2020.

[160] Yong Zhang, Dun-wei Gong, and Jian Cheng. Multi-objective particle swarm optimization approach for cost-based feature selection in classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 14(1):64–75, 2017.

[161] Yong Zhang, Chun-lin He, Xian-fang Song, and Xiao-yan Sun. A multi-strategy integrated multi-objective artificial bee colony for unsupervised band selection of hyperspectral images. *Swarm and Evolutionary Computation*, 60:100806, 2021.

[162] Y. Zhou, J. Kang, S. Kwong, X. Wang, and Q. Zhang. An evolutionary multiobjective optimization framework of discretization-based feature selection for classification. *Swarm and Evolutionary Computation*, 60, 2021.

[163] E. Zitzler, K. Deb, and L. Thiele. Comparison of multi-objective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8:173–195, 2000.

[164] E. Zitzler, M. Laumanns, and S. Bleuler. A tutorial on evolutionary multi-objective optimization. In *Workshop on Multiple Objective Metaheuristics (MOMH)*. Springer-Verlag, Berlin, 2004.

[165] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm for multi-objective optimization. In K.C. Giannakoglou, D.T. Tsahalis, J. Périaux, K.D. Papailiou, and T. Fogarty, editors, *Proc. Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece.

[166] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(2):257–271, 1999.

[167] Djaafar Zouache. *Fouille de données basée algorithmes bio-inspirés*. PhD thesis, Université Ferhat Abbas Sétif 1, 2016.

Abstract

The ability to extract knowledge from large datasets is essential for innovation and informed decision-making, a process known as knowledge extraction or data mining. Traditional methods often fall short in fully utilizing data potential, necessitating the development of new algorithms for better insights.

This thesis explores an innovative approach by integrating deep learning with advanced feature selection techniques to improve the classification accuracy of COVID-19 cases from chest X-ray images. The dataset includes X-ray images categorized as COVID-19, pneumonia, and normal. We employ the Binary Multi-Objective Henry Gas Solubility Optimization Algorithm (B-MOHGSO) for feature selection and leverage models like AlexNet, VGG19, GoogleNet, and ResNet for feature extraction. Eight versions of B-MOHGSO were tested, with k-nearest neighbors (k-NN) as the classifier. The study highlights the significant impact of S-shaped and V-shaped transfer functions on binary transformations and classifier performance in high-dimensional medical imaging. Notably, B-MOHGSO algorithms, particularly those using V-shaped transfer functions, excelled in selecting relevant features while maintaining high accuracy. When combined with the VGG19 model and SVM classifier, B-MOHGSO significantly reduced the feature set without sacrificing performance.

The application of B-MOHGSO in COVID-19 classification is crucial for identifying key features that enhance diagnostic processes and treatment strategies. By adapting MOHGSO for discrete optimization, this research aims to address the complexities of high-dimensional medical data and improve healthcare analytics outcomes.

keywords: Data mining, Feature selection; Multi-objective Henry Gas Solubility Optimizer (MOHGSO); COVID-19 classification; Deep learning; Transfer functions.

Résumé

La capacité d'extraire des connaissances précieuses à partir de vastes ensembles de données est cruciale pour favoriser l'innovation et orienter la prise de décision éclairée. Ce processus, souvent appelé extraction de connaissances ou fouille de données, utilise des algorithmes pour identifier efficacement les motifs et les relations dans des bases de données complexes. Cependant, les méthodes traditionnelles peuvent ne pas exploiter pleinement le potentiel des données disponibles ou manquer d'optimisation pour pallier leurs limitations. Par conséquent, une approche plus fondamentale est nécessaire, impliquant le développement de nouveaux algorithmes ou cadres capables d'extraire des informations précieuses à partir de grands bases de données.

Dans cette thèse, nous explorons une telle approche en intégrant l'apprentissage profond avec des techniques avancées de sélection d'attributes pour améliorer la précision de la classification des images radiographiques pulmonaires associées au COVID-19. La base de données comprend des images radiographiques classées en trois catégories : COVID-19, pneumonie, et normal. Nous exploitons des modèles de réseaux de neurones convolutifs de pointe (AlexNet, VGG19, GoogleNet et ResNet) pour extraire des attributes discriminantes. Afin d'optimiser la sélection de ces attributes, nous proposons d'utiliser un algorithme d'optimisation inspiré de la physique, l'algorithme d'optimisation de la solubilité du gaz de Henry multi-objectifs binaire (B-MOHGSO). Cet algorithme, basé sur les principes de la chimie physique, permet de sélectionner de manière efficace les attributes les plus pertinentes pour la classification. Huit versions de l'algorithme HGSO multi-objectifs binaire ont été testées pour optimiser la classification, évaluées en utilisant les k-plus proches voisins (k-NN) comme classificateur. L'impact des fonctions de transfert en forme de S et en forme de V sur la transformation binaire des valeurs continues a également été étudié, car ces fonctions jouent un rôle crucial dans la performance du processus de sélection d'attributes. La méthodologie proposée démontre une performance supérieure en termes de précision, d'efficacité computationnelle, et de sélection d'attributes. La robustesse de nos résultats est en outre assurée par la validation croisée en k-fold, soulignant le potentiel de cette approche pour optimiser le processus d'extraction de connaissances dans des bases de données vastes et complexes.

mots-clés: Fouille de données; Sélection d’attributs; Multi-Objectifs Henry Gaz Solubilité Optimisateur (MOHGSO); Classification de la COVID-19; Deep learning; Fonctions de transfert.

ملخص

تُعد القدرة على استخراج المعرفة القيمة من مجموعات البيانات الضخمة أمرًا بالغ الأهمية لتعزيز الابتكار وتوجيه عملية اتخاذ القرارات المستنيرة. تُعرف هذه العملية غالبًا باستخراج المعرفة أو التنقيب عن البيانات، حيث تستخدم الخوارزميات لتحديد الأنماط والعلاقات في مجموعات البيانات المعقدة بكفاءة وفعالية. ومع ذلك، قد لا تستفيد الطرق التقليدية بشكل كامل من الإمكانيات المتاحة في البيانات أو قد تفتقر إلى التحسين اللازم لمعالجة قيودها. لذلك، هناك حاجة إلى نهج أساسي أكثر، يتضمن تطوير خوارزميات أو أطر عمل جديدة قادرة على استخراج رؤى قيمة من مجموعات البيانات الكبيرة.

في هذه الأطروحة، نستكشف مثل هذا النهج من خلال دمج التعلم العميق مع تقنيات اختيار الميزات المتقدمة لتعزيز دقة تصنيف حالات كوفيد19 باستخدام صور الأشعة السينية للصدر. تتألف مجموعة البيانات هذه من صور أشعة سينية مصنفة إلى ثلاث فئات: كوفيد19، الالتهاب الرئوي، والرئة السليمة. نستكشف إمكانيات تقنيات التحسين المستوحاة من الفيزياء في التعلم الآلي من خلال استخدام خوارزمية تحسين ثنائية متعددة الأهداف (B-MOHGSO) لاختيار الميزات، بالاعتماد على نماذج AlexNet ، VGG19 ، GoogleNet و ResNet لاستخراج الميزات. تم اختبار ثمانية إصدارات من هذه الخوارزمية HGSO متعددة الأهداف الثنائية لتحسين التصنيف، وتم تقييمها باستخدام المصنف (k-NN) . كما تم التحقيق في تأثير وظائف التحويل على شكل S وعلى شكل V على التحويل الثنائي للقيم المستمرة، حيث تلعب هذه الوظائف دورًا حاسمًا في أداء عملية اختيار الميزات. توضح الطريقة المقترحة أداءً متفوقًا من حيث الدقة والكفاءة الحسابية واختيار الميزات. يتم ضمان قوة نتائجنا أيضًا من خلال التحقق المتقاطع باستخدام k-fold، مما يسلط الضوء على إمكانية هذا النهج في تحسين عملية استخراج المعرفة من مجموعات البيانات الكبيرة والمعقدة.

كلمات مفتاحية: التّنقيب عن البيانات؛ اختيار الميزات؛ التحسين متعدد الأهداف؛ تصنيف كوفيد 19؛ التعلم العميق؛ وظائف النقل.