BULGARIAN ACADEMY OF SCIENCES
INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGIES
Department: "Information Processes and Decision Support Systems"

Edjola Naka

Optimization Algorithms for Data Management

Dissertation Thesis

for acquiring Educational and Scientific Degree
"DOCTOR"

In Professional field 4.6. "Informatics and Computer Science",
in Doctoral Program: Informatics

Scientific Supervisor: Prof. Dr. Vassil Georgiev Guliashki

Sofia, 2024

Table of Contents

Dedication

To my parents for their efforts in inspiring and supporting my education

Acknowledgments

I am truly excited and thankful for all the people that were present during these four years of my PhD studies. This was a lengthy period in my life, a journey filled with moments of joy, gratitude, pride, hope, and inspiration, together with moments of personal and professional challenges. I am grateful that this journey has transformed me emotionally and professionally, and I hope to contribute with humility and inspiration to my academic career and research in the future.

This journey starts and ends with the unconditional help and support of my supervisor, Prof. Dr. Vassil Guliashki, who has believed in me in offering this opportunity, unconditioned support, freedom of research, and his valuable advice on improving this dissertation. Thank you very much for everything.

A special appreciation for their support is for my dearest colleagues, and friends from Alexander Moisiu University of Durres, Alsa, and Eris, for the long talks about this challenge, support in my daily workdays, and their friendship. Moreover, I want to thank Prof. Dr. Lindita Mukli for her understanding, and encouragement, particularly in the final phase of the dissertation.

I will be forever grateful to my family—my beloved parents, sisters, Euglena, and Eriona, and nephew Ansel—for standing by me, supporting and understanding my sacrifices, the absent periods, and the difficulties. I thank God they experienced this journey alongside me and for their presence in my life.

Lastly, I would like to send my deepest appreciation to my lifelong love, Arbër, whose persistent emotional support has made this day possible, and our baby boy, Ario, which his arrival ends this chapter, and starts more beautiful ones.

Dissertation Structure

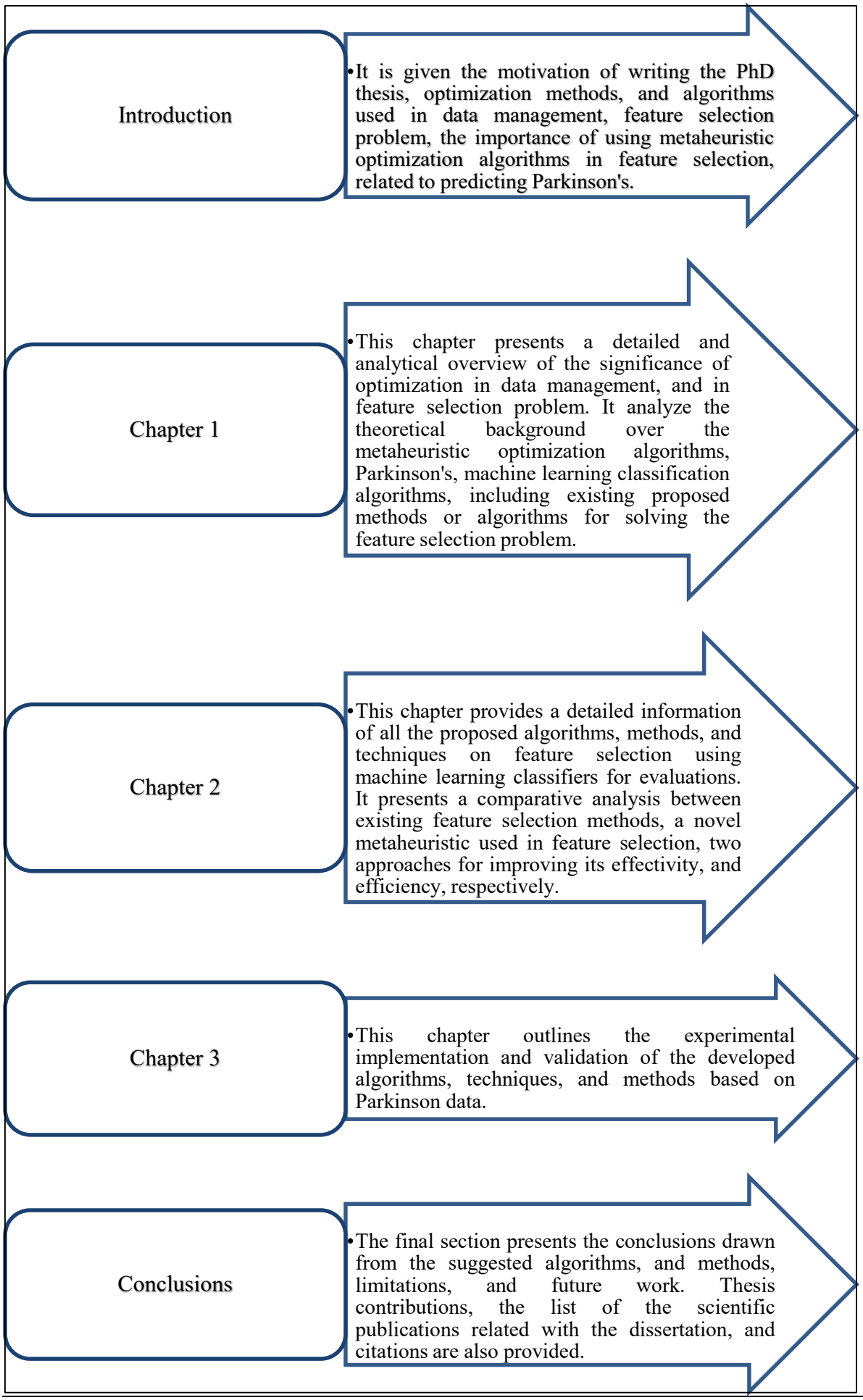
Naka, E. **Optimization Algorithms for Data Management**. Scientific supervisor: Prof. Dr. Vassil Georgiev Guliashki. Department Information Processes and Decision Support Systems. Doctoral Program: “Informatics”. Sofia, 2024. The dissertation with 165 pages, consists of an introduction, 3 chapters, conclusions - a summary of the results obtained, thesis contributions, a list of 7 publications on the dissertation, declaration of originality of the thesis, and a bibliography of 287 references.

Keywords

Data Management, Feature Selection, Metaheuristic, Optimization, Parkinson, Machine Learning, Algorithm

Roadmap of the dissertation

The thesis is structured in the following parts:


Introduction
•It is given the motivation of writing the PhD thesis, optimization methods, and algorithms used in data management, feature selection problem, the importance of using metaheuristic optimization algorithms in feature selection, related to predicting Parkinson's.
Chapter 1
•This chapter presents a detailed and analytical overview of the significance of optimization in data management, and in feature selection problem. It analyze the theoretical background over the metaheuristic optimization algorithms, Parkinson's, machine learning classification algorithms, including existing proposed methods or algorithms for solving the feature selection problem.
Chapter 2
•This chapter provides a detailed information of all the proposed algorithms, methods, and techniques on feature selection using machine learning classifiers for evaluations. It presents a comparative analysis between existing feature selection methods, a novel metaheuristic used in feature selection, two approaches for improving its effectivity, and efficiency, respectively.
Chapter 3
•This chapter outlines the experimental implementation and validation of the developed algorithms, techniques, and methods based on Parkinson data.
Conclusions
•The final section presents the conclusions drawn from the suggested algorithms, and methods, limitations, and future work. Thesis contributions, the list of the scientific publications related with the dissertation, and citations are also provided.

List of Abbreviations

Abbreviation	Description
FS	Feature Selection
ML	Machine Learning
MHOA	Metaheuristic optimization algorithm
VPL	Volleyball Premier League
BVPL	Binary Volleyball Premier League
ALO	Antlion optimizer
BALO	Binary Antlion optimizer
BVPL_BALO	Binary Volleyball Premier League Antlion Optimizer
k-NN	k-Nearest Neighbour
SVM	Support Vector Machine
RF	Random Forest
GA	Genetic Algorithm
GSA	Generalized Simulated Annealing
ACO	Ant Colony Optimization
ABC	Artificial Bee Colony
ASO	Atom Search Optimization
BA	Bat Algorithm
DE	Differential Evolution
DF	Dragon Fly algorithm
FA	Firefly Algorithm
GWO	Grey Wolf Optimization
HHO	Harris Hawk Optimization
MFO	Moth Flame Optimization
PSO	Particle Swarm Optimization
SSA	Salp Swarm Algorithm
TGA	Tree Growth Algorithm
WOA	Whale Optimization Algorithm
EOA	Equilibrium Optimizer Algorithm
SCA	Sine Cosine Algorithm
TLBO	Teaching Learning-based Optimization
GOA	Grasshopper Optimization Algorithm
TF	Transfer Function
IG	Information Gain
GR	Gain ratio
JMI	Joint Mutual Information
mRmR	Minimum redundancy maximum relevance
SBS	Sequential Backward Search
SFS	Sequential Forward Search
RS	Random Search

Introduction

At our current pace, the data industry generates around 2.5 quintillion bytes of data per day, primarily from sources such as official state statistics, the internet, social media, digital photos, communication mediums, the IoT, and other business and service providers [1], motivating the work in this thesis for searching and exploring efficient and effective optimization algorithms and methods for reducing and selecting the most prominent features from the generated data. According to Statista Digital Economy Compass 2019, the global trend in data creation will rise from 175 zettabytes in 2025 to 2,142 zettabytes in 2035 [2], according to their predictions, which confirm the necessity for extracting, and selecting the most important information.

Correct understanding, analysis, and reporting of these data are essential to assist decision-makers and industries in making optimal choices. Public and non-public organizations research, investigate, and heavily finance the health field, one of the largest industries, with the aim of interpreting and predicting diseases. The United States spends almost 18 percent of its GDP on health care, which is more than any other country, followed by Germany and Switzerland in 2022 [3]. Forecasters predict a continuous increase in global per capita consumer healthcare spending of 21.55% U.S. dollars between 2024 and 2029 [4]. The amount of data generated in the field of medicine is staggering, and it continues to grow rapidly. Clinical data, genomic data, imaging data, sensor data, and healthcare IoT data are some of its sources. However, managing and examining this data presents significant challenges in terms of storage, processing, privacy, and data integration.

Increased life expectancy is arguably one of the greatest achievements of health systems around the world. However, it has also led to increases in age-related neurological disorders, such as Alzheimer’s and other dementias, stroke, and Parkinson’s, necessitating global health policies not only to focus on survival but also to minimize health loss due to disability by promoting function and independence. In 2021, 3.40 billion individuals had nervous system health loss, and 11.1 million individuals died from a nervous system condition [5]. For adults aged 80 years and older, the leading causes were stroke, Alzheimer’s and other dementias, and Parkinson’s. Experts estimate that the number of Parkinson's patients will rise from 4.1 million in 2005 to 8.7 million in 2030 [6].

Optimization is a group of methods, techniques, and algorithms used to find the best solution for a problem with one or several criteria and a set of constraints. Various fields, including mathematics, engineering, economics, and computer science, widely use these methods. There are different techniques for preprocessing the variety of generated data, among them dimensionality reduction and feature selection (FS). The ultimate one is an optimization problem that involves defining binary decision variables to indicate whether the feature is selected, an objective function that evaluates the performance of a model built using the selected features, and constraints that may limit the number of features selected. The feature selection optimization problem can be formulated as follows:

$$\max(\text{ or } \min)\, M\left(L(X_S, Y)\right)$$
$$subject\ to\ \sum_{j=1}^{d} s_j \leq k,$$
$$s_j \in \{0,1\}, j = 1, 2, \dots, d$$

where s_j indicates that the feature $s_j = 1$ is selected or not $s_j = 0$; M shows an evaluation metric that measure the quality of a generated subset of feature, L shows the supervised learning algorithm that takes as an input the subset of features; dataset $D = \{(x_i, y_i)\}, i = 1, \dots, n$ where x_i is a feature vector with n features, and y_i is the target variable, d shows the index of the of the j_{th} vector from the subset of the selected features. The constraint $s_j \leq k$ can limit the number of selected features.

Over the years, FS has gained more attention and interest from the research community. A Google Scholar search for "feature selection" yielded between one million and 50,000 relevant papers up until February 2022 [7], demonstrating the significant interest in this field. FS is a critical step in the machine learning pipeline to improve model performance, reduce complexity, and enhance interpretability. However, it requires careful consideration of the data, the problem domain, and the chosen feature selection method. Searching for the minimum feature subset selection is a NP-hard problem [8], for which provably efficient algorithms do not exist.

Metaheuristic optimization algorithms (MHOAs), a type of optimization algorithm, have been widely used in recent decades to reduce the number of features and select the most relevant, important, and significant features from diverse datasets. Some key characteristics that make them frequently used are that they are not related to a specific

problem, include a stochastic search, offer an iterative improvement of the candidate solution, and involve exploration and exploitation to efficiently navigate the search space in order to reach global optimum. Many metaheuristics have been proposed over the years. Particle Swarm Optimization (PSO), traditional Genetic Algorithm (GA), and differential evolution (DE) were the three most essential metaheuristic optimization algorithms among the 300 metaheuristic algorithms observed [9]. According to a literature review consisting of 1222 publications from 1983 to 2016 [10], PSO has gained immense popularity among researchers due to its simplicity and effectiveness in numerous scientific and industrial applications. Next in line are artificial bee colony (ABC), Ant Colony Optimization (ACO), and GA. However, classic metaheuristics like Tabu Search (TS), DE, simulated annealing (SA), variable neighborhood search (VNS), and iterated local search (ILS) keep getting a lot of attention because they work well in a wide range of optimization problems. Among modern methods, harmony search (HS), cuckoo search (CS), bat algorithm (BA), firefly algorithm (FA), and fireworks algorithm (FWA) have shown the efficiency of producing quality solutions. Another metaheuristics study [11] found that classical algorithms like SA, DE, PSO, GA, and ACO significantly influence the field. The most cited metaheuristics based on Google Scholar data up until December 31, 2022 [12] are PSO, which ranks first with 75,000 citations, followed by GA with over 70000 citations. Next, the most cited metaheuristics are ACO, DE, SA, TS, Grey Wolf Optimizer (GWO), ABC, CS, and finally HS, in that order.

In addition to utilizing single metaheuristics, combinations of them or local improvements on them are largely used. A sample of 111 recent studies were analyzed that used new, hybrid, or improved optimization algorithms [13]. They observed that in percentages, algorithms based on new metaphors are around 19%, combinations of existing algorithms are 16%, and improved versions are more frequent (65%). From the other side, SA and sine cosine algorithm (SCA) were most commonly used in conjunction with other metaheuristic algorithms. The most frequent fields of application were feature selection, electrical engineering, and structural engineering, with 28%, 22%, and 13%, respectively. On May 28, 2024, a search using the keywords "metaheuristic" and "feature selection" on Google Scholar yielded approximately 138,000 results, demonstrating the significant interest in incorporating metaheuristics into feature selection.

Over the years, a variety of MHOAs have been used as a search method to select the most relevant and representative features in feature selection. It is important to note that no universal metaheuristic approach can effectively solve all types of optimization problems across all application domains. The no-free lunch (NFL) theorem establishes that for any algorithm, any elevated performance over one class of problems is offset by performance over another class [14].

Given the widespread use of the previously mentioned metaheuristics in FS, this dissertation introduces the binary volleyball premier league (VPL) algorithm first used on FS, where the continuous metaheuristic debuted in 2018. Given that the usual metaheuristics are proposed for solving continuous problems, and since FS requires that they provide only binary solutions (0 or 1) in order to identify which feature is not important or it is important, it was necessary to improve, and adapt the non-binary VPL in a Binary Volleyball Premier League (BVPL) dedicated for the FS problem. This metaheuristic algorithm simulates the original volleyball game conditions and the volleyball teams' competition in a league. Each team will compete with the others, and in the end of the season, the winning team will represent the best subset of features which are more representatives on predicting a phenomenon. Machine Learning (ML) classifiers are usually used for evaluating the quality of the solutions. The primary objective is to investigate, analyse, and improve the VPL to better solve the FS problem focusing on predicting Parkinson's, and its efficiency and efficacy are investigated and improved to enhance the final achieved optimum and to produce it in a quicker time. Two reasons for applying the BVPL metaheuristic and all the proposed improvements on Parkinson's data are:

- It is one of the most important neurological diseases, and a lot of data are generated by non-profit organizations, health services, research group projects, and governments, from whom need to be extracted the most important information.
- Since MHOAs have demonstrated remarkable efficacy in the FS problem, BVPL adaptability in Parkinson's disease prediction is being studied as it hasn't been utilized in FS before.

To achieve these objectives, this dissertation comprises seven interconnected studies that utilize binary volleyball premier league algorithm, and improvements, feature

selection methods, and machine learning classification algorithms to predict Parkinson's ([15], [16], [17], [18], [19], [20], and [21]). This research intends to show how metaheuristic optimization algorithms and machine learning can be used to get rid of features from the Parkinson's datasets that aren't useful to improve prediction accuracy and reduce the average size of the number of features that are chosen. The dissertation's goal is to create and improve new metaheuristic optimization algorithms for the feature selection problem in Parkinson's prediction using machine learning (ML) classifiers, with a focus on improving the efficiency, effectiveness and execution time of the Parkinson’s prediction algorithms.

1. Data Management, Machine Learning, and Metaheuristic algorithms: A state-of-the-art overview

This chapter describes general concepts and definitions about the use of optimization methods and algorithms on different data management key fields, with a greater focus on the feature selection problem. It summarizes concepts on metaheuristic optimization algorithms, their application in feature selection, the function of machine learning classifiers in feature selection, and provides an overview of Parkinson's disease. Additionally, it showcases practical advancements in enhancing metaheuristics, developing hybrid metaheuristics, and combining them with other feature selection techniques to forecast Parkinson's disease.

1.1 Data Management and Optimization

1.1.1 Data Management

“Data Management is the development, execution, and supervision of plans, policies, programs, and practices that deliver, control, protect, and enhance the value of data and information assets throughout their lifecycles” [22]. Data management activities are wide-ranging. They include everything from the ability to make consistent decisions about how to get strategic value from data to the technical deployment and performance of databases. Thus, data management requires both technical and business skills. Data and information are also vital to the day-to-day operations of most organizations. Nowadays, a lot of digital data are generated from organizations in every time. This data can be classified according to states as data at rest, data in motion, and data at use. According to the kind of data that organizations create, different data structures and infrastructures are used. Data management systems are built on data management platforms that, in addition, integrate databases, data warehouses, and big data management systems. Data can be stored in relational or NoSQL databases, meanwhile big data processing involves a set of techniques or programming models to access large-scale data to extract useful information for supporting and providing decisions.

Databases are a kind of structured data which can help business to take quick decisions mostly related with reporting. Database systems are used to manage collections of data that are: highly valuable, relatively large, and accessed by multiple users and applications. The primary goal of using database systems is to store and access data quickly and efficiently. In the case where organizations have also data at motion and data in use, different data structures are recommended to be created and used. NoSQL databases were an essential requirement for improving the scalability and performance of the database. NoSQL databases were created for managing unstructured data as XML data, JSON, text documents, etc. The majority are referred to NoSQL as databases that do not store data according to the relational model and are not using SQL language.

The concept of the Data Warehouse emerged in the 1980s as a necessity to integrate data from different source databases with the goal of having all the necessary data centralized for analysis. The evolution of DWH in time has transformed it into a large repository of historical integrated data for supporting better decisions. DWHs are usually modeled with relational schema: star and snowflake. Due to redundant data in DWHs, and the lower number of necessary joins in queries, they have better performance than relational databases. Extract, Transform and Load (ETL) process is used to extract data from different sources, transform in appropriate formats and load them in data warehouses. On Line Analytical Processing (OLAP) is a group of technologies used for doing multidimensional analysis for business data. OLAP extract the data from DWH and organize them in data cubes which are pre-computed multidimensional views of data, and can be optimized for quick analyses.

The exponential growth of data had developed a new approach called Big Data. Big Data is mostly characterized by 5V [23] which are: Volume, Variety, Velocity, Veracity and, Value, but more V-s are added in the future with the aim to explain in more details the big data. Big data infrastructure should be able to manage extreme parallel processing, high-speed replications, high availability, distributed file-based storage, linearly scalable infrastructure, localized processing of data, and storage of results [24].

Building and storing data in these infrastructures is interrelated with technologies, and fields for analyzing them in order to benefit from them. Machine Learning (ML) is one approach used for these analyses. Arthur Samuel, a pioneer in Machine Learning

described it as a: "Field of study that gives computers the capability to learn without being explicitly programmed" [25]. It is a field of artificial intelligence that includes methods and algorithms that learns from past events and builds new models able to self-learn from data. ML methods and techniques could analyze data from data warehouses and big data technologies. ML algorithms create models that are used to design predictable ones. Big Data from several sources must be processed correctly and ML is widely used in their analysis.

1.1.2 Optimization process

A key step in many decision-making and design processes is the optimization phase, which itself contains several stages. The purpose of the optimization process is to help determine realistic and practical outcomes of management decision-making and design processes. Optimization techniques are applied to real-life systems that have perceived complex management problems. Most analysts break a decision-making process down into six major steps or phases [26]:

1) Identifying and clarifying the problem
2) Defining the problem
3) Formulating and constructing a mathematical model
4) Obtaining a solution to the model
5) Testing the model, evaluating the solution, and carrying out sensitivity analysis
6) Implementing and maintaining the solution

In the above decision-making process, steps 2–5 represent the optimization process. Problems that seek to maximize or minimize a mathematical function of a number of variables, subject to certain constraints, form a unique class of problems called optimization problems. The mathematical function that needs to be optimized is known as the objective function, containing usually several variables. Optimization problems can be written in a generic form as in Eq. (1.1):

$$\begin{array}{lll} \underset{x \in R^d}{min} & f_i(x), & (i = 1,2,\dots,M), \\ subject\ to & h_j(x) = 0, & (j = 1,2,\dots,J), \\ & g_k(x) \le 0 & (k = 1,2,\dots,K), \end{array} \tag{1.1}$$

where $f_i(x)$, $h_j(x)$, $g_k(x)$ are functions of the design vector $x = (x_1, x_2, \dots, x_d)^T$. Here the components x_i are called decision variables, and they can be real continuous,

discrete, or a mix. The functions $f_i(x)$ where $i = 1,2,\dots,M$ are called the objective functions or simply cost functions, and in the case of M=1, there is only a single objective. The space spanned by the decision variables is called the search space R^d, whereas the space formed by the objective function values is called the solution space or response space. The equalities for h_j and inequalities for g_k are called constraints. It is worth pointing out that the inequalities can be written in the other way, ≥ 0, and the objective function can be also formulated as a maximization problem.

Classification of the optimization methods types is not fully derived, but there are some characteristics that can be included for this categorization as presented on Figure 1.1 [27].

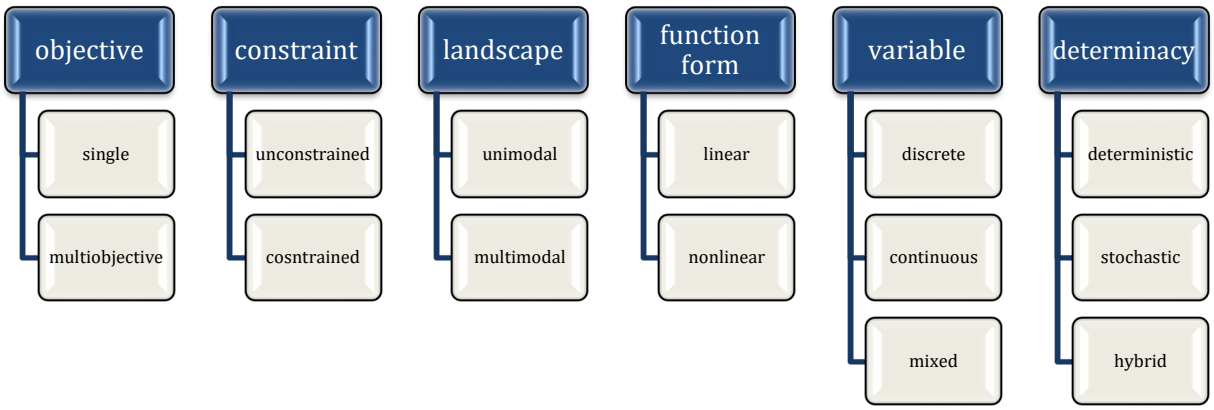


Figure 1.1. Categorization of optimization algorithms

Usually, the last category is largely studied in the last decades, and a general categorization is deterministic, stochastic, and hybrid. Deterministic algorithms follow a rigorous procedure, and its path, values of design variables and the functions are repeatable; it means that for the same starting point, they will follow the same path whether you run the program today or tomorrow. All the mentioned categories above, are usually deterministic. On the other hand, the stochastic algorithms always have some randomness. The final results may be no big difference, but the paths of each individual are not exactly repeatable. Also, there are possible hybrid of deterministic and stochastic algorithms that can be offered. Regarding stochastic algorithms, in general they are divided in two types: heuristic and metaheuristic, though their

difference is small. H*euristic* means 'to find' or 'to discover by trial and error'. Quality solutions to a tough optimization problem can be found in a reasonable amount of time, but there is no guarantee that optimal solutions are reached. This is usually good enough when it is not necessarily wanted the best solutions but rather good solutions which are easily reachable. Further development over the heuristic algorithms are the so-called metaheuristic algorithms. Here *meta-* means 'beyond' or 'higher level', and they generally, perform better than simple heuristics. In addition, all metaheuristic algorithms use certain tradeoff of randomization and local search.

1.1.3 Optimization and data management

Optimization is considered very helpful in optimizing different structures of data management. For example, query optimization in a relational database management system is a process which helps for selecting the optimal queries execution plan [28]. Data management uses query optimization to maximize the speed and precision of data retrieval. The conventional optimization technique includes selection ordering, optimizer choices in a multi-way join query (as access methods, join order, join algorithm, and pipeline), finding a single robust query plan, or finding a small set of plans that are appropriate for different situations. Metaheuristics have been largely used in relational databases. An efficient metaheuristic algorithm called adaptive Cuckoo search for querying and generating an optimal query plan for large resource description framework is proposed in [29]. The proposed approach has provided significant results in terms of query execution time. The NP- complete multi join query ordering (MJQO) problem is solved in [30] by means of a heuristic algorithm combining the basic search algorithms Cuckoo and Tabu Search. In [31] it is proposed a multi-colony ant algorithm for distributed join query optimization based on Max-Min Ant System. Bees Algorithm is another one which has resulted more effective than Ant Colony Optimization algorithm in the problem of multi join query optimization [32].

Optimization in DWHs is the process of selecting adequate techniques in order to make queries run faster by maximizing the exploitation of the systems resources. DWHs store historical data, therefore, they use large and complex queries. Data warehouses can have distributed architectures as e result of techniques as horizontal partitioning and parallel processing that can be applied in them, hence query processing and optimizing

affect the performance. Join optimization problem is considered also in DWH. High response time of analytical queries is one of the most challenging issues of data warehouses. In DWH, materialized view selection can reduce the response time of the analytical queries. For this purpose, the search space is firstly constructed by producing the set of all possible views based on given queries and then, the (semi-) optimal set of materialized views will be selected so that the queries can be answered at the lowest cost using them. For example, a novel coral reefs optimization-based method is introduced for materialized view selection in a data warehouse [33]. The optimization problem in the data cube system design is to optimize an initial set of cubes such that the system can answer a large number of queries and satisfy the bounds. In [34] was proposed an OLAP data cube selection discrete particle swarm algorithm that achieves solid results. In the case of big data, an application of metaheuristic is GWO algorithm which is proposed to make an appropriate scaling decision to provide the cloud resources for serving of cloud workloads [35]. Regarding the NoSQL databases, for example, a metaheuristic algorithm comprising of Best Fit Decreasing with ACO is proposed for data allocation in a distributed architecture of graph NoSQL databases [36]. In another case, PSO, and FA algorithms are used in positioning and optimization of traffic in NoSQL databases [37], [38].

ML and optimization benefit from each other and contributes respectively [39]. Almost all ML algorithms can be formulated as an optimization problem to find the extremum of an objective function. The main steps of ML are to build a model hypothesis, to define the objective function, and solve optimization problem to find out the maximum or minimum of the objective function to determine the parameters of the model [40]. According to the modelling purpose and the problem to be solved, the ML algorithms can be divided into supervised learning, semi-supervised learning (SSL), unsupervised learning, and RL. Regarding the supervised learning, the goal is to find an optimal mapping function f(x) to minimize the loss function of the training samples, shown in Eq. (1.2):

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y^i, f(x^i, \theta)) \tag{1.2}$$

where N is the number of training samples, θ is the parameter of the mapping function, x^i is the feature vector of the ith sample, y^i is the corresponding label, and L is the loss function such as: square of Euclidean distance, information gain, cross-entropy etc.

The existing literature analyzing the hybridization of metaheuristics and ML usually discuss the two approaches: ML is employed to enhance metaheuristics, and the other in which metaheuristics are used to improve the performance of ML techniques. ML techniques can be used in improving metaheuristics in different processes [41], [42], [43]: designing search components or parameters of metaheuristics, for example in initial solution(s), search operator design, parameter tuning, fitness evaluation, and improve the performance of cooperative metaheuristics by adjusting their behavior during the search process. Also, it can be used in population management, operators, local search, reduction of search space, algorithm selection, hyper-heuristics, and new types of metaheuristics.

Optimization algorithms should have some added properties from a ML perspective as scalability to large problems, good performance in terms of execution times and memory requirements, fast convergence to an approximate solution, exploitation of problem structure, robustness and numerical stability for class of machine learning models attempted, simple and easy implementation of algorithm [44] etc.

There are also other approaches for hybridizing metaheuristics, as matheuristics, simheuristics, and learnheuristics. A matheuristic is the hybridization of an exact method with metaheuristics, the hybridization of simulation methods and metaheuristics is called simheuristics, and learnheuristics combines the metaheuristics with machine learning.

1.2 Feature Dimensionality

1.2.1 Dimensionality reduction

Dimensionality reduction is the process of transforming high-dimensional data into a low-dimensional space so that the low-dimensional representation retains meaningful features from the original data. The process of mapping high-dimensional data to low-dimensional space through projections will inevitably lead to the loss of some original information. The problem that needs to be resolved at present is to obtain useful

reduction data from the high-dimensional data set to meet the recognition accuracy and storage requirements under the premise of maintaining the essential characteristics of the original data optimally. The advantages dimensionality reduction offers includes: easy model building, simplify model debugging, efficient selection of relevant features, reduce massively model training time, improve the model performance, easy model interpretation, and visualizations [45]. Usually, there are two main approaches to dimensionality reduction:

- Feature Selection: This approach selects a subset of the original features while discarding the rest. The selected features are considered the most informative for the given task. Next paragraph details more information about FS.
- Feature Extraction: This approach creates new features that are combinations of the original features. These new features, called components or embedding's, are constructed in such a way that they capture as much of the variance in the data as possible. Feature extraction algorithms are divided into two categories: linear and nonlinear, and various categorization are mentioned here [46]. In linear algorithms, some methods are for example: Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Factor analysis (FA). Regarding non-linear methods some of them are: T-Distributed Stochastic Neighbour Embedding (t-SNE), kernel function, manifold learning, isometric map, deep auto encoder etc.

1.2.2 Feature Selection

Feature selection is the process of selecting all relevant features and discarding the redundant and irrelevant ones, to maximize the classification rate of the classifier and diminish the complexity of the original dataset when faced with all the features of the dataset.

Let it be the feature set, $F = f_1, f_2, \dots, f_n$ where n is the dimension of the dataset, and the set $C = c_1, c_2, \dots, c_l$ of class labels, where l is the number of different values of a class label in a dataset. The goal of FS is to find a subset $S = s_1, s_2, \dots, s_d$ where d < n that provides better prediction accuracy than the full features dataset. FS seeks to choose a small subset of the relevant features from the original ones. FS can potentially

yield benefits such as reduced data volume, improved precision, streamlined outcomes, and decreased feature complexity [47]. There are three main methods to address the issue of FS: filter, wrapper, and embedded methods [48]. The selection of features in filter methods is independent of the choice of a machine learning classifier, whereas wrapper methods rely on the performance of the classifier algorithm when evaluating different subsets of features. The embedded feature selection methods are implemented using algorithms with their own built-in feature selection methods. Often is included another categorization: hybrid methods which utilizes more than one strategy for selecting a feature to create subsets. For example, one method from filter and other from wrapper creates a hybrid method. Given a specific learning algorithm, a typical wrapper method performs two steps: (1) search for a subset of features; and (2) evaluate the selected features. It repeats (1) and (2) until some stopping criteria are satisfied. Feature set search component first generates a subset of features; then the learning algorithm acts as a black box to evaluate the quality of these features based on the learning performance. Filter methods are typically more computationally efficient than wrapper methods. However, due to the lack of a specific learning algorithm guiding the feature selection phase, the selected features may not be optimal for the target learning algorithms. Lastly, embedded methods incorporate feature selection as an integral part of the classifier algorithm. Thus, they inherit the merits of wrapper and filter methods – (1) they include the interactions with the learning algorithm; and (2) they are far more efficient than the wrapper methods since they do not need to evaluate feature sets iteratively. Details of them have been discussed in previous review papers [49], [50] [51], [52]. Currently, no single feature selection method stands above the rest. Each method has its strengths and weaknesses, as described for example in [49] .

The wrapper-based approach in FS, includes using different supervised learning algorithms of ML as an approach for testing the fitness of the solutions generated by metaheuristics. These algorithms are either search-based or correlation-based. For the search-based framework, a typical feature selection process consists of three basic steps namely subset generation, subset evaluation, and stopping criteria.

Figure 1.2 indicates that search-based feature selection includes two key factors: the evaluation criterion and the search strategy. Search strategies are usually categorized into complete, sequential, and random models.

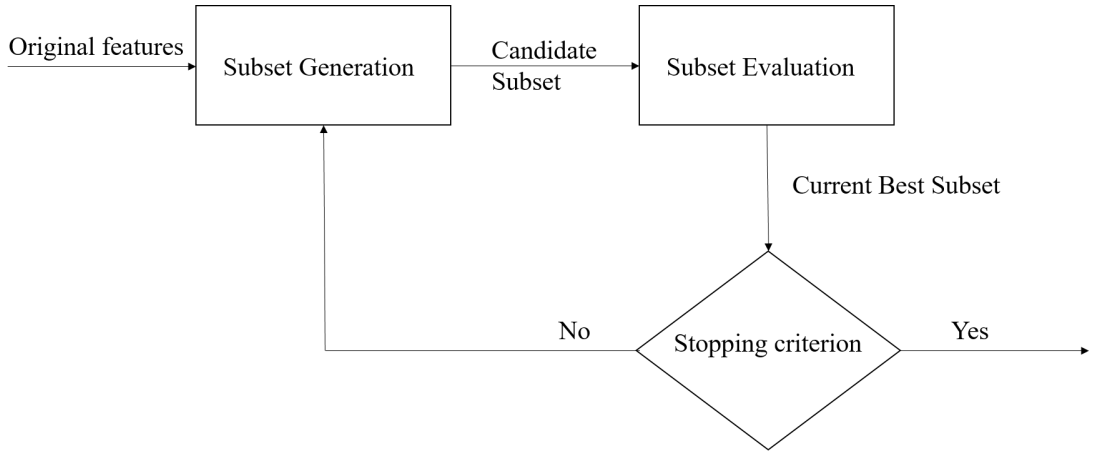


Figure 1.2. The structure of search-based feature selection [53]

The correlation-based framework considers the feature–feature correlation and feature–class correlation. Generally, the correlation between features is known as feature redundancy, while the feature–class correlation is viewed as feature relevance. Then an entire feature set can be divided into four basic disjoint subsets: (1) irrelevant features, (2) redundant features, (3) weakly relevant but non-redundant features, and (4) strongly relevant features. The correlation-based feature selection framework is shown in Figure 1.3, which consists of two steps: relevance analysis, which determines the subset of relevant features, and redundancy analysis, which determines and eliminates the redundant features from the relevant ones to produce the final subset.

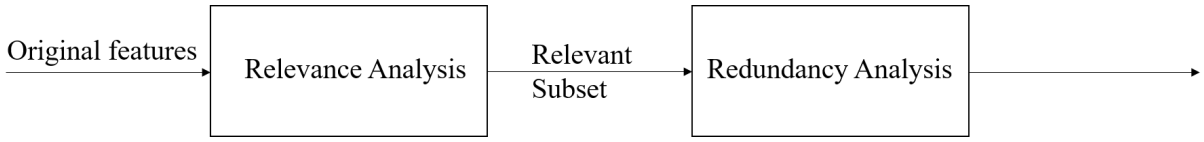


Figure 1.3. The structure of correlation-based feature selection [53]

This framework has advantages over the search-based framework as it searches and allows for an efficient and effective way of finding an approximate optimal subset [53]. An optimal feature selection algorithm should select non-redundant and strongly relevant features.

1.2.3 Feature importance.

One approach to dimensionality reduction, feature importance methods, ranks the features of a dataset, meanwhile feature selection reduce the size of the features of a dataset. Feature importance is useful for machine learning tasks because it allows practitioners to understand which features in a dataset contribute most to the final prediction and which are less important. Incorporating dimensionality reduction techniques into machine learning algorithms can significantly increase their performance [54]. Therefore, higher feature importance reliability might yield lower feature importance errors and higher prediction accuracy [55]. The information generated by feature importance methods can be used in a variety of ways, such as in feature selection, model interpretability, business decision-making, and improving model performance. The feature importance scores can be feed to a wrapper model, like a metaheuristic optimization algorithm, to perform feature selection. Since the wrapper methods build and evaluate the classifier, they can be computationally expensive for each feature subset considered, and they have the potential to overfit the predictors to the training data, necessitating external validation [56].

There are some popular proposed methods of feature importance like permutation [57] which measures the change in the model's performance when the values of a particular feature are randomly shuffled, or recursive feature elimination (RFE) [58] which begins with all features and recursively eliminates the least important ones until it reaches the desired number of features. Other models that measure feature importance, like linear models or tree-based models, often combine with RFE. Relief is another method which evaluates the importance of features by considering the weights of their nearest neighbors [59]. Gradient-boosting algorithms like XGBoost, LightGBM, and CatBoost also provide feature importance scores based on the contribution of each feature to the model's performance [60].

Cosine similarity is a method used for discovering similarities between two objects based on how far apart they can be. As the cosine similarity between any two patterns increases, they are considered more similar. Generally, if the cosine similarity value between two items is 1, then the items are considered to be highly similar; when it is 0, they are not similar. Various studies have used the cosine method for selecting and reducing the number of features in high-dimensional datasets. For example, Euclidean

distance and cosine similarity have been used in feature selection for ranking through a score between the features and the class vector for intrusion detection in networks [61]. Another proposed method is based on cosine similarity and mutual information to find out a relevant feature subset [62]. Besides the usage of the cosine method in text documents mostly, it is integrated into the calculation of the exploration stage of the binary Golden Jackal Optimization for updating the position of golden jackal pairs to prevent premature convergence [63]. In another work, an access network selection algorithm based on cosine similarity distance and a PSO algorithm was proposed [64]. The key property of the proposed approach was to minimize the cosine similarity distance between every candidate network and the ideal network. Choosing a better neighbor is very important for improving the exploitation of artificial bee colonies (ABC), therefore the cosine similarity between two individuals is employed to choose a better neighbor [65].

1.3 Parkinson's Disease

Parkinson’s is a degenerative condition of the brain associated with motor symptoms as slow movement, tremor, rigidity, and imbalance, and other complications, including cognitive impairment, mental health disorders, sleep disorders, pain, and sensory disturbances. Over the next 30 years, it is expected an increase in the frequency of Parkinson's, positioning it as the second most prevalent neurodegenerative condition after Alzheimer's disease, with over 10 million individuals worldwide suffering from Parkinson’s [66], [67]. The control and management of Parkinson’s should be strengthened, especially when considering the aging tendency of the population [68].

Diagnosing Parkinson’s continues to be challenging, and research into the condition's early stages is ongoing. Three stages comprise Parkinson's: the preclinical stage, characterized by no obvious symptoms; the prodromal stage, characterized by symptoms that increase the likelihood of future diagnosis; and the manifestation stage, characterized by noticeable symptoms [69]. It is really challenging to predict how Parkinson's may develop because symptoms and treatment outcomes differ from person to person. Despite the lack of a known cure, medications, surgery, and other therapies can manage Parkinson’s symptoms. The most popular and efficient drug is still levodopa or carbidopa. Other drugs, such as anticholinergics, or therapies, like deep

brain stimulation [70], can also alleviate Parkinson's symptoms, particularly tremors, and help patients use fewer medications.

Recent developments over the last few years include the validation of clinical diagnostic criteria, the introduction and testing of research criteria for prodromal Parkinson's, the identification of genetic subtypes, and an increasing number of genetic variants linked to Parkinson's risk. The development of diagnostic biomarkers has made significant strides while genetic and imaging testing are now routine in clinical practice. Parkinson is changing from a clinical to a biomarker-supported diagnostic entity. This means that the disease can be found earlier, different subgroups with different prognoses can be recognized, and new therapies that change the course of the disease can be created. Additionally, a variety of scans can be used to investigate the structure and operation of the brain and other components of the nervous system as: CT (Computerised Tomography) scan, MRI (Magnetic Resonance Imaging) scan, DaTSCAN™-SPECT scan, PET (Positron Emission Tomography) scan [66].

Early signs of Parkinson include bradykinesia, rigidity, tremor, and gait changes. Later signs include changes in posture, gait freezing, and balance. Non-motor symptoms that may appear early on include hyposmia, sleep disorders, anxiety, depression, autonomic dysfunction, mild cognitive impairment, and dementia [69].

Experts try to evaluate and predict Parkinson's using different types of data. Typically, public datasets account for 51.7% of Parkinson's data from a list of 209 studies, meanwhile original data were collected from human participants in 43.1% of papers [71]. Parkinson's datasets have a usage rate of at least 12.20%, out of a total of 82 studies [7].

Academics, and not only are searching for the most effective model for predicting Parkinson's using machine learning and optimization algorithms due to the relevance of this disease. Machine learning has proven to be highly advantageous in this technique because of the large volume of data it's capable of analyzing and processing. Not many researchers have used metaheuristic approaches to look into the FS problem for neurological disorders (Parkinson's, autism, and schizophrenia) and psychological disorders (stress, schizophrenia, and insomnia) [72].

All the tests of the thesis were evaluated on ten Parkinson's datasets. Nine of these datasets are publicly available, named shortly D1 ([73], [74]), (D2_S, D2_M, D3_S,

D3_M ([75], [76], [77]), D4 ([78], [79]), D5 ([80], [81]), D6 ([82], [83]), D7 ([84], [85]) while one was obtained from the Parkinson's Progression Markers Initiative - D8 [86][1], through a private request made by the author. This dataset named Gait_Data_Arm_swing[2], was proposed in order to obtain quantitative, objective motor measures that could inform on pre-clinical symptoms, progression markers, dynamic changes of function throughout disease, and potential modifiers and mediators of motor symptoms. The gait system used consists of three lightweight wireless wearable sensors with three axial accelerometers, gyroscopes, and magnetometers. During all gait measurements, the participants wear the sensors on both wrists and the lower back to quantify temporal measures. The dataset extracts measures related to performance, including a standard walk and dual-task walk (36 features), sway (8 features), and TUG (12 features). The original dataset contained a significant number of missing values. To manage these missing values, were removed all patients for which there were missing values in each column. Among the columns, only SP_U and SP_DT have missing values; therefore, they were removed from the analysis. In this scenario, we will distinguish between patients with Parkinson's and those in the prodromal stage. The last phase, prodromal symptoms include non-motor symptoms that have a substantial influence on the individual's everyday functioning, and where the patient is not fully evaluated with Parkinson's. Chapter 3 provides general information about the dataset dimensions during the experiment evaluations.

1.4 Metaheuristic Optimization Algorithms

1.4.1 Concepts about metaheuristics, taxonomy, and applications

Metaheuristics is a high-level problem-independent algorithmic framework that provides a set of strategies to develop heuristic optimization algorithms [87]. Sorensen and collaborators have described in detail the history of heuristics and metaheuristics

[1] Funding for the D8 dataset: PPMI – a public-private partnership – is funded by the Michael J. Fox Foundation for Parkinson's Research and funding partners, including 4D Pharma, Abbvie, AcureX, Allergan, Amathus Therapeutics, Aligning Science Across Parkinson's, AskBio, Avid Radiopharmaceuticals, BIAL, Biogen, Biohaven, BioLegend, BlueRock Therapeutics, Bristol-Myers Squibb, Calico Labs, Celgene, Cerevel Therapeutics, Coave Therapeutics, DaCapo Brainscience, Denali, Edmond J. Safra Foundation, Eli Lilly, Gain Therapeutics, GE HealthCare, Genentech, GSK, Golub Capital, Handl Therapeutics, Insitro, Janssen Neuroscience, Lundbeck, Merck, Meso Scale Discovery, Mission Therapeutics, Neurocrine Biosciences, Pfizer, Piramal, Prevail Therapeutics, Roche, Sanofi, Servier, Sun Pharma Advanced Research Company, Takeda, Teva, UCB, Vanqua Bio, Verily, Voyager Therapeutics, the Weston Family Foundation and Yumanity Therapeutics.

[2] Dataset D8 used in preparation of this article was obtained on August 01, 2022 from the PPMI database (www.ppmi-info.org/access-data- specimens/download-data), RRID: SCR 006431. For up-to-date information on the study, visit www.ppmi-info.org.

[88], [87], [89]. Descartes first mentioned heuristics in 1637, when he studied the methods and rules of discovery and invention. This led to the development of heuristic reasoning, which refers to thinking strategies that enable us to make judgments or even find solutions, not rigorously but swiftly by identifying the most likely provisional solution. Fred Glover first mentioned metaheuristics in 1986, when he first described the TS algorithm, and they remain widely used in various problems today.

In mathematical programming, a heuristic algorithm is a procedure that determines near-optimal solutions to an optimization problem. However, heuristic algorithms achieve this by sacrificing optimality, completeness, accuracy, or precision for speed. Heuristic algorithms construct feasible solutions in a certain number of steps for a specific problem set, considered to be problem-dependent. For each instance of the optimization problem to be solved, a feasible solution is given at an acceptable cost, and the degree of deviation of the feasible solution from the optimal solution is generally not predictable in advance. However, since they tend to be too greedy, they usually fall into a local optimum and thus fail to obtain a globally optimal solution. Metaheuristic algorithms are strategies that guide the search process to find near-optimal solutions. In fact, they may even accept a temporary deterioration of the solution in a specific problem, which allows them to explore the solution space more thoroughly, leading to a hopefully better solution (sometimes coinciding with the global optimum). A "random factor" is another difference between heuristic and metaheuristic algorithms. Given an input for the same problem, the heuristic algorithm performs fixed steps and outputs. This is not the case with metaheuristic algorithms, as they incorporate random variables in most of their phases which can impact the global optimum. –p0

The thesis is mainly focused about metaheuristic optimization algorithms (MHOAs), which are problem-independent methods for quickly searching and using a lot of possible solutions to find almost perfect answers to difficult optimization problems. A metaheuristic method seeks to find a near-optimal solution instead of specifically trying to find the exact optimal solution, usually has no rigorous proof of convergence to the optimal solution, and is usually computationally faster than an exhaustive search [90].

Most MHOAs divide the search process into phases of exploration and exploitation. During the exploration phase, also known as diversification, an algorithm abruptly alters the solutions and delves into the search space. Depending on the nature of those

variables, the search space might be definite or indefinite. In either case, an algorithm needs to search the most promising regions to find the global optimum. Randomly changing the variables can achieve this. Exploitation, often referred to as intensification, follows. After identifying the promising regions of a search space, a MHOA should locally search them to improve accuracy. Reducing the magnitude or rate of random changes in the solutions can achieve this. Finding a good balance between exploration and exploitation is challenging for an MHOA. Because metaheuristic algorithms are capable of both exploration and exploitation, this has led to the improvement of existing metaheuristics, the creation of new ones, and particularly the fusion of existing metaheuristics.

There are a lot of taxonomies proposed in years (for example see [91], [92], [11]). A summarized categorization of metaheuristics is presented as [12]: based on the source of inspiration (evolutionary, swarm intelligence, physical law-based, and miscellaneous), based on population size (trajectory-based and population-based algorithms), based on population movement (based on differential vector movement and algorithms based on solution creation), and by the number of parameters (free-parameter based algorithms, mono-parameter based algorithms, bi-parameter based algorithms, tri-parameter based algorithms, penta-parameter based algorithms, and miscellaneous (more than five parameters). A recent focus on the area of metaheuristics is also on hyper-heuristics, which are search techniques for selecting, generating, and sequencing metaheuristics to solve challenging optimization problems. Generally, hyper-heuristics employ lower-level heuristics and/or meta-heuristics to select or generate the most suitable (meta)-heuristics, as a lower-level heuristic or a meta-heuristic algorithm can perform this task more efficiently than traversing the search space itself [93].

Various studies have examined the application of metaheuristics in different fields. For example, metaheuristics have been applied to food processing and production technologies, as well as other system-wide optimizations such as transportation, warehousing, production planning, and scheduling [94]. Fields as engineering design, digital image processing, computer vision, networks and communications, power and energy management, data analysis and machine learning, robotics, medical diagnosis, and many other fields have successfully implemented nature-inspired metaheuristics [95]. Text classification [96], text clustering [97], solving classical and emergent

problems in the finance area [98], microarray gene expression data [99], drones [92], photovoltaic generators, lithium ion batteries, and PEM fuel cells [100] are among the other applications of metaheuristics. However, the metaheuristic research continues to expand significantly, and in the future more applications in new fields could be proposed [12].

There is currently a trend to develop improved versions of established algorithms. Unfortunately, these studies present a deficient level of innovation because they only recombine well-known optimization components. Sörensen gives the overview that there are more than enough "novel" methods, and there is no need for the introduction of new metaphors [101]. Reimagining new algorithms under new names and hiding their components under metaphorical language is also, undoubtedly, the most harmful practice. This is the case of things like Black Widow Optimization and Coral Reef Optimization, which are not really innovative but rather inefficient mixtures of evolutionary operators, specifically GA, as shown by the authors [13].

1.4.2 Recently proposed metaheuristics

Approximately 540 new metaheuristics have been developed, with about 385 of them appearing in the last decay, and only in 2022 alone, around 47 ‘novel’ metaheuristics were proposed [12]. In this paragraph are listed some recent proposed metaheuristics for year 2023 until May 2024. The research was conducted using Google Scholar. The keywords used were “novel + metaheuristics + 2023, or 2024”. Table 1 stores this information. Only newly proposed metaheuristics, not "novel" enhancements of pre-existing ones, served as the basis for the selection.

Table 1.1. The list of the metaheuristics proposed in 2023-2024

No.	MHOA	Inspiration	Year	Reference
1	Squid Game Optimizer	The primary rules of a traditional Korean game	2023	[102]
2	Red-tailed hawk optimization algorithm	The hunting strategy of the bird from detecting the prey until the swoop stage	2023	[103]
3	Energy Valley Optimizer	Principles regarding stability and different modes of particle decay	2023	[104]
4	The drawer algorithm	The selection of objects from different drawers to create an optimal combination	2023	[105]
5	Crayfish Optimization Algorithm	Simulates crayfish’s summer resort behaviour, competition behaviour and foraging behaviour	2023	[106]
6	American Zebra	The social behaviour of American zebras in the wild	2023	[107]

7	Walrus Optimization Algorithm	The process of feeding of mammal (placed on Arctic Ocean) when migrating, fleeing, and fighting predators	2023	[108]
8	Mother optimization algorithm	Mimics the human interaction between a mother and her children	2023	[109]
9	Coati Optimization Algorithm	Mimics the diurnal mammal's coati behaviour in hunting, and the process of confronting, and escaping from predators	2023	[110]
10	Red Piranha Optimization	Mimics the hunting behaviour of Red Piranha fish	2023	[111]
11	Golf Optimization Algorithm	Regulations governing the game of golf, coupled with strategic considerations reflective of players' tactics	2023	[112]
12	Sand cat swarm optimization	Mimics the sand cat behaviour in searching and attacking the preys	2023	[113]
13	Red Panda Optimization	Imitates the foraging strategy of red pandas in nature	2023	[114]
14	Osprey Optimization Algorithm	Imitates the behaviour of osprey bird in hunting fish and carrying fish to a suitable position to eat it	2023	[115]
15	Deep sleep optimizer	Mimics the sleeping patterns of humans to solve optimization problems	2023	[116]
16	Lyrebird Optimization Algorithm	Imitates the natural behaviour of lyrebirds when they sense potential danger	2023	[117]
17	Sinh Cosh Optimizer	Based on the mathematical inspiration of the characteristics of Sinh and Cosh	2023	[118]
18	Migration Algorithm	The process of human migration, which aims to improve job, educational, economic, and living conditions	2023	[119]
19	Skill Optimization Algorithm	Human efforts to acquire and improve skills	2023	[120]
20	Gold rush optimizer	Simulates how gold-seekers prospected for gold during the Gold Rush Era	2023	[121]
21	One-to-One-Based Optimizer	To use the knowledge of all members in the process of updating the algorithm population while preventing the algorithm from relying on specific members of the population	2023	[122]
22	Fire Hawk Optimizer	The foraging behaviour of whistling kites, black kites, and brown falcons	2023	[123]
23	Waterwheel Plant Algorithm	Modelling the waterwheel plant's natural behaviour while on a hunting expedition.	2023	[124]
24	Gazelle Optimization Algorithm	Gazelles' survival ability in their predator-dominated environment.	2023	[125]
25	Spider wasp optimizer	Replicating the hunting, nesting, and mating behaviours of the female spider wasps in nature	2023	[126]
26	Snow ablation optimizer	Emulates the sublimation and melting behaviour of snow	2023	[127]
27	Green Anaconda Optimization	Mechanism of recognizing the position of the female species by the male species during the mating season and the hunting strategy of green anacondas	2023	[128]
28	Nutcracker optimization algorithm	Mimicking the search, cache, and recovery behaviours of nutcrackers	2023	[129]
29	Young's Double-Slit Experiment optimizer	Young's double-slit experiment, which proved that light does indeed act like a wave	2023	[130]
30	Dark Forest Algorithm	Dark forest law (a civilization once discovered will inevitably be attacked by other civilizations in the universe).	2023	[131]
31	Coronavirus metamorphosis optimization algorithm	Inspired by coronavirus disease 2019 (COVID-19)	2023	[132]
32	Migration Search Algorithm	Based on the way individuals communicate with one another and the dynamic migration behaviour of animal populations	2023	[133]

		as they explore the world		
33	Calico Salmon Migration Algorithm	Inspired by the natural behaviour of calico salmon during their migration for mating	2023	[134]
34	Subtraction-Average-Based Optimizer	To use the subtraction average of searcher agents to update the position of population members in the search space	2023	[135]
35	Tree optimization algorithm	Inspired from the growth of trees	2023	[136]
36	Quad tournament optimizer	Several candidates generated by several search agents are competed in the tournament mechanism to find the best candidate for replacement.	2023	[137]
37	Great Wall Construction Algorithm	The competition and elimination mechanisms observed among workers during the construction of the ancient Great Wall	2023	[138]
38	Sea-horse optimizer	The movement, predation and breeding behaviours of sea horses in nature	2023	[139]
39	Mountaineering Team-Based Optimization	The social behaviour and human cooperation needed in considering the natural phenomena to reach a mountaintop	2023	[140]
40	Exponential Distribution Optimizer	Based on the exponential probability distribution model	2023	[141]
41	Growth optimizer	Originates from the learning and reflection mechanisms of individuals in their growth processes in society	2023	[142]
42	Group learning algorithm	Emerged from the way individuals inside a group affect each other, and the effect of group leader on group members	2023	[143]
43	Liver Cancer Algorithm	Mimics the liver tumour growth and takeover process	2023	[144]
44	Al-Biruni earth radius	Computation of the earth radius to estimate the search space surrounding the solutions in the cooperative behaviour of swarm members to fulfil their global goals	2023	[145]
45	Kepler optimization algorithm	Kepler's laws of planetary motion.	2023	[146]
46	Cubature Kalman Optimizer	Estimation ability of the cubature Kalman filter (CKF). CKF algorithm is used to estimate the true value of a hidden quantity from an observation signal that contain an uncertainty	2023	[147]
47	Kookaburra Optimization Algorithm	The natural behaviour of kookaburras in nature when hunting and killing prey	2023	[148]
48	Swarm magnetic optimizer	Imitates the behavior of two magnets close to each other: pushing toward or pulling away from each other.	2023	[149]
49	Optical microscope algorithm	The magnification capabilities of an optical microscope on the target object	2023	[150]
50	Quadratic Interpolation Optimization	Generalized quadratic interpolation and its applications to real-world engineering problems	2023	[151]
51	Lotus Effect Algorithm	Combines efficient operators from the dragonfly algorithm with the self-cleaning feature of water on flower leaves known as the lotus effect	2024	[152]
52	Partial Reinforcement Optimizer	Psychological theory in evolutionary learning and training	2024	[153]
53	Pufferfish Optimization Algorithm	Imitates the natural behaviour of pufferfish in nature	2024	[154]
54	Crested Porcupine Optimizer	Various defensive behaviours of crested porcupine to protect themselves against predators	2024	[155]

55	Elk herd optimizer	Breeding process of the elk herd.	2024	[156]
56	Stadium spectators optimizer	Actions of stadium spectators affecting behaviour of players during a match	2024	[157]
57	Ship rescue optimization	The motion process of ship rescue according to the ship manoeuvring equation of motion	2024	[158]
58	Football team training algorithm	Training method of the football team,	2024	[159]
59	Greylag Goose Optimization	Seasonal migrations of the greylag goose	2024	[160]
60	Artificial Protozoa Optimizer	To model the foraging, dormancy, and reproduction behaviour of protozoa mimics	2024	[161]
61	Puma Optimizer	Inspired from the intelligence and life of Pumas	2024	[162]
62	Snow Geese Algorithm	Inspired by snow geese flight behaviour	2024	[163]
63	Walrus optimizer	Inspired by the behaviours of walruses that choose to migrate, breed, roost, feed, gather and escape by receiving key signals (danger signals and safety signals)	2024	[164]
64	Hyperbolic Sine Optimizer	Inspired by hyperbolic sinh function	2024	[165]
65	Object-Oriented Programming Optimization Algorithm	Inspired by the inheritance concept of Object-Oriented programming languages, where the features of a class are classified into three types according to inheritance probability: public, private, and protected	2024	[166]
66	Geyser Inspired Algorithm	Geological phenomenon named geyser	2024	[167]
67	Prism Refraction Search	Refraction of light through a triangular prism	2024	[168]
68	Literature research optimization algorithm	The mathematical model of the literature research process	2024	[169]
69	Tactical unit algorithm	Address the optimal chiller loading (OCL) problem in parallel chiller systems and other energy system	2024	[170]
70	Horned lizard optimization algorithm	Mimics crypsis, skin darkening or lightening, blood-squirting, and move-to-escape defense methods in lizards	2024	[171]
71	Nizar Optimization Algorithm	Determine the effective points by using the effective mappings and individuals of the population	2024	[172]
72	Hiking Optimization Algorithm	Hiking based on Tobler's Hiking Function	2024	[173]
73	Electric eel foraging optimization	Foraging behaviours exhibited by electric eels in nature.	2024	[174]
74	Hippopotamus optimization algorithm	Inherent behaviours observed in hippopotamuses, position updating in rivers or ponds, defensive strategies against predators, and evasion methods,	2024	[175]
75	Crested Porcupine Optimizer	Various defensive behaviours of crested porcupine: sight, sound, odor, and physical attack	2024	[176]
76	Botox Optimization Algorithm	Botox operation mechanism	2024	[177]
77	Leaf in Wind Optimization	The natural phenomenon of falling leaves in the wind	2024	[178]
78	GOOSE algorithm	Based on the goose's behaviour during rest and foraging	2024	[179]
79	Election Optimizer Algorithm	By the democratic electoral system, focusing on the presidential election	2024	[180]
80	Red-billed Blue Magpie Optimizer	The cooperative and efficient predation behaviours of red-billed blue magpies	2024	[181]

In this period, around 80 new MHOAs were created, of which 50 were only in 2023, and the trend from 2024 until May 2024 shows that this number will increase in the future. This preliminary search just verifies the increasing trend and inspiration in proposing metaheuristics with the aim to improve different optimization problems, including FS.

1.4.3 Metaheuristic algorithms and feature selection

The process of FS, which aims to identify and retain only the most useful features while discarding noisy, non-informative, irrelevant, and redundant information, can improve machine-learning models. MHOAs can be used to solve various problems, including the FS problem [91], [7], [72]. The ability to work without gradients, their adaptability, their simplicity, and their independence from the specific problem [87], explain the preference to use also them in FS.

A fitness function, also known as an objective function, plays a crucial role in optimization by quantifying the degree to which a particular solution achieves the desired outcome. The fitness function assesses the quality of a solution by assigning a numerical value, typically called a fitness score or objective value, based on how well it meets certain criteria or objectives. Metaheuristics initiate the search process by selecting random solutions, with the goal of finding an improved solution in each iteration. Exploration is vital in the initial iterations for discovering the entire search space, whereas exploitation is crucial in the last iterations for locating better solutions. They generate new solutions by applying the fitness function, and based on the unique characteristics of each algorithm, they produce a list of the top solutions. The feature selection process primarily employs the following steps, as illustrated in Figure 1.4. Accuracy, stability, scalability, and computing cost are the main difficulties are encountered when using metaheuristics in FS [182], [7].

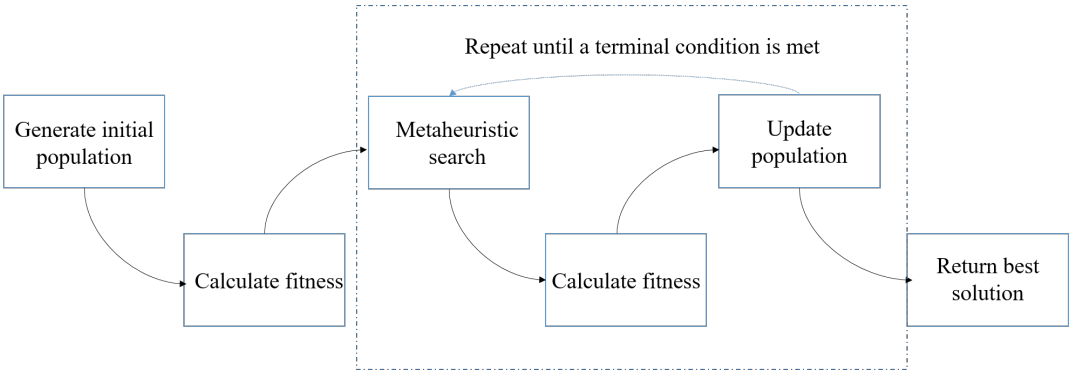


Figure 1.4. The main steps performed by metaheuristics on feature selection [7]

Dokeroglu et al. conducted a comprehensive analysis and study of the application of metaheuristics in FS [7]. They concluded that there have been over 200 000 published papers related to classical FS techniques, with GA and PSO being the most studied metaheuristics. With over 23,000 search results, the ABC, FA, and CS were the three metaheuristics most frequently cited [7]. PSO, GWO, and GA were the most frequently used metaheuristics in FS problem [183].

Typically, metaheuristics generate continuous values, rendering them unsuitable for direct application to FS. Different discretization and binarization methods can be applied on the continuous metaheuristics [184] in order to be adapted for the feature selection conditions. The proposed binarization conglomerate contains two main group classifications. The first group, known as the two-step binarization method, maps the binarization onto an intermediate space. These techniques allow working with the continuous metaheuristics without operator modifications and include two steps after the original continuous iteration. These two steps transform the continuous solution into a binary one. The second group, known as continuous-binary operator transformation, adapts the metaheuristic operator to a binary problem.

The two-step binarization technique involves using transfer functions (TF) to transform continuous values within the range of 0 to 1, and then transferring the real number using methods as standard or complement to convert them into binary values, 0 or 1. In 1997, Kennedy et al. [185] introduced transfer functions in the field of optimization. Their main advantage is that they provide a probability between 0 and 1 at a low computational cost. These methodologies facilitate the utilization of continuous metaheuristics without necessitating any modifications to the operators. The value 1 in FS indicates the selection of the feature; if not, it remains unselected. The MHOA's optimal solution involves storing the more significant features of the dataset in a vector of features. There are several types of transfer functions, with the most used S-shaped and V-shaped TFs [186]. There are also other types such as the X-shaped [187], U-shaped [188], Z-shaped [189], linear [190], and quadratic [191] TFs. For mathematical details, see for example [192]. Table 1.2 provides a description of the equations for the S-shaped and V-shaped transfer functions which are mostly used, and also in this thesis.

Table 1.2. The equations of S-shaped and V-shaped transfer functions

S-shaped		V-shaped	
Name	Equation	Name	Equation
S_1	$T(x) = \frac{1}{1+e^{-2x}}$	V_1	$T(x) = \left\lvert \text{erf}(\frac{\sqrt{\pi}}{2}x) \right\rvert$
S_2	$T(x) = \frac{1}{1+e^{-x}}$	V_2	$T(x) = \lvert\tanh(x)\rvert$
S_3	$T(x) = \frac{1}{1+e^{-\frac{x}{2}}}$	V_3	$T(x) = \left\lvert \frac{x}{\sqrt{1+x^2}} \right\rvert$
S_4	$T(x) = \frac{1}{1+e^{-\frac{x}{3}}}$	V_4	$T(x) = \left\lvert \frac{2}{\pi}\arctan(\frac{\pi}{2}x) \right\rvert$

The second step of two-step binarization includes the transformation of the search agent into a binary solution by using some methods. Two popular ones are: standard, and complement method, but are also others as static probability, elitist, and roulette elitist [192]. The standard method is calculated as in Eq. (1.2):

$$x_i^d(t+1) = \begin{cases} 0 & if\ rand < TF_1(t+1) \\ 1 & if\ rand \geq TF_1(t+1) \end{cases} \quad (1.2)$$

Meanwhile the complement one as (Eq. (1.3)):

$$x_i^d(t+1) = \begin{cases} x_i^d(t)^{-1} & if\ rand < TF_2(t+1) \\ x_i^d(t) & if\ rand \geq TF_2(t+1) \end{cases} \quad (1.3)$$

The transfer functions are not always preferred for binarization. In a total of 82 research papers, examining 22 different metaheuristics, 56.1% of the studies do not employ a transfer function in the process of binarization. Out of the remaining 36 studies, 25 of them (69.44%) use S-shaped transfer functions, while 18 of them (50%) use V-shaped transfer functions [7].

A new binarization strategy, utilizing the Q-Learning (QL) algorithm was proposed [193] to address the challenges faced in employing binarization techniques in MHOAs built for addressing continuous problems. QL serves as the intelligent operator in the proposal, employing a two-step binarization technique based on a reward system. The inclusion of QL in the selection of binarization schemes resulted in variations in the percentages of exploration and exploitation. This led to improvements in the quality of

the solutions compared to techniques that do not incorporate these disturbances in their static versions. An analysis on various combinations of four families of transfer functions (S-shaped, V-shaped, X-shaped, and Z-shaped) and the five binarization methods: standard, complement method, static probability, elitist, and roulette elitist was conducted. The techniques were utilized on 45 distinct instances of the set covering issue problem. The experimental findings demonstrated that the sets of activities that include at least one elite or elitist roulette binarization rule yielded the most favorable outcomes in terms of fitness and were statistically superior to the other sets of actions [192]. Moreover, the influence of binarization rules on the effectiveness of metaheuristic algorithms is more significant than the transfer functions.

1.4.4 Proposed approaches in improving metaheuristics

Researchers have developed various techniques to enhance the performance of metaheuristics for feature selection. Techniques as chaotic maps, local search, and fuzzy learning enhance the exploration and exploitation capabilities of metaheuristics, potentially leading to superior solutions [183]. Operator modifications, opposition-based learning, chaotic maps, Levy flight, and transfer functions are the most commonly used operators and components to enhance the performance of metaheuristics, accounting for 24%, 15%, 12%, 7%, and 5% of the total [13].

Nine categories, including new operators, hybridization, update mechanism, modified population structure, different encoding scheme, new initialization, new fitness function, multi-objective, and parallelism, are proposed also as modification techniques in a total of 156 articles in the modified nature-inspired algorithms on FS domain [194]. Examples of new operators include chaotic maps, rough sets, selection operators, sigmoidal functions, transfer functions, crossover, mutation, and Levy flight, among others. The initialization control parameters of population-based metaheuristic algorithms play a significant role in improving their performance. Researchers have identified this significance, and they have put much effort into finding various distribution schemes that will enhance the diversity of the initial populations of the algorithms and obtain the correct balance of the population size and number of iterations that will guarantee optimal solutions for a given problem set. As for example, in three metaheuristics—BA, the GWO and the butterfly optimization algorithm (BOA)— were tried different distributions to start the process, including random, beta,

uniform, logarithmic normal, exponential, Rayleigh, Weibull, Latin hypercube sampling, and sobol [195]. It was concluded that BA is sensitive to the initialization schemes, whereas GWO and BOA are not.

Exploitation strategies are known to enhance convergence speed toward a global optimum, but they are also known to increase the probability of entrapment into local optima. Conversely, search strategies that favor exploration over exploitation tend to increase the probability of finding regions within the search space where the global optimum is more likely to be located, but this comes at the cost of deterioration in the algorithm's convergence speed. In a test of 42 benchmark optimization functions, including multimodal, unimodal, hybrid, and shifting ones, the optimal outcome was achieved when the balance between exploitation and exploration was greater than 90% for exploitation and less than 10% for exploration [196]. It can be argued that the metaheuristic schemes could improve their results by gradually reducing the number of their search agents Under such conditions, it is better to consider in the search strategy only the best B solutions from the existent N (B<<N), and the computational complexity can be reduced.

Chaos theory is a relevant framework to explore when metaheuristic algorithms become trapped in a local optimum, often known as premature convergence [197]. Implementing chaos theory is the most appropriate method for resolving those issues. It possesses the characteristics of non-repetition, ergodicity, and dynamism. The dynamic property of algorithms ensures that a diverse range of solutions is produced by exploring different landscapes in the search space. Additionally, the ergodicity and lack of repetition boost the speed of the search process. Chaotic optimization not only increases the algorithm's speed but also diversifies the movement pattern.

Tizhoosh [198] was the first to propose a well-known technique - opposition-based learning (OBL) in the field of intelligence computation. The goal of the optimization technique-based OBL approach is to find a more beneficial solution between the current search agents and their corresponding opposing solutions. This method typically initializes the current search agents randomly. The fitness values for both solutions are calculated, and the best one is picked to proceed to the following iteration.

1.4.5 Hybrid feature selection methods

Usually, filter, wrapper methods, and metaheuristics are employed in FS in predicting Parkinson's. A machine learning-based detection of Parkinson's disease is proposed for the D6 dataset [199]. Boruta, RFE, and Random Forest (RF) have been used for the feature selection process. Four classification algorithms are considered to detect Parkinson disease. It was found that bagging with RFE outperformed the other methods. The lowest number of voice characteristics for Parkinson's diagnosis achieved 82.35% accuracy. The filter FS method, minimum redundancy maximum relevance (MRMR), was used to select a convenient number of features for the D1 dataset. The random forest with 20 features selected by mRmR feature selection algorithms provides overall accuracy of 90.3%, which is better in comparison to all other machine learning-based approaches [200]. In another combination, XGBoost classifier with mRMR feature selection method outperformed all other models with a high accuracy of 95.39% when both MFCC and TQWT features were selected in D5 dataset [201]. GA is used as an FS method on the biomedical voice D1 dataset for 10 classifiers. GA had a significant impact on the accuracy of the two most accurate models, Decision Tree (DT) and Naïve Bayes (NB), with a final improved accuracy of 97.96% in predicting PD [202]. GA and PSO are used to determine the optimal subset of features on the D5 dataset [203]. These subsets are evaluated by 11 ML classifiers. Out of all the bio-inspired machine learning classifiers used, the following ones were the most effective: GA-inspired Ada Boost (AB) delivered the maximum classification accuracy of 90.7%, producing the dimensionality reduction of 41.43% by selecting only 441 features. BPSO with MLP achieved a maximum classification accuracy of 89% and produced 46.48% of the dimensionality reduction by selecting only 403 features. The Flower Pollination Algorithm and Extreme Gradient Boost Algorithm pairs achieved the highest testing accuracy of 93% in detecting PD on the D5 dataset [204].

Hybrid metaheuristics, in fact, represent the most efficient algorithms for many classical and real-life difficulties. The vast array of efficient hybrid metaheuristics proposed to address a wide range of problems demonstrates this. Nowadays, combining metaheuristics has become a common strategy to solve optimization problems. Hybridization in the context of metaheuristics refers to combining two or more metaheuristic algorithms to create a new, often more efficient, method. Over the years,

there has been a noticeable fluctuation in interest in metaheuristic hybridization, according to the data.

Hybrid metaheuristics can be classified based on many objectives, such as the level of hybridization, the order of execution, and the control strategy [205]. Hybrid metaheuristics are distinguished into two types based on the level at which the various algorithms are combined: high-level and low-level combinations. High-level combinations retain the individual identities of the original algorithms while cooperating over a relatively well-defined interface. In contrast, low-level combinations heavily rely on each other, exchanging individual components or functions of the algorithms. Hybrid metaheuristics can also be divided based on the execution process as batch, interleaved, or parallel. Based on their control strategy, the hybrid metaheuristics can be classified into integrative (coercive) and collaborative (cooperative) combinations. In integrative approaches, one algorithm is considered a subordinate or embedded part of another. Collaborative combinations of algorithms share information but do not embed it.

In the hybridization process involving two metaheuristics, one typically serves as the foundational or base algorithm, while the other acts as an enhancement. This enhancement specifically targets and strengthens aspects of the base metaheuristic that may be perceived as weaker than other metaheuristics. From 2019 to April 2023, a review of 24 articles revealed various metaheuristics employed as foundational or base algorithms in the hybridization process [183]. Three different studies have employed the grey wolf optimizer (GWO) as their base, making it the most frequently used foundational metaheuristic. With a count of six, simulated annealing (SA) emerges as the most frequently used metaheuristic for enhancement.

Four different types of combinations are proposed related to hybrid metaheuristics [206]: (1) Combining metaheuristics with complementary metaheuristics, such as combining P-metaheuristics (evolutionary algorithms, scatter search, and ant colonies), which are more efficient in diversifying the search space, with S-metaheuristics (local search, tabu search, etc.), which are more intensification-based search algorithms, has resulted in the development of very powerful search algorithms. (2) Metaheuristics are combined in conjunction with precise methods from mathematical programming

approaches; (3) Metaheuristics are combined with constraint programming approaches; (4) Metaheuristics combined with machine learning techniques.

For instance, the BDMSAO metaheuristic combines the binary variants of the dwarf mongoose optimization (P-metaheuristic) and the simulated annealing algorithm (S-metaheuristic) [207]. The hybrid BDMSAO algorithm employs the BDMO as the global search method and uses SA as the local search component to enhance the limited exploitative mechanism of the BDMO. The new hybrid algorithm was evaluated using eighteen UCI machine learning datasets of low and medium dimensions. The BDMSAO was also tested using three high-dimensional medical datasets to assess its robustness. The results demonstrated the BDMSAO's efficacy in solving challenging feature selection problems on varying dataset dimensions, as well as its outperformance over ten other methods in the study.

Another classical hybrid approaches categorization is presented in [208]:

- Low-level relay hybrid- a MHOA is embedded into a single-solution algorithm.
- High-level relay hybrid (HRH) – two MHOAs are executed in sequence in homogenous (same algorithms) or heterogeneous (different algorithms) manner
- Low-level teamwork hybrid (LTH) – a MHOA is embedded into a population-based algorithm
- High-level teamwork hybrid – two metaheuristics are executed in parallel

An enhanced hybrid metaheuristic approach that uses GWO and WOA was proposed as a hybridization variant [209]. The main objective of the proposed technique is to alleviate the drawbacks of both algorithms, including immature convergence and stagnation in local optima. The combination of the two metaheuristics follows the high-level relay hybrid (HRH) approach. There were proposed three different strategies in this work: serial gray-whale optimizers, random switching gray-whale optimizers, and adaptive switching gray-whale optimizers named HSGW, RSGW, and ASGW. The results showed that HSGW outperformed all other approaches, proving that this combination as a better one. Another hybridization model based on WOA and SA is proposed considering (1) low-level teamwork hybrid (LTH) and (2) high-level relay

hybrid (HRH) [210]. In LTH, the WOA algorithm incorporates the SA algorithm to find the optimal solution in the vicinity of both the randomly chosen solution and the best-known solution, thereby substituting the original one. This process improves the exploitation ability of the WOA algorithm. In this approach, the SA algorithm acts as an operator within the WOA system. Conversely, the HRH model employs the SA algorithm after applying the WOA algorithm to identify the optimal solution and then using the SA algorithm to refine that final solution. The performance of the proposed approaches is tested on 18 standard benchmark datasets from the UCI repository and compares them with three well-known metaheuristics: ALO, PSO, and GA. The most effective strategy, which employs SA to enhance the nearby region of the best solution discovered in each iteration of the WOA, and utilizes tournament selection to choose the search agents with the highest performance, demonstrates superior classification accuracy compared to all other proposed methods.

1.5 Supervised learning algorithms

Machine learning is an interdisciplinary field within the study of artificial intelligence (AI) that specifically deals with the formation of algorithms and systems capable of acquiring knowledge and making predictions or choices by analyzing data. ML can be categorized in a more comprehensive manner as follows:

- **Supervised Learning**: Supervised learning is applicable to datasets that include a response variable, which is also referred to as a label. The dependent variable has the potential to be either continuous or categorical. The algorithm acquires knowledge of the response variable by analyzing the given collection of predictor variables. Additionally, the problem can be classified as a classification task if the response variable is categorical, and as a regression task if it is continuous.
- **Unsupervised Learning**: If a dataset lacks a response variable, no guidance is available for learning from the predictor variables. In other words, the learning process must be unsupervised. The learning process is determined by a measure of similarity or distance between each row in the dataset.

Clustering and Association Rule Mining are the predominant approaches employed in unsupervised learning.

- **Semi-supervised Learning**: This involves a combination of both supervised and unsupervised learning techniques. It employs a limited quantity of labeled data in conjunction with a substantial amount of unlabeled data. It is advantageous when the process of assigning labels to data is costly or time-consuming.
- **Reinforcement Learning**: Both supervised and unsupervised learning algorithms need clean and accurate data to produce the best results. Reinforcement learning is an ideal choice in cases, where only an initial state of the data is available as an input and there is no single possible response but rather many outcomes are possible. This is about training agents to make decisions by trial and error. The agent learns to achieve a goal in an uncertain, potentially complex environment. It receives feedback in the form of rewards or penalties.

When predictions are required in many categorization tasks, supervised learning algorithms are frequently used. A class output of a dataset and a list of numeric and non-numeric variables are the two inputs for supervised learning techniques. Supervised learning includes algorithms where a target feature has known values, and the learning and testing of the algorithm is done by dividing the full dataset in training and test dataset. The training dataset is used for building a model and accuracy of the trained model is controlled from the test dataset. In our situation, these techniques are utilized to determine whether a person has PD or not.

ML is largely used in predicting different illnesses in people. Classification is applied most frequently in the medical industry, and fields such as cancer, Alzheimer's, Parkinson's, and renal disease have all seen a lot of use of neural networks combined with other supervised algorithms [211]. Mei et al. carried out an in-depth review of the literature on machine learning techniques employed in the diagnosis and differential diagnosis of PD [71]. A review of the research published up through February 14, 2020, was performed. The analysis included a total of 209 papers. On average, 2.14 machine learning models per study were applied to the diagnosis of PD [71]. Regarding the

supervised machine learning algorithms, out of 448 methods from 209 studies, 132 were support vector machines (SVM) and their variants, 76 were neural network algorithms, 82 were ensemble learning algorithms, and 33 were nearest neighbor algorithms and their variants. Rana et al. identified existing ML-based research to diagnose Parkinson's disease in terms of handwritten patterns, voice attributes, and gait datasets and determined the most appropriate techniques [212]. L1-Norm SVM with K-fold cross-validation produced the best accuracy for voice features to diagnose PD (99%), for handwritten patterns, bagging ensemble (97.96%), and for gait analysis, SVM (100%).

1.5.1 k-nearest neighbour

It is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. It is first developed by Evelyn Fix and Joseph Hodges in 1951, and later expanded by Thomas Cover. K-Nearest Neighbor is a popular and widespread supervised learning algorithm in feature selection [91]. Regarding using k-NN in feature selection, it is highly used with a ratio of 60.98% from an overall of 82 selected studies [7], and in 77% of the selected papers in another review [183].

This algorithm is typically used as a classification algorithm, working off the assumption that similar points can be found near one another. For classification problems, a class label is assigned on the basis of a majority vote—i.e., the label that is most frequently represented around a given data point is used. K-NN is easy to understand and implement, and it can handle both linear and nonlinear data. However, its performance can be sensitive to the choice of K, the scale of the features, and irrelevant features [213].

Nearest-neighbor classifiers follow a very similar idea of learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it. The training tuples are described by n attributes. Each tuple represents a point in an n-dimensional space. In this way, all the training tuples are stored in an n-dimensional attribute space. When given an unknown tuple, a k-nearest-neighbor classifier searches the attribute space for the k training tuples that are closest to the unknown tuple. These k training tuples are the k “nearest neighbors” of the unknown tuple. Then k-nearest-neighbor

classifier chooses the most common class label among the k nearest neighbors as the predicted class label of the unknown tuple

"Closeness" is defined in terms of a distance metric, such as Euclidean distance. The Euclidean distance between two points or tuples, say $X_1 = (x_{11}, x_{12}, \dots, x_{1n})$, and $X_2 = (x_{21}, x_{22}, \dots, x_{2n})$. In other words, for each numeric attribute, we take the difference between the corresponding values of that attribute in tuple X_1 and in tuple X_2, square this difference, and accumulate it. The square root is taken of the total accumulated distance count. Eq. (1.4) shows the calculation of this distance.

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n}(x_{1i} - x_{2i})^2} \tag{1.4}$$

Some advantages, and disadvantages of k-NN are [214]:

- K-NN algorithms are easy to implement, adapts easily when new training samples are added, and has only a hyper parameter which makes it easier to use.

Meanwhile disadvantages are:

- It takes up more memory and data storage compared to other classifiers, tends to fall victim to the curse of dimensionality.
- k-NN is also more prone to overfitting.

1.5.2 Support vector machines

SVM is a supervised algorithm with the objective to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points. To separate the two classes of data points, there are possible hyperplanes that could be chosen. The objective is to find a plane that has the maximum margin, i.e., the maximum distance between data points of both classes. The SVM finds this hyperplane using support vectors ("essential" training tuples) and *margins* (defined by the support vectors). Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. The first paper on support vector machines was presented in 1992 by Boser, Guyon & Vapnik [215], even though the groundwork for SVMs has been around since the 1960s. SVMs can be used for numeric prediction and classification. SVM can be: 1) Linear SVM, which are used for linear regression and classification problems, and 2) Nonlinear SVM, which are applied for

the classification of linearly inseparable data. Such SVMs are capable of finding nonlinear decision boundaries. In the first step, are transformed the original input data into a higher dimensional space using a nonlinear mapping. The second step searches for a linear separating hyperplane in the new space using linear SVM. If a linear separator couldn't be found, observations are usually projected into a higher-dimensional space using a kernel function where the observations effectively become linearly separable. Examples are polynomial kernel, Gaussian radial basis function, sigmoid etc. Gaussian radial basis function kernel is used in this thesis. Its equation is presented in Eq. (1.5):

$$K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2} \tag{1.5}$$

where σ is the variance, $\|X_i - X_j\|^2$ is the squared Euclidean distance between the two feature vectors.

SVM offer some advantages, and disadvantages [216]. It works relatively well when there is a clear margin of separation between classes, very effective in high dimensional spaces, has a solid theoretical foundation, and generalization capacity. Some weaknesses of SVM are the selection of parameters, algorithmic complexity that affects the training time of the classifier in large data sets, development of optimal classifiers for multi-class problems and the performance of SVMs in unbalanced data sets.

1.5.3 Random Forest

The Random Forests (RFs) popular algorithm was introduced in [57]. It is composed of multiple independent decision trees that are trained independently on a random subset of data. RF trains several decision tree classifiers (in parallel) on various subsamples of the dataset and various subsamples of the available features. RF are examples of ensembles methods. This method combines a series of k models (classifiers) $M_1, M_2, \dots, M_k$ with the aim of creating an improved classification model M*. A data set D is used to create k training sets, $D_1, D_2, \dots, D_k$where $D_i (1 \leq i \leq k)$ is used to generate classifier M_i. Given e new data tuple to classify, the base classifiers each vote by returning a class prediction. The ensemble returns a class prediction based on the votes of the base classifiers. RFs tend to have high accuracy prediction and can handle large numbers of features due to the embedded feature selection in the model generation

process. They reduce the risk of overfitting, provides flexibility, and easiness to determine feature importance, but they are time-consuming, requires more resources to store the data [217].

1.5.4 Performance metrics

In this subsection are listed the more frequent used metrics for measuring the performance of the classifiers, and evaluating the subset of the generated features by the FS methods. Fitness functions are an important tool for evaluating the quality of solutions generated by the FS methods. The weighted multi-objective functions, including such as accuracy, number of selected features, error rate, etc. are the most commonly used objective function classification in the related literature (73% of the collected research papers) [183]. Usually, fitness function is used to find a balance between the number of selected features and classification accuracy The most common fitness function is calculated in Eq. (1.6):

$$Fitness = 0.99 * CE + 0.01 * {}^{n_f}/_{n_t} \quad (1.6)$$

where n_f is the number of selected features, n_t is the number of total features. CE is the classification error = 1 – accuracy. Other popular metrics considered related with the fitness metric are best, worst, average, and standard deviation of fitness formulated in Eqs. (1.7, 1.8, 1.9, 1.10):

Best fitness: It represents the smallest fitness value provided in total executed runs, where fi is the fitness function.

$$best\ fitness = min^{nruns}_{i=1} f_i \quad (1.7)$$

Best fitness: It represents the largest fitness value provided in total executed runs.

$$\text{worst}\ fitness = max^{nruns}_{i=1} f_i \quad (1.8)$$

Average fitness: It represents the mean of the best fitness provided in all the runs.

$$Average\ fitness = \frac{1}{nruns} \sum_{i=1}^{nruns} f_i, \quad (1.9)$$

Standard deviation describes the variation of the optimal fitness from the average fitness.

$$Standard\ deviation = \sqrt{\frac{1}{nruns-1}\sum_{i=1}^{nruns}(f_i - average\ fitness)^2} \quad (1.10)$$

Two last metrics are average accuracy, and number of selected features, respectively in Eq. (1.11), and Eq. (1.12).

Classification accuracy: It is a metric that defines how accurate a classification model is for a given set of features. It is calculated as:

$$Average\ accuracy = \frac{1}{nruns}\sum_{j=1}^{nruns}\frac{1}{N}\sum_{i=1}^{N} match(C_i, L_i), \quad (1.11)$$

where N is the number of test points, C_i denotes the output label for data point i, match denotes the comparator that returns 0 when two labels are not identical, and 1 when they are same, and L_i denotes the reference label for i.

Average number of selected features: It represents the average of the number of selected features (n_f) over *nruns* times and is defined as follows

$$Average\ no.features = \frac{1}{nruns}\sum_{i=1}^{nruns} n_f(i) \quad (1.12)$$

The two most often used measures for evaluating classification quality are accuracy and the F1-measure in FS [7]. When there are imbalanced classes in the dataset, the F1-score is chosen over accuracy for evaluation [7]. Almost all studies (90.24% of them) utilize more often accuracy and fitness value as a performance metric in FS [7]. The other commonly used metrics are the number of selected features and execution time, in respective order. Accuracy was the most frequently used performance metric also observed in another study (83.3% of cases) [71]. Another bibliometric analysis based on the most popular metrics used in evaluating the metaheuristics [183], confirm that accuracy is the most used classifier metric in the literature (85% of the papers), then it comes fitness and computational time with 58% and 46%. The number of features selected (NFS) is also the most used feature metric in the literature, present in 83% of the collected papers. Wilcoxon signed-rank test is the most used in the literature for comparing metaheuristics metrics (21% of the collected papers), followed by the Friedman test and Wilcoxon rank-sum test, each used in 12% of the cases.

1.5.5 Hyper-parameter optimization and metaheuristics

Machine learning (ML) algorithms are highly configurable by their hyper parameters. These parameters often substantially influence the complexity, behavior, speed as well as other aspects of the learner, and their values must be selected with care in order to achieve optimal performance. Human trial-and-error to select these values is time-consuming, often somewhat biased, error-prone and computationally irreproducible. As the mathematical formalization of hyper parameter optimization (HPO) functions most as a black-box optimization, often in a higher-dimensional space, this is better delegated to appropriate algorithms and machines to increase efficiency and ensure reproducibility. HPO is crucial in machine learning because it aims to find the best set of hyper parameters for a given model and dataset. Some popular methods for hyper parameter optimization are grid search, random search, Bayesian optimization etc.

Metaheuristics can be used as an option for optimizing the parameters of the machine learning algorithms, and has resulted very effective compared to traditional methods. GA and PSO were used to optimize parameters of Naïve Bayes, SVM, RF, Decision Tree and Multi-Layer Perceptron on seven datasets. Multinomial Naïve Bayes with Genetic Algorithm performed the best overall [218]. Hybrid metaheuristics are also employed in hyper parameter optimizations, as in [219] where a novel hyper-parameter optimization methodology is presented to combine the advantages of a Genetic Algorithm and Tabu Search for an efficient search of good values of hyper-parameters of deep convolutional neural networks This method is compared with four other methods: Bayesian optimization, SA, Covariance Matrix Adaptation Evolution Strategy and Random Search. Experimental results show that the proposed Tabu_Genetic Algorithm finds a better model in less time, and has better search capabilities as an effective method for finding the hyper-parameters of learning algorithms over some traditional optimization methods. GA, DE, ABC, GWO, PSO, and teaching learning-based optimization (TLBO), were used to optimize hyper parameters and compare these algorithms with grid search method (GS) for detecting fraud transactions on 7 classifiers [220]. The results obtained show that the proposed metaheuristic algorithms are superior to the grid search method and generate better results in a short computational time.

There are also cases when the same metaheuristic is used in the same time as a search method in FS and also in optimizing parameters of supervised learning algorithms, as for example GOA and multiverse optimizer in selecting the best parameters of SVM [221], [222] which has improved the results when are integrated for hyper-parameter tuning.

1.6 Chapter conclusions

Data management and machine learning largely utilize optimization methods, including metaheuristics. The last ones are capable of solving a variety of precise and heuristic problems, and the scope of their applications continues to expand. For solving the feature selection problem, which is a NP-hard problem, metaheuristics can be very adequate. This chapter's goal is to look closely at how metaheuristic algorithms, machine learning classification algorithms, hyper-parameter optimization, and performance metrics are used to evaluate FS subsets. The purpose of this is to demonstrate the significance and outcomes of their implementation in enhancing the precision of Parkinson's prediction and other applications. There is an increasing trend to propose new metaheuristics inspired by actual phenomena. In parallel, the binary variants of them have been created or need to be created and applied in FS in the future in order to observe their adaptability in FS. Researchers often combine different filter and wrapper methods, either with or without metaheuristics, as well as hybridize metaheuristics in diverse approaches. Filter and feature importance methods remain crucial in feature selection, especially in high-dimensional datasets, because their integration as a first step in eliminating irrelevant and unidentifiable features can significantly reduce the computational time required by wrapper methods to select dominant features. Metaheuristics offer numerous enhancements to avoid stagnation in the local optimum, with the integration of opposition-based learning, chaotic maps, and Levy flight among the most crucial techniques. Metaheuristic hybridization has proven to be highly effective not only in predicting Parkinson's, but also in enhancing the exploration and exploitation phases of the metaheuristic. There are indications that it is more necessary to invest in the metaheuristic exploitation phase than to explore endlessly for better solutions. In addition to the advantages, the researchers need to exercise caution in their investigation, enhance the experiment conditions, and carefully interpret the results of the proposed algorithms. This chapter emphasizes the

significance of each feature selection method, but there are still potential algorithms or approaches that could be proposed, which could be more effective and contribute to the FS problem.

❖ The goal of the dissertation is:

To create and develop new metaheuristic optimization algorithms for the feature selection problem and improve them in efficacy, efficiency, and performance time, contributing to Parkinson's prediction using machine learning algorithms. It encompasses the analysis of existing research methodologies and methods for selecting relevant and important features from Parkinson's data, interconnected with innovative metaheuristic algorithms used in feature selection by improving their exploration and exploitation capabilities, with the aim of maximizing Parkinson's prediction accuracy. The objective is to provide new algorithms to support forecasting Parkinson's regardless of the input data, as well as to propose a new method for identifying the most important features while reducing the data's dimensionality, maintaining a reasonable machine learning accuracy, and a reasonable execution time.

In this regard, the goal of the dissertation was achieved by the following research tasks:

Task 1: To evaluate the "state of the art" of optimization methods, mostly metaheuristics, in data management, and feature selection, for predicting Parkinson's emphasizing their importance in this field.

Task 2: To carry out a comparative analysis of different filter and wrapper methods that uses the heuristic simulated annealing algorithm for hyper-parameter optimization of three machine learning classifiers with the aim of achieving high prediction accuracy for Parkinson's.

Task 3: To propose a new effective metaheuristic algorithm named “Binary Volleyball Premier League” applied for the first time in feature selection for predicting Parkinson's.

Task 4: To enhance the effectiveness of the Binary Volleyball Premier League by incorporating an opposition-based learning technique into its final solution, which will strengthen its search space exploration in favor of increasing the accuracy of Parkinson's prediction.

Task 5: To propose a novel hybrid metaheuristic, named BVPL_BALO, that merges Binary Volleyball Premier League learning phase and Binary Antlion Optimizer phase of generating new solutions, contributing in improving the exploitation of the actual solutions in order to improve the effectiveness of Binary Volleyball Premier League.

Task 6: To incorporate an “occurrence list” procedure into the hybrid metaheuristic Binary Volleyball Premier League and Antlion Optimizer algorithm which reduces its performance time resulting in a significant improvement in its efficiency comparing to Binary Volleyball Premier League.

Task 7: To develop a new advanced method that improves the efficiency of the Binary Volleyball Premier League and Antlion Optimizer algorithms by incorporating a feature ranking algorithm to prioritize reducing the feature dimensionality on high-dimensional datasets before employing the reduced number of features into the hybrid metaheuristic BVPL_BALO.

2. The proposed metaheuristic algorithms and, methods on feature selection Parkinson-based

This chapter introduces novel metaheuristic algorithms, methods, and enhancements used for the first time in FS and Parkinson's prediction. The Volleyball Premier League algorithm, which was first proposed in 2018 for continuous optimization problems, serves as the foundational metaheuristic algorithm of the newly proposed metaheuristic algorithms. This algorithm has not been used in feature selection before and requires modification and adaptation for this purpose.

The field of metaheuristic algorithms is experiencing substantial growth, with academics actively seeking to enhance and integrate current algorithms for solving different problems. This has resulted in a significant increase in applications in this field, which seems to have been ongoing for some decades.

Parkinson is one of the most important illnesses in the field of neurological diseases, limiting patients' daily activities and affecting their physical and mental wellbeing. Researchers, the public health sector, non-profit organizations, and the public have an immense interest in understanding the factors that could worsen the effect of Parkinson's on patients. Predicting Parkinson's is important for improving patient outcomes, advancing medical research, reducing healthcare costs, and enhancing the overall quality of life for individuals at risk of developing the disease. Medical industries, research groups, and non-profit organizations generate a significant amount of data about Parkinson's, leading to an increase in patient data. This includes patient symptoms, medical history, exam measurements, biomarkers, and more recently, genetic and scanning data. This has led to an increase in the dimensionality of the provided data, coinciding with the use of methods and algorithms for selecting, reducing and identifying the most important features from Parkinson data.

The methodology for solving the proposed tasks follows these sequential steps as presented in Figure 2.1.

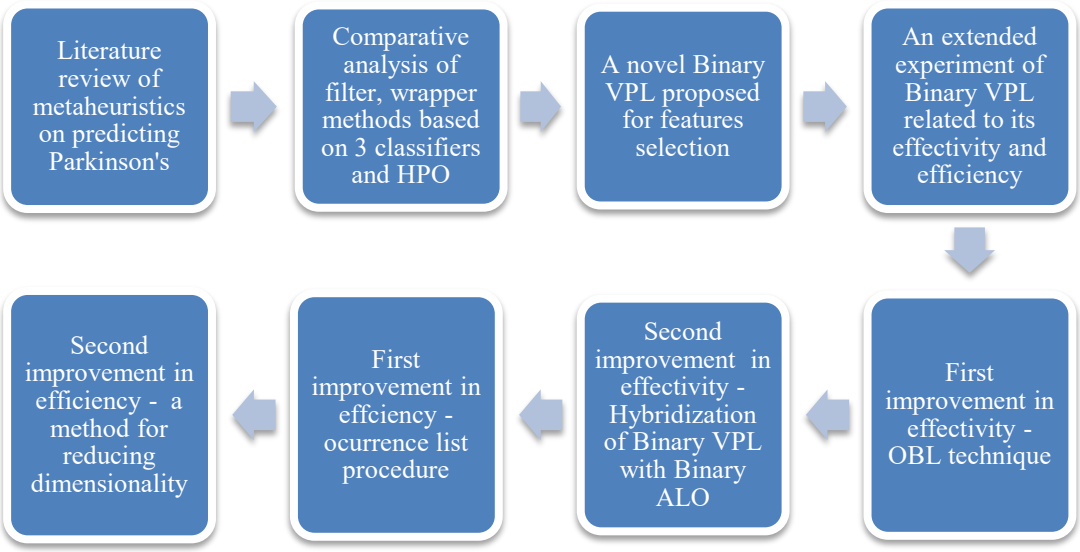


Figure 2.1. The steps of the methodology of the research

Firstly, the chapter content starts with reviewing, searching, and understanding the trend of using metaheuristics in predicting Parkinson's. Seven public datasets based on speech, vocal, and handwriting ability of people with Parkinson form the foundation of this study. The goal was to observe and analyze the use of metaheuristic algorithms on FS in Parkinson's prediction, with a focus on highlighting the best results, such as accuracy or the number of selected features.

Secondly, a first investigation by choosing some popular filter and wrapper methods of FS, including GA is applied to one public voice Parkinson's dataset in order to analyze which of the methods can predict Parkinson with a higher accuracy. The evaluation of each subset selected is done using three ML classifiers: rbfSVM, k-NN, and RF. The effect of using generalized simulated annealing (GSA) is observed in order to evaluate its effect on hyper-parameter optimization of the three classifiers.

In the next phase, a metaheuristic BVPL was proposed for FS because of its ability to search for better solutions in all the steps and for its exploitation ability. The experiments were conducted on the same Parkinson's dataset, using a k-NN classifier and comparing it with PSO. The aim is to provide a preliminary analysis of the effectiveness and efficiency of the binary VPL in FS Parkinson's prediction.

The previous experiment's good results for the proposed BVPL were crucial in evaluating and comparing the binary VPL more closely with other 19 metaheuristics, including PSO, and nine additional Parkinson's datasets, (9 publics and 1 private related

to gait analysis). This broader analysis confirms BVPL's superiority in most datasets for its effectiveness in predicting Parkinson's and shows its disadvantages in terms of execution time.

Regarding improving its effectiveness in predicting PD, two improvements of BVPL are proposed: (1) integrating the opposite solution of the final solution provided by BVPL and choosing this OBL solution if it has better fitness; and (2) improving the exploitation phase of BVPL by hybridizing it with a binary ALO, named BVPL_BALO, which has provided good results from the previous comparative analysis.

Related to improving execution time, meaning its efficiency, two improvements are proposed: (1) In BVPL_BALO, it is integrated a procedure named here *occurrence list,* which reduces the execution time by storing in advance the generated teams and the cost of each generated solution and retracting from here if it is provided anytime an existing solution; and (2) proposing a method for reducing dimensionality by including a feature ranking algorithm before employing feature selection utilizing the hybrid BVPL_BALO, which evaluates the similarity between the rows of the datasets in order to rank the features according to their importance.

This chapter satisfies the criteria outlined partly on Task 1 and continues on Tasks 2 to 7 of the dissertation. Each of the steps that follow the thesis's research methodology is continued in the subsequent subsections.

2.1 Trends of metaheuristics in feature selection Parkinson-based

Primarily, it was conducted in-depth research on the binary metaheuristics used in FS to predict Parkinson's, fulfilling a part of the requirements of Task 1. This review was necessary to understand the frequency with which metaheuristics were proposed as a solution to select the most identifiable features for Parkinson's prediction and which criteria were applied for this evaluation. A comprehensive search was done on Google Scholar and Research Gate, limiting it to publications published until 2022. The search was implemented using the keywords "metaheuristics algorithms" + "feature selection." There were selected papers that included only Parkinson datasets in their analysis. The collection of datasets under consideration consists of a total of seven, as presented in a summarized format in Table 2.1.

Table 2.1. The Parkinson's datasets

PD datasets	Description		
	Type	***Dimension***	***No. times used***
D1	Voice	195x23	22
D2_S, D2_M	Hand writing	368x16	4
D3_S, D3_M	Hand writing	264x16	1
D5	Speech	756x754	10
D7	Voice	1040x26	6

A set of exclusion and inclusion criteria were applied to an overall total of 175 conference papers and journals. The selection process involved choosing papers written in the English language and employing supervised learning algorithms for their applications. Reviews and papers that solely employ the metaheuristic technique for hyper-parameter optimization within machine learning algorithms were excluded. A total of 34 publications were finally selected. Figure 2.2 illustrates the distribution of publications, with Elsevier publishing the largest number.

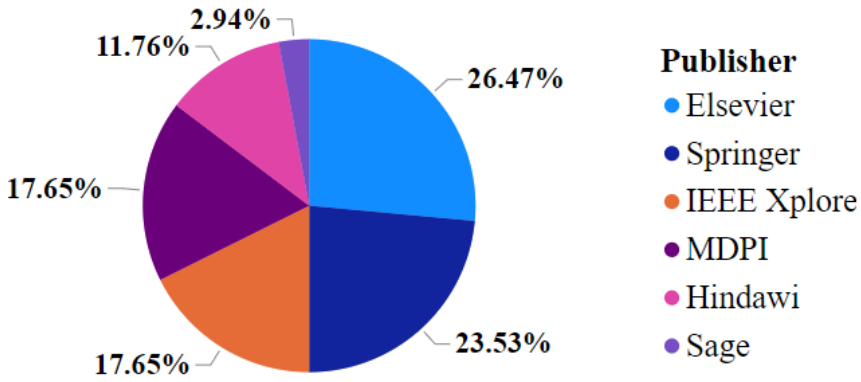


Figure 2.2. Distribution of source research papers (%)

Table 2.2 presents the best proposed metaheuristics highlighting the ML algorithm, the performance metrics, fitness function, and results related mostly with the accuracy, and number of selected features. In some cases, were used one or more Parkinson's datasets simultaneously to evaluate the metaheuristic's performance.

Table 2.2. Description of the metaheuristics, ML algorithms, metrics, fitness function, and results

Dataset	Metaheuristic	Supervised learning algorithm	Performance metrics	Fitness function	Results	Ref.
D4	GA	Ada Boost (AB)	Accuracy (acc) Precision, sensitivity, F1-score, execution time, N_{sf}	min features and max acc (not a formula)	Mean (acc) = 0.907, 441 features	[203]
D4	An improvement of ABC (ABCv2)	k-nearest neighbor (k-NN), k not specified	Acc, F1 score, MCC	Fitness is calculated using k-NN algorithm (not a formula)	Mel frequency cepstral coefficients features (MFCC) are the most discriminative features, 35 features using k-NN, mean (acc) = 86%.	[223]
D4	An improvement of PSO (PSOVA1)	k-NN (k= 5)	Acc, F1 score, N_{sf}, and computational cost	0.90*acc + 0.1* (1/ N_{sf})	mean (acc) = 81.15% and mean (N_{sf}) = 273.1	[224]
D4	Mothflame Optimization (B-MFO-V2)	k-NN (k= 5)	Acc, N_{sf}, sensitivity, and specificity	0.99*CE + 0.01 * N_{sf}/N_{tf} CE = classification error	mean (acc) = 86.03%, mean (N_{sf}) = 79.1.	[225]
D4	Grey Wolf Optimization (GWO) algorithm	Light Gradient Boosted Machine (LGBM)	Acc, precision, sensitivity, F1 score, area under receiver operating (AUC), computational time	alpha*CE + beta * N_{sf}/N_{tf} alpha and beta not specified directly	acc = 90.5 %, all features	[226]
D4	Quadratic binary Harris Hawk Optimization (Q4BHHO))	k-NN (k= 5)	Acc, and N_{sf}	0.99*CE + 0.01 * N_{sf}/N_{tf}	Logistic-Tent map, mean (acc) = 0.9083, mean (N_{sf}) = 210.00	[191]
D4	Chaotic Atom Search Optimization (ASO)	k-NN (k= 5)	Acc, and FS ratio (FSR) = N_{sf} /N_{tf}	0.99*CE + 0.01 * N_{sf}/N_{tf}	mean (acc) 0.91.67, mean (FSR) = 0.4953	[227]
D4, D5	A non-linear hybrid binary grasshopper optimization algorithm (GOA) and Whale Optimization algorithm (WOA)	k-NN (k= 5)	Acc, and N_{sf}	Not a formula	D4 – mean (acc) = 0.913, mean (N_{sf}) = 111.7; D5 – mean (acc) = 0.69.8, mean (N_{sf}) = 5.7	[228]

D4	A combination of Fuzzy Monarch Butterfly Optimization Algorithm + Levy Flight CSA + Adaptive FA	Fuzzy convolution bi-directional long short-term memory (FCBi-LSTM)	Acc, F1 score, MCC	Acc of the classifier	mean (acc) = **0.9877,** MFCC features + Wavelet + Concat (baseline, vocal fold, and time frequency features)	[229]
D1	Hybrid PSO-GWO	k-NN (k= 5)	Acc, time, N_{sf}	0.90*CE + 0.1 * N_{sf}/N_{tf}	mean (acc) = 0.92, mean (N_{sf}) = 4.25	[230]
D1	Binary Pigeon Optimization (POA)	Long short-term memory for classification, and Beetle Antenna Search Optimization for parameter optimization	Acc, time, N_{sf}	0.9*CE + 0.1 * N_{sf}/N_{tf}	max (acc) = 0.9286, min (N_{sf}) = 4	[231]
D1	Binary Bat Algorithm (BBA)	feed forward back propagation - based network	Acc, N_{sf}, misclassification rate, sensitivity, false positive rates (FPR), precision and negative predicted values	Not a formula	Acc = 0.936, N_{sf} = 6	[232]
D1, D2, D5	Modified GWO	Random Forest (RF), Decision Trees (DT)	Acc, Detection Rate and False Alarm Rate	Fitness = sum of each feature importance + weight factor * (1- (N_{sf}/N_{tf}))	D1 - acc = 0.9387 (RF); D2 spiral - acc = 0.9241 (RF, DT); D2 meander – acc = 0.9304 (RF); D5 - acc = 1.00 (RF, DT)	[233]
D1	A hybrid chaotic Crow Search (CrS) (logistic map) and PSO algorithm (exponential map)	k-NN (k not specified)	Acc, FSR	0.99*CE + 0.01 * N_{sf}/N_{tf}	mean (acc) = 0.9263, mean (FSR) = 0.2773	[234]
D1	A new binary GOA	k-NN (k= 5)	Acc, FSR, time.	0.90*CE + 0.1 * N_{sf}/N_{tf}	mean (acc) = 0.95023, mean (FSR) = 0.580)	[235]
D1	An enhanced black widow optimization (BWO) algorithm	k-NN (k= 5)	Acc, N_{sf}	0.90*CE + 0.1 * N_{sf}/N_{tf}	mean (acc) = 0.996154, mean (N_{sf}) = 2.15	[236]
D1	Improved Equilibrium	k-NN (k = 5), SVM (2 classes)	Acc, N_{sf}	0.01*CE + 0.99 * N_{sf}/N_{tf}	mean (acc) = 0.9259, mean	[237]

	Optimization Algorithm (EOA)				(N_{sf}) = 4.95	
D2, D3	Optimized CrS algorithm (OCSA)	k-NN (k=3), RF, DT	Acc, N_{sf}	Fitness = feature importance + weight factor * (1- (N_{sf}/N_{tf}))	Acc = 100% for all the three classifiers, N_{sf} = 7	[238]
D1, D2, D5	Improved sailfish optimization (SFO)	bidirectional gated recurrent unit (BiGRU), and Rat Swarm optimizer for hyperparameter optimization	Acc, N_{sf}, Detection Rate, False Alarm Rate	alpha*CE + (1-alpha) * N_{sf}/N_{tf} alpha not specified	D1 - 10 features, acc = 0.953; D5 - 7 features, acc =1.00; D2 spiral - (4 features, acc = 0.933; D2 meander - 6 features, acc =0. 94.	[239]
D1, D2, D5	Modified GOA	RF	Acc, Detection Rate, False Alarm Rate, N_{sf}	Fitness = sum of each feature importance + weight factor * (1- (N_{sf}/N_{tf}))	D1 - acc = 0.9487; D5 - acc = 1.00; D2 spiral - acc = 0.9291; D2 meander - acc = 0.937	[240]
D1	Chaotic FA	k-NN (k not specified)	Acc, sensitivity, F1 score, Mean Squared Error, FPR, MCC, AUC	0.9*p(Y/X) + 0.1* (1 - N_{sf}/N_{tf}) where p(Y/X) is the learning acc of the evaluator	Logistic FA - acc = 0.90	[241]
D1	Parametric t-test and nonparametric Wilcoxon sum rank test are applied for feature importance in BA, GWO	k-NN (k= 5)	Acc, time, FS stability	0.8*Acc + 0.2* (N_{sf} / (N_{tf} - N_{sf}))	Acc = 0.928 for BA-t and GWO-t	[242]
D1	GOA	SVM, GOA also is used for hyper-parameter optimization of SVM	Acc, time, N_{sf}	Acc of the classifier	best (acc) = 100%, 6 features	[221]
D1, D5	A chaotic bacterial foraging optimization (BFO) with Gauss mutation (CBFO)	An enhanced fuzzy k-NN	Acc, sensitivity, specificity, AUC	Fitness = (sum (avg (test error)))/k; k=10	D1 - acc = 0.9697; D5 – acc = 0.8368	[243]
D1, D2, D5	Optimized Cuttlefish Algorithm (CFA)	Logistic Regression	Acc, N_{sf}, time	Fitness = -1*(1/m) * (log(h)*Y + log (1-h) * (1-Y)); m = number of examples, h is a	D1 - acc = 0.92194, 14 features; D5 - acc = 0.8348, 17 features; D2 meander - acc =	[244]

				hypothesis function, Y = output values	0.8712, 7 features; D2 spiral - acc = 0.8846, 8 features	
D1	A hybrid GA, and GWO	Kernel Extreme learning Machine (KELM)	Acc, sensitivity, specificity, precision, geometric mean, F1 score, N_{sf}	Fitness = 0.99 * acc + 0.01 * ((N_{tf} – N_{sf}) / N_{tf})	Acc = 0.9745	[245]
D1	A PSO improvement	Extreme learning machine (ELM)	Acc, sensitivity, precision, F1 score, N_{sf}	Fitness = acc	Acc = 88.72, N_{sf}=12	[246]
D1	PSO	Fuzzy k-NN	Acc, sensitivity, specificity, AUC, N_{sf}	Fitness = (sum (testing acc))/k, k =5	acc = 0.9747, N_{sf} = 11.9	[247]
D1	PSO	Nonlinear Least Squares Twin SVM (LSTSVM) with Gaussian kernel, PSO - hyperparameter optimization	Acc, sensitivity, specificity	Fitness = max(sensitivity)	Acc = 0.9795	[248]
D1	B-VPL	k-NN (default k)	fitness, N_{sf}	Fitness = 0.99*CE + 0.01 * N_{sf}/N_{tf}	Min fitness =-0.00091, avg (N_{sf}) = 2.73 features	[17]
D1	GA	Decision Tree (DT)	Acc, time	Max(acc)	mean (acc) = 0.9796, 9 features	[202]
D1	Multiverse Optimizer (MVO)	SVM, MVO - hyperparameter optimization	Acc, N_{sf}	Max(acc)	mean (acc) = 95.92. N_{sf} = 12.2	[222]
D4	A binary PSO with a multiple inertia weight strategy	k-NN (k=5)	Acc, N_{sf}, time	0.99*CE + 0.01 * N_{sf}/N_{tf}	mean (acc) = 0.896, mean (N_{sf}) = 347.10 features	[249]
D1	Set-based PSO	k-NN(k=1)	Acc and N_{sf}	max (acc)	mean (acc) = 0.9919, mean (N_{sf}) = 12.9	[250]

Resampling methods are statistical techniques used to generate new samples from existing data and provide the opportunity to train and test algorithms with the aim of reducing overfitting and improving models' performance. Table 2.3 summarizes an investigation into the resampling methods from the review papers.

Table 2.3. Resampling methods

Resampling method	*References*
k-fold CV	(k =10 [224], [225], [251], [227], [230], [237], [241], [243], [245], [246], [248], [242], [222], [250]; (k =5 [236], [247]), (k=20 [235]), (k = 10, k = 3 [222])
Leave-One-PersonOut CV	[229]

The results of metaheuristics are usually compared with others to see if there is a significant difference between them. It is common practice to compare the outcomes of different metaheuristic algorithms, such as accuracy or average fitness, using either parametric or nonparametric statistical tests. Table 2.4 contains a list of references that used statistical tests, as well as their names.

Table 2.4. Statistic tests

Statistic test	References
Wilcoxon sum- rank	[224], [227], [235], [236], [241], [243], [242], [222]
Friedman	[228], [234]
Wilcoxon signed-rank	[251], [237]
t-test	[249]
ANOVA F-test	[250]

All the provided tables give the background data in order to generate some statistics about the trend of most used metaheuristics, classification algorithms, performance metrics, the form of fitness function, resampling methods, statistical tests, Parkinson's dataset, and the best results related to predicting Parkinson's.

2.2 A comparative analysis of filter, and wrapper methods

Since filter and wrapper methods have been very popular and extremely helpful in feature selection problems, an approach is proposed that provides a comparative analysis of different feature selection methods in a voice Parkinson dataset (D1) in order to find an optimal subset with relevant features that gives the highest accuracy. The performance of each feature selection method is evaluated through the accuracy of

three popular supervised learning algorithms: k-NN, RF, and radial basis SVM (rbfSVM). Generalized Simulated Annealing (GSA) is used to improve the accuracy of the hyper-parameters of the classifiers. Figure 2.3 mentions all the methods used.

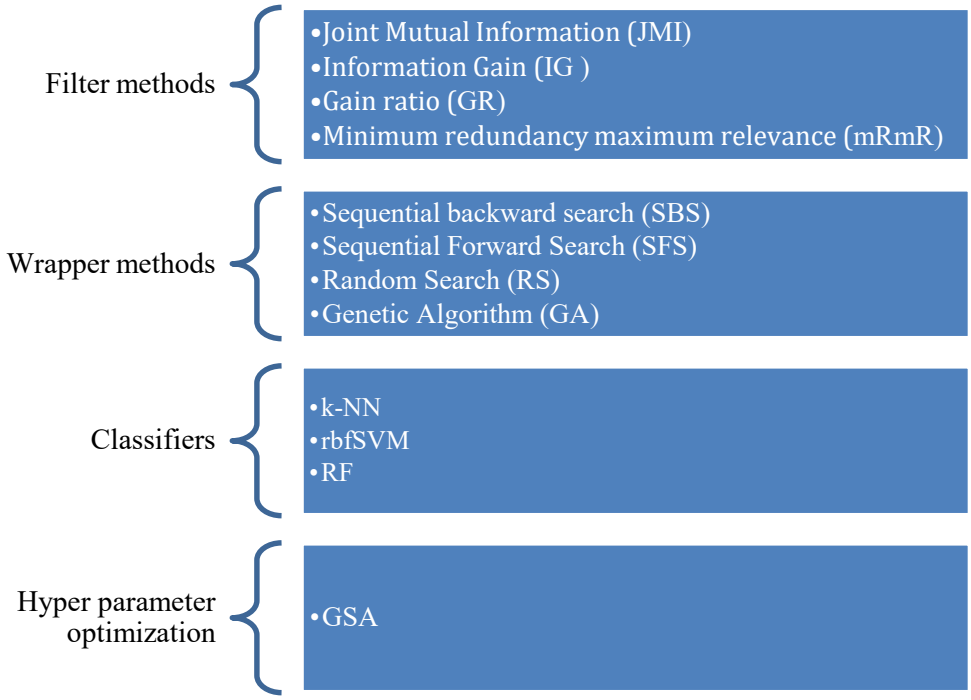


Figure 2.3. The methods used on the comparative analysis

This implementation accomplishes two goals:

- To evaluate and determine the benefits of the filter and wrapper methods in predicting Parkinson and their overall effectiveness.
- The GSA algorithm is employed in hyper-parameter optimization of certain classifiers to highlight the significance of this heuristic approach in improving performance metrics of ML classifiers.

The filter methods do not include a classifier in the process of selecting the optimal subset of features. A typical filter algorithm consists of two steps. The first step involves the ranking of features using statistical measures to determine their significance, while the second step selects the features with the highest rankings to create classification models. Figure 2.4 presents the methodology used for filter methods, which primarily involves generating subsets and then evaluating them using classifier machine learning algorithms.

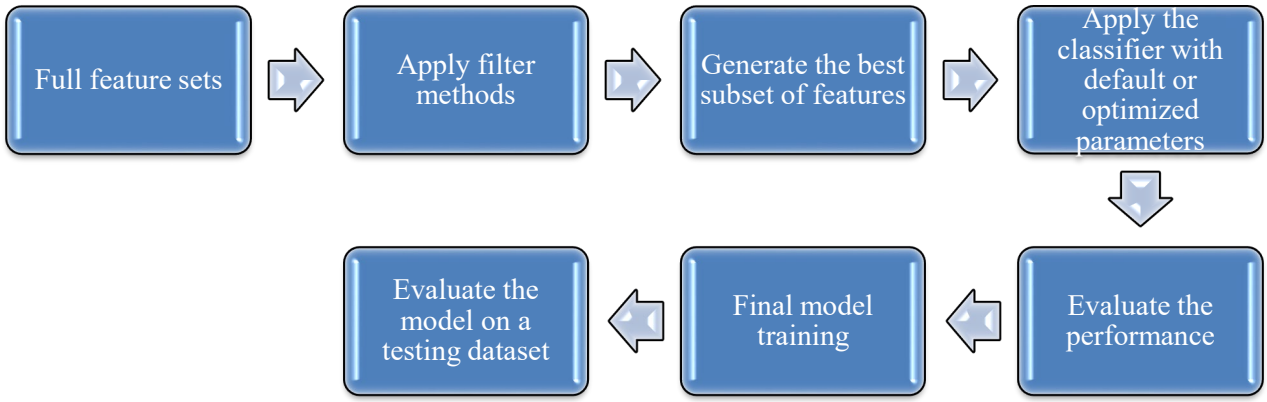


Figure 2.4. The process of applying filter methods in feature selection

IG and GR are entropy-based filters. These algorithms find weights of attributes basing on their correlation with the class attribute. IG it is used in FS for ranking the features according the gain of each variable in the context of the target variable. IG tells how important a given attribute of the feature vectors is. The calculation formula is given in Eq. (2.1) [252]:

$$IG = H(Class) + H(Features) - H(Class, Feature) \tag{2.1}$$

where $H(X)$ is Shannon's Entropy for a feature X and $H(X,Y)$ is a joint Shannon's Entropy for a feature X with a condition to Y.

GR computes a weight for a feature without examining other available features. If features are dependent, this will generally not be reflected in their weights. The equation is given in Eq. (2.2) [252]:

$$GR = \frac{H(Class)+H(Feature)-H(Class,Feature)}{H(Feature)} = \frac{IG}{H(Feature)} \tag{2.2}$$

JMI tries to maximize the mutual information between a subset of selected features and the target variable. The method starts with a feature of a maximal mutual information with the target Y. Then, it greedily adds features in the set X with a maximal value using the following criterion as in Eq. (2.3) [253]:

$$J(X) = \sum_{W \in S} I(X, W; Y) \tag{2.3}$$

where S is the set of already selected features, and $I(X;Y)$ calculates mutual information between each feature and the decision Y.

The mRmR is a widely used filter method for feature selection that uses mutual information to calculate measures of relevance and redundancy between the different features and the class label [254]. The method starts with a feature of a maximal mutual information with the decision Y. Then, it adds features on set X with a maximal value using the following criterion summarized in Eq. (2.4) [253]:

$$J(X) = I(X;Y) - \frac{1}{|S|}\sum_{W \in S} I(X;W) \tag{2.4}$$

where S is the set of already selected features and $I(X;Y)$ calculates mutual information between each feature and the decision.

Meanwhile, wrapper methods use a learning algorithm as a black box for the feature subset selection. The feature search component generates a set of features, and the feature evaluation component uses the classifier to estimate the performance, which is then returned to the feature search component for the next iteration of feature subset selection. The feature set with the highest estimated value is chosen as the final set to learn the classifier. They iteratively add or remove features, aiming to find the combination that leads to the best model performance. Wrapper methods can require more computational time due to the complexity of each classifier's steps. The chosen wrapper methods are: backward, forward search, and random search. Backward search (SBS) starts with an empty set and adds one by one features from the full set, while forward search (SFS) starts with the full dataset and removes the features one by one, generating in the end the final feature subset [47]. The two schemes cannot guarantee finding the optimal subset; therefore, it can be used a random feature generation using the Random Search (RS) method, with the idea of not stucking to some local minima [47]. This method starts searching for the features randomly, and adding or removing features is done also randomly.

Moreover, it is also integrated the GA which is an evolutionary metaheuristic, developed by John Holland and his collaborators in the 1960s and 1970s in [255] based on Charles Darwin’s theory of natural selection. The essence of GA involves the encoding of an optimization function as arrays of bits or character strings to represent chromosomes, the manipulation operations of strings by genetic operators, and the selection according to their fitness, with the aim to find an optimal solution to the problem concerned. It includes the following steps:

```
1.Generate an initial random population
2.Calculate the fitness of the individuals
3.repeat
4.Generate the best individuals using selection operator
5.Generate new individuals using crossover, and mutation operators
6.Evaluate fitness of the new individuals
7.Select individuals for the next generation
8. Until the stopping criteria is fulfilled
```

Two of the most notable advantages of GA are: the ability to deal with complex problems and parallelism [256].

Figure 2.5 presents the methodology followed for the wrapper methods, where they generate a set of candidate feature subsets and uses machine learning algorithms to train and evaluate each of these subsets.

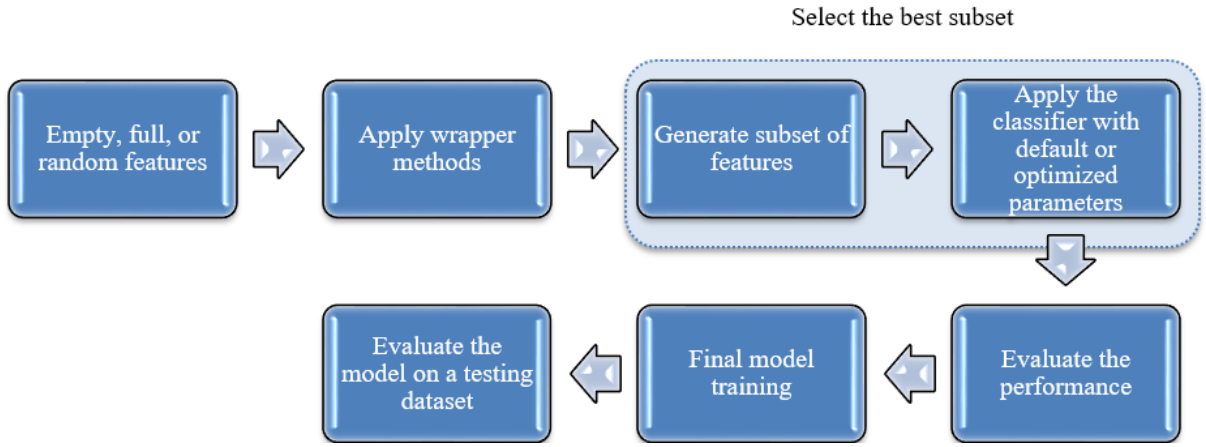


Figure 2.5. The process of applying wrapper methods in feature selection

Simulated Annealing is a physics-based algorithm inspired by the annealing process that happens to particles within a material [257]. Proper regulation of temperature and cooling rate is essential in the annealing process. The objective function is seen as the energy function of a molten metal, and one or more artificial temperatures are applied and subsequently reduced, similar to the annealing method, in order to attain the global minimum. It is a single-solution based heuristic algorithm which utilize single candidate solution and improve this solution by using local search.

Fast Simulated Annealing (FSA) is a semi-local search and consists of occasional long jumps. The cooling schedule of the FSA algorithm is inversely linear in time which is

fast compared with the classical simulated annealing (CSA) which is strictly a local search and requires the cooling schedule to be inversely proportional to the logarithmic function of time [258].

Generalized Simulated Annealing (GSA) heuristic algorithm is proposed as an approach that combines the classical SA ("Gaussian visiting distribution") [257] and fast ("Cauchy - Lorentz visiting distribution") simulated annealing [258] and is considered quicker than both of them [259]. Generalized Simulated Annealing extends SA to handle non-differentiable and non-convex objective functions. GSA was developed to overcome the issue of moving across the entire search space according to Tsallis statistics. It uses the Tsallis-Stariolo form of the Cauchy-Lorentz visiting distribution. For more mathematical details refer to [260].

The implementation of GSA is used for hyper-parameter tuning, in finding the best optimal parameters for each ML classifier in order to increase their accuracy. The mlr package in the R language [261], has two existing functions named *tuneParams()* and *makeTuneControlGenSA()*. The first function optimizes the hyperparameters of the classifier algorithm, while the second function applies GSA.

2.3 A new Binary Volleyball Premier League algorithm in Feature Selection

2.3.1 Mathematical formulation of Volleyball Premier League algorithm

The foundational algorithm of this work is the Volleyball Premier League Algorithm, which Moghdani and Salimifard [262] first created. VPL is considered a human-based algorithm that is inspired by the volleyball league. The composition of players consists of active players, who are those who participate in a game or competition from the initial stages, and passive players, who are substitutes who have the potential to enhance the team's overall performance and are selected by the coach. In VPL, a league represents a population, a team represents a solution, an iteration is a season, a week means the schedule, and the winning team at the end of each season represents the best solution. The VPL Algorithm encompasses 11 distinct steps. The initial phase involves

the initialization process, which starts with the utilization of two matrices named formation and substitution with random values. Their dimensions are the team's number of players and the dataset's number of features. The second phase consists of the setting of the match schedule, which determines the schedule and order of the competitions among the participating teams. In this competitive setting, two teams compete with each other and afterwards, a winner is determined. Probability of winning and power to win the match are calculated when each team plays against each other. Afterward, the losing team is exposed to three strategies: knowledge exchange, repositioning, and substitution. The winning teams should adjust their positions taking into account the best team values, whereas losing teams cannot. During the learning phase, the three first ranked-teams are pointed out, and the lower-ranked teams learn from the teams with better performance. At the end of the season, the best teams are promoted to higher leagues and the worst teams are relegated from the league and will be substituted with new ones. The three concluding stages employed to enhance the efficacy of the proposed solutions are season transfers, promotions, and relegations.

In Figure 2.6 it is presented a flowchart where all the steps of VPL are emphasized, and it introduces the idea of programming the VPL in R language. The algorithm executes 11 steps for each iteration (season), and stores the best solution (team) for each season (run) in the vector Best_Results[run]. The algorithm applies 100 seasons of volleyball leagues to search for the team that provides the lowest fitness. The results of each run are independent from each other.

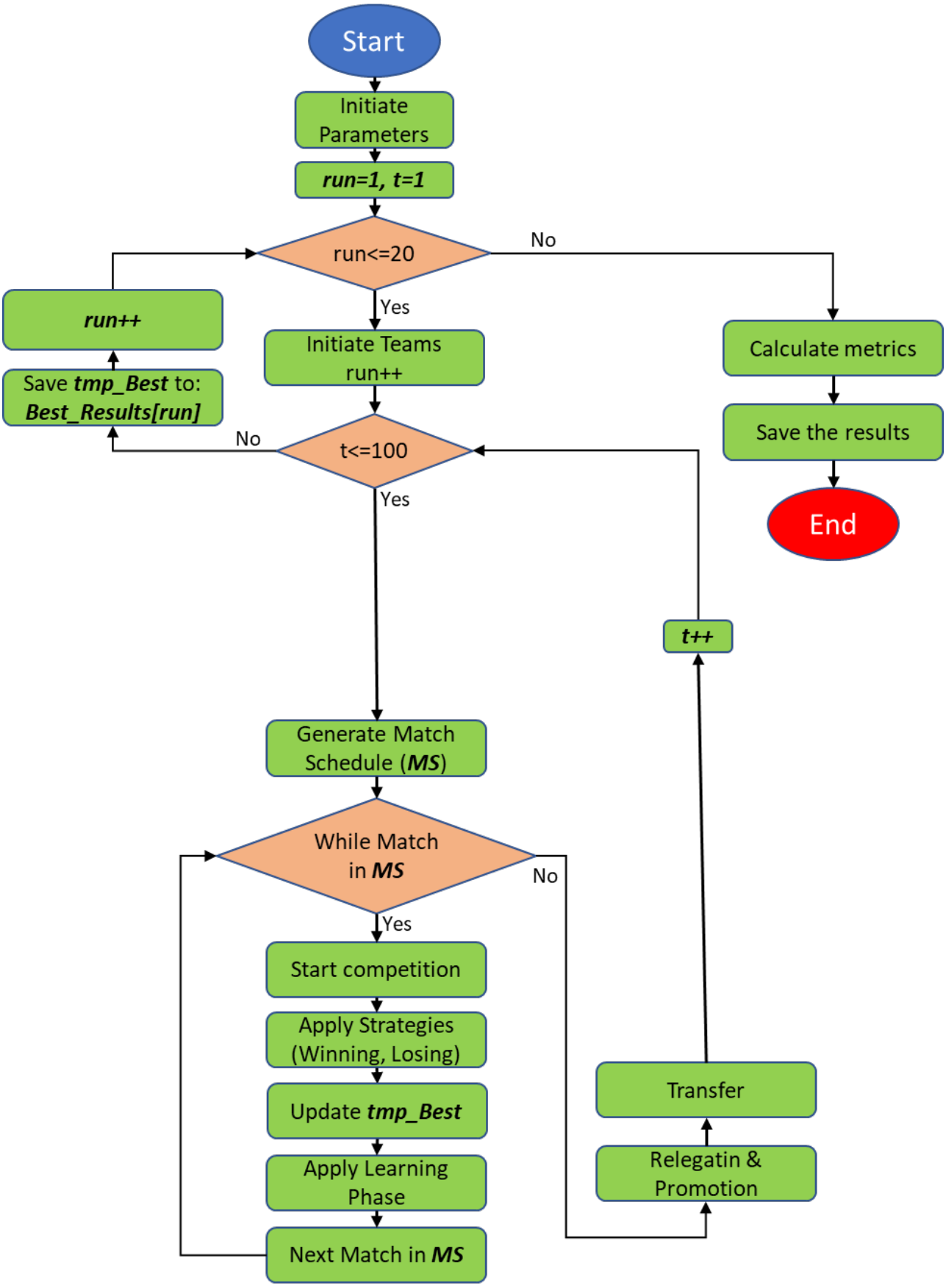


Figure 2.6. The flowchart of volleyball premier league algorithm

VPL algorithm starts with the initialization of the teams which represents the set of initial solutions to the problem. There are two teams: Formation and Substitutes which are created as two matrices F and S with dimensions calculated as (number of teams * number of variables). Initially, X_j^F and X_j^S are the values of Formation and Substitutes of the j_{th} variable, lb_j and ub_j are lower bound and upper bound of the variable j, respectively. $Rand()$ is a uniformly distributed random number between 0 and 1, and the equations are as in Eq. (2.5), and Eq. (2.6):

$$X_j^F = lb_j + Rand * (ub_j - lb_j) \tag{2.5}$$

$$X_j^S = lb_j + Rand * (ub_j - lb_j) \tag{2.6}$$

In the step of match schedule, the single round robin (SRR) method is used to generate the league's schedule, artificially. In an SRR tournament, a team plays against all the other teams. Here it is used the TouRnament package from R software [263] to create the match schedule, differently from the original algorithm. In order to determine the strength of each team in a week and the formation of the team i, it is introduced team power index as in Eq. (2.7):

$$\varphi(i) = \frac{f(X_i^f)}{Z} \tag{2.7}$$

$$Z = \sum_{i=1}^{n} f(X_i^f) \tag{2.8}$$

where $\varphi(i)$ indicates the power index for the team i, $f(X_i^f)$ is the fitness value of team i is computed from its formation and Z indicates the total sum of fitness values in a week (Eq. (2.8)). Since each team is measured according to its weight, the chance of winning a match can be calculated using the fitness value. Let us consider two teams j and k playing in a match, with their formations X_j^f, X_k^f, respectively. So, the power index for the respective are teams are formulated as in Eq. (2.9), and Eq. (2.10):

$$\varphi(j) = \frac{f(X_j^f)}{Z} \tag{2.9}$$

$$\varphi(k) = \frac{f(X_k^f)}{Z} \tag{2.10}$$

Let $p(j,k)$ indicates the probability that team j wins the match (Eq. (2.11)). Then we have:

$$p(j,k) = \frac{\varphi(j)}{\varphi(j) + \varphi(k)} \tag{2.11}$$

Since $p(j,k)$ shows the chance of winning a match, the winner of any match can be determined using a uniformly distributed random number $r \in [0,1]$.

If $$r \leq p(j,k), \tag{2.12}$$

team j wins the match, otherwise the team k is the loser.

After determining the winner of the match, to set a new formation, strategies for winner and loser teams are applied. Knowledge sharing, repositioning, and substitution are the three strategies which can be considered for a losing team, whilst the winning team can use the leading role strategy.

In the phase of knowledge sharing strategy, the coach makes an update on the teams technical and tactical knowledge related to the match as in Eq. (2.13), and Eq. (2.14):

$$X_j^f(t+1) = X_j^f(t) + r_1\gamma^f\left(ub_j - lb_j\right) \tag{2.13}$$

$$X_j^s(t+1) = X_j^s(t) + r_2\gamma^s\left(ub_j - lb_j\right) \tag{2.14}$$

where γ^f, γ^s are coefficients of formation and substitutes, respectively, r_1, r_2 are random numbers uniformly distributed between 0 and 1. Let it be δ_{ks} the rate of knowledge sharing in each team. Number of knowledge sharing in each competition is calculated as in Eq. (2.15):

$$N_{ks} = [J\ \delta_{ks}] \tag{2.15}$$

N_{ks} denotes the number of knowledge sharing positions for each team and J represents the number of total positions in each team.

In the volleyball game, players can be assigned to different positions during a match. This procedure is called as repositioning strategy such that a team changes positions of active players to gain a better performance. Let δ_{rs} indicates the rate of repositioning procedure in each team. The number of repositioning procedures in each competition is defined as in Eq. (2.16):

$$N_{rs} = [J\ \delta_{rs}] \tag{2.16}$$

Let select two positions i and j randomly, and two variables A and B which represent players from formation and substitutes teams. We assign properties of positions i and j to A and B respectively. The formulas are as in Eq. (2.17)-Eq. (2.20):

$$A_f = X_i^f \tag{2.17}$$

$$A_s = X_i^s \tag{2.18}$$

$$B_f = X_j^f \tag{2.19}$$

$$B_s = X_j^s \tag{2.20}$$

Then we assign A and B to the vice verrsa positions j and i, respectively. The formulas are as in Eq. (2.21) – Eq. (2.24):

$$X_i^f = B_f \tag{2.21}$$

$$X_i^s = B_s \tag{2.22}$$

$$X_j^f = A_f \tag{2.23}$$

$$X_j^s = A_s \tag{2.24}$$

During the match, a substitution strategy happens when a formation player is substituted by a player who is currently a member of substitution team. The substitution process is used to search for better solution in the algorithm. Let r indicates a uniformly distributed random number between 0 and 1. The number of substitution in each competition is calculated as in Eq. (2.25):

$$N_s = [r\, J] \tag{2.25}$$

N_s denotes the number of substitution procedures in each team and J represents the number of positions. Within a competition, we determine the loser team, and we select randomly an index of a position called h. All members of the formation set F and substitutions set S are exchanged randomly.

After applying the losing strategies, it is applied the winner startegy. Each winning team detects its position within the search space by combining some features of the history of the best team and some random variations. In the winner strategy, the new position of a team is defined based on itself $X^g(t)$, the best team $X^g(t)^*$, and the inertia weight ω^g, where $g \in \{f, s\}$. Eq. (2.26), and Eq. (2.27) are defined for winning strategy, where ω^f, ω^s denote inertia weights of formation and substitutes, and r_1, r_2 are two uniformly distributed random numbers between 0 and 1, respectively.

$$X^f(t+1) = X^f(t) + r_1\, \omega^f \left(X^f(t)^* - X^f(t)\right) \tag{2.26}$$

$$X^s(t+1) = X^s(t) + r_2\, \omega^s \left(X^s(t)^* - X^s(t)\right) \tag{2.27}$$

Next phase is the learning phase. Let it denote the best solution as $rank1$, the second and the third best solution as $rank2$ and $rank3$. The remaining teams will follow these three teams. We define the following equations to capture the learning behavior of the team:

$$\theta = dbr_1 - b \tag{2.28}$$

$$\vartheta = dr_2 \tag{2.29}$$

where θ, ϑ are coefficient values, d is equal to constant β, r_1, r_2 are two uniformly random numbers between 0 and 1, and b is linearly decreased from β to 0. b is calculated as in Eq. (2.30):

$$b = \beta - \left(t \left(\beta / T\right)\right)$$

(2.30)

where t is the current iteration, and T represents the maximum number of iterations in the algorithm. The value of θ is in the range $-b\beta \; to \; \beta F$ and the next position of the current team can be anywhere within the search space. This is used in the algorithm to make a better exploitation. Using θ, ϑ, Eq. (2.31) shows the main formula which explain the learning phase:

$$X_j^g(t+1)_\emptyset = X_j^g(t)_\emptyset - \theta \left(|\vartheta \, X_j^g(t))_\emptyset - X_j^g(t)|\right) \tag{2.31}$$

In the above equation, there are six sets, generated by the sets $g = \{s, f\}$ and $\emptyset = \{1, 2, 3\}$ where s and f denote the formation and substitutes properties of the team. The index $\emptyset$ may takes values 1 to 3, representing $rank1, rank2, rank3$ teams of the current iteration. $X_j^g(t)$ is the value of position j, and $X_j^g(t+1)_\emptyset$ shows the value of the position j of property g related to the best solutions $\emptyset$. The volleyball coach, coaches the players in a way to get closer to the performance of team $rank1$. For the purpose of mathematical modeling of the learning process, it is assumed that teams $rank1, rank2, rank3$ have better knowledge about the best possible formation and substitutes. The following equations (Eq. (2.32) - Eq. (2.39)) are given to capture the learning phase for formation and substitutes properties:

$$X_j^f(t+1)_1 = X_j^f(t)_1 - \theta \left(|\vartheta \, X_j^f(t))_1 - X_j^f(t)|\right) \tag{2.32}$$

$$X_j^f(t+1)_2 = X_j^f(t)_2 - \theta \left(|\vartheta \, X_j^f(t))_2 - X_j^f(t)|\right) \tag{2.33}$$

$$X_j^f(t+1)_3 = X_j^f(t)_3 - \theta \left(|\vartheta \, X_j^f(t))_3 - X_j^f(t)|\right) \tag{2.34}$$

The three best teams ($rank1, rank2$ and $rank3$) are considered. The new team X is formulated as:

$$X_j^f(t+1) = \frac{X_j^f(t+1)_1 + X_j^f(t+1)_2 + X_j^f(t+1)_3}{3} \tag{2.35}$$

$$X_j^s(t+1)_1 = X_j^s(t)_1 - \theta\ (|\vartheta\ X_j^s(t))_1 - X_j^s(t)|) \quad (2.36)$$

$$X_j^s(t+1)_2 = X_j^s(t)_2 - \theta\ (|\vartheta\ X_j^s(t))_2 - X_j^s(t)|) \quad (2.37)$$

$$X_j^s(t+1)_3 = X_j^f(t)_3 - \theta\ (|\vartheta\ X_j^s(t))_3 - X_j^s(t)|) \quad (2.38)$$

$$X_j^s(t+1) = \frac{X_j^s(t+1)_1 + X_j^s(t+1)_2 + X_j^s(t+1)_3}{3} \quad (2.39)$$

To summarize, the learning phase improve the exploitation process of VPL.

Season transfer is a process in which a player moves from one team to another. Based on this concept, it is defined an operator in the VPL to imitate the process and to help the algorithm to converge toward an optimal solution. In the season transfer time, it is defined a set $H \subset N$, consisting on teams which have been chosen randomly. All positions of each member of set H is selected randomly from the current available teams, $if\ r > 0.5$, where r is a uniformly distributed random number between 0 and 1. Let it be δ_{st} the percentage of teams which participate in season transfers. Therefore, the number of teams which participate in season transfer is formulated as in Eq. (2.40):

$$N_{st} = [N\ \delta_{st}] \quad (2.40)$$

After a season is ended, top teams are moved up to the higher division league, and the worst teams would go down to a lower division for the next season. This process is called promotion and relegation. The number of teams that move depends on the league regulations. Let δ_{pr} indicates the rate of promoted and relegated teams at the end of a season, and the number of teams is calculated as in Eq. (2.41):

$$N_{pr} = \left[N\delta_{pr}\right] \quad (2.41)$$

where N_{pr} denotes the number of teams moved to another league, and N is the total number of teams in the league. N_{pr}teams with the worst values are removed from the league, and N_{pr} new teams are generated and added to the league. To generate each promoted team, the formation and substitutes properties are randomly selected from the formation and substitutes properties of teams in the current premier league, respectively.

2.3.2 The proposed Binary Volleyball Premier League algorithm feature selection- based

The thesis primarily relies on the VPL algorithm to propose and assess its effectiveness and advantages in feature selection. Compared to other metaheuristics, VPL has not been widely adopted by researchers. The VPL was selected based on several factors:

- It had not been utilized before for feature selection.
- It encompasses numerous ways to enhance the solutions till the optimal outcome is attained.

On the contrary, it possesses certain drawbacks:

- A decreased convergence rate
- Possibility of trapping in a local optimum
- A learning phase which affect the performance of VPL
- The algorithm exhibits increased complexity due to the substantial number of phases and frequent calculations of team costs.

A solution enhancing the slow convergence speed and the risk of entrapment in a local optimum for VPL is proposed using the chaotic maps [264]. The chaotic maps can be applied in the initialization of the teams, competition, knowledge sharing strategy, substitution strategy, winner strategy, and learning phase. They are tested on some benchmark functions. It is concluded that by using chaotic maps, the convergence speed is increased. The Iterative map function has given the best results. Moghdani et al. offer an enhanced VPL based on Sine-Cosine (SCA) metaheuristic algorithm [265]. SCA operators for generating the new solutions were used in the learning phase and the performance of VPL was improved.

The actual VPL is used for solving optimization problems where the improved teams of each stage start and generate real values. On the other side, FS is known as a binary problem. By combining the original VPL functionality with several additional operators, a new variation of the algorithm has been designed to optimize solutions in a binary space. When solving the problem of feature selection, the most important step is to create a solution that can represent a subset of features. The VPL teams represent the solution, and it is necessary to binarize these teams to provide the optimal feature

combination for maximum accuracy. Each position in the solution can have binary states: “1” or "0." A value of 0 signifies the absence of feature selection, whereas a value of 1 indicates the selection of a feature. Therefore, as an input for the feature selection, it serves the dataset with all the features, and then the sum of all the 1’s is the provided solution. The FS process, when combined with BVPL, consists of the following steps:

1. The initialization phase is the initial step of any FS technique, and it depends on all the original features present in the dataset. The Formation and Substitute matrices divide the total number of features. The players in "Formation" define the maximum number of possible selected features, which is pre-defined by the user. The Substitute matrix includes the other features that are not in the formation in which the VPL phases, namely repositioning, substitution, and knowledge transfer, can interchange with those of the Formation.

2. The second stage is the subset discovery to select candidate subset of feature for evaluation. The binary VPL is proposed for this phase. The two-step binarization is encoded in each new generated team (solution) during all the phases of VPL. Mathematically they are presented in Chapter 1, respectively Table 1.2, Eq. (1.2), and Eq. (1.3), and are integrated in BVPL algorithm as in Figure 2.7.

3. The third stage involves evaluating the feature subset generated. An assessment measure will evaluate the subset of features generated by the second stage, identifying their performance. In this instance, the subset of features that have been selected is validated on the test set using this cost function.

```
Cost function (Team) {
---- Cost calculation
Calculate Team Accuracy using k-NN
error=1-Accuracy
alpha=0.99
Cost=alpha*error+(1-alpha) *(length (Selected Features) / dimensions)
Return Cost
}
```

4. Predetermining the number of iterations reached is one of the stopping criteria's tasks in the fourth stage. Once the stopping criterion is satisfied, the loop will stop.

5. The best achieved result is stored.

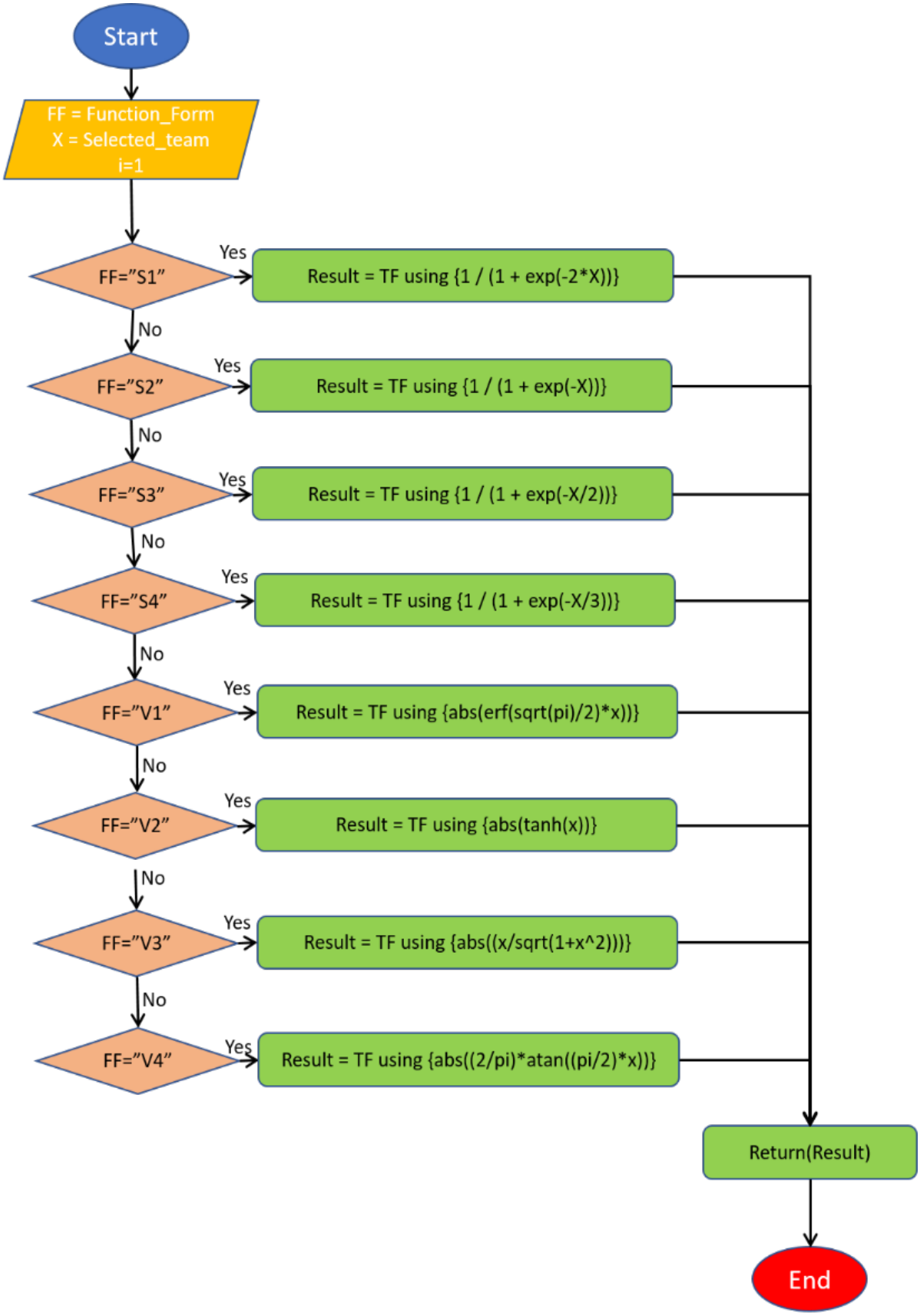


Figure 2.7. The flowchart of two-step binarization

The final pseudocode of BVPL algorithm, combining its steps and the properties specifically for FS is presented in Algorithm 1.

Algorithm 1. Pseudocode of BVPL

```
Input: t = 0, parameters
Output: mean, and standard deviation of fitness, average of selected
features, average accuracy
Initialization
For nruns = 1 to nruns
t =1;
  While t < max_iteration
         Generate a league schedule
         For i=1: (N-1)
     Best team =Select best team according to the cost_function
// ---Two-step binarization is applied each time for converting a
continuous team to a binary one ---//
// --- Cost function is applied each time that the fitness of the
team needs to be calculated -//
           For (each match in schedule table of week i)
           Apply Competition procedure between team A, and B
           Determine loser and winner teams
           Apply different strategies for loser and winner teams
           Update Best team
           Apply learning phase
              End For
     i=i+1
         End For
    Apply promotion and relegation process
    Apply season transfer process
    t = t + 1
    End While
End For
```

2.4 Improving effectivity of Binary Volleyball Premier League algorithm

In this section, there are proposed two solutions with the goal to predict with a higher accuracy Parkinson's using the binary metaheuristic Volleyball Premier League.

2.4.1 Integration of opposition-based learning in Binary Volleyball Premier League algorithm

The main principle of "opposition-based" learning (OBL) is to evaluate simultaneously the fitness values of the current solution and its corresponding opposite solution, then retain the dominant individual to continue with the next iteration, thus effectively strengthening population diversity. OBL has been widely implemented to enhance the

optimization performance of many basic metaheuristics. Many optimization methods have been developed by utilizing the OBL concept in a) the initialization (population-level), b) during evolution, or c) operation level [266].

In this connection, an integration of "opposition-based" learning in the binary Volleyball Premier League algorithm is proposed here. OBL is integrated with the aim of searching for a better solution than that provided by the BVPL, and to explore new other solutions. The mathematical equation for OBL is applied as in Eq. (2.42), where x_{op_sol} is the opposite solution, and x_{act_sol} is the actual solution found better so far.

$$x_{op\_sol} = lower\ boundary + upper\ boundary - x_{act\_sol} \tag{2.42}$$

The idea of using OBL is expressed as in the Figure 2.8.

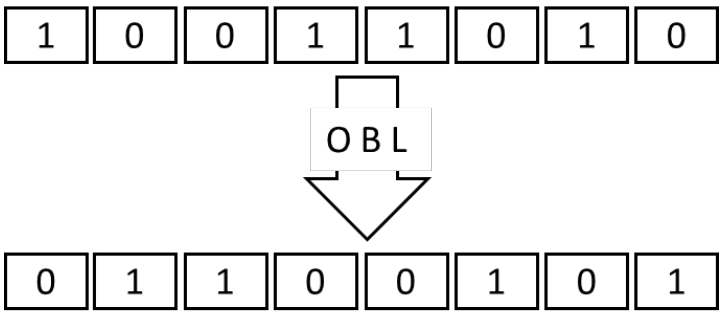


Figure 2.8. Selection of features when using opposition-based learning

Let suppose that the vector of the features evidenced as the best one after applying all the steps of BVPL at the end of iteration is the upper one, where 1 means that the feature is selected, and 0 the feature is not selected. After applying the OBL technique, the vector of features is the same as in the lower part where now the selected features correspond to the 1 value which were not selected in the best solution provided by BVPL.

In this case, an OBL technique is implemented in the final phase of BVPL, after concluding season transfer where besides the best solution (team) found so far, it is calculated also the cost of the opposite-based learning solution (here is stored in the variable named Temp). After comparing them, the solution which provides the lower

cost, will be retained and will be used as the best solution provided so far in the next solution. The implementation of this idea is shown in Figure 2.9.

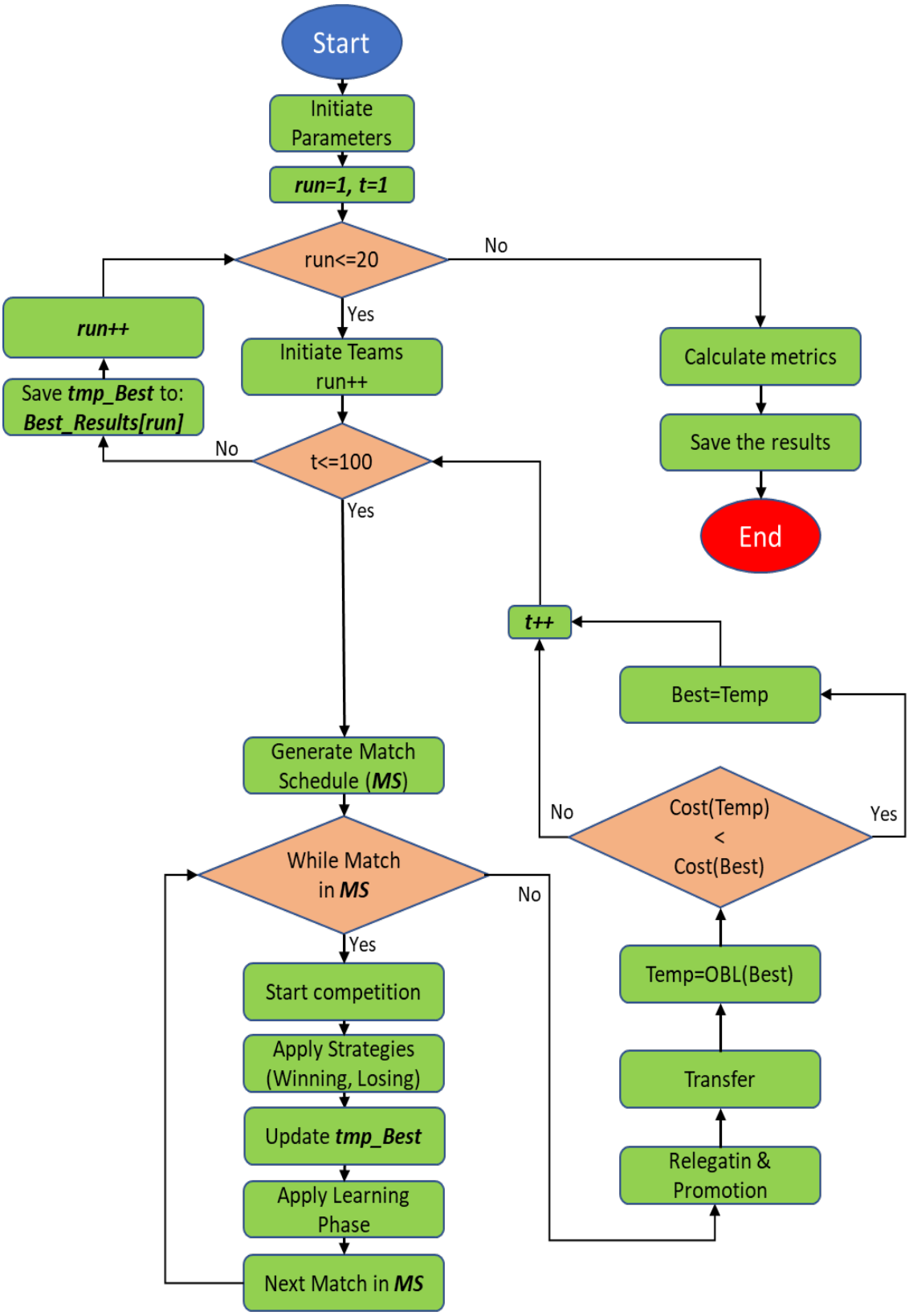


Figure 2.9. The flowchart of opposition-based learning BVPL

Integrating OBL give some advantages:

- If the best optimal solution is chosen from the OBL, it is then incorporated into the subsequent iteration, aiding in the generation of another optimal solution. The influence of the top three ranked teams on the new solution will impact the outcomes of the next iteration, leading to a prediction with higher accuracy.
- Since BVPL is limited to the number of features selected, the OBL_BVPL solution offers a wider selection of features. This improves its exploration abilities, reducing the risk of remaining at the local optimum.

2.4.2 A hybrid Binary Volleyball Premier League and Antlion Optimizer metaheuristic algorithm

A new hybrid metaheuristic optimization algorithm that combines two different metaheuristics—the ALO metaheuristic and the BVPL metaheuristic algorithms—to solve the problem of FS is a new proposal in the thesis. The aim of this integration is to improve the learning phase of the BVPL algorithm since affects the performance of VPL. Hybridization is employed to prevent becoming trapped in a local optimum giving the option to diversify solutions. Unlike OBL, which is utilized after an iteration and improves a team (solution) cost, the utilization of antlion improves all the teams using a probability rate specified by the user.

The ALO algorithm was proposed by Mirjalili [267] and mimics the hunting mechanism of ant lions in nature. In the ALO algorithm, the first positions of antlions and ants are initialized randomly and their fitness functions are calculated. Then, the elite antlion is determined. In each iteration for each ant, one antlion is selected by the roulette wheel operator and its position is updated with the aid of two random walk around the roulette selected antlion and elite. The new positions of ants are evaluated by calculating their fitness functions and comparing with those of antlions. If an ant becomes fitter than its corresponding antlion, its position is considered as a new position for the antlion in the next iteration. Also, the elite will be updated if the best antlion achieved in the current iteration becomes fitter than the elite. These steps are repeated until the end of iterations. The binary ALO proposed here is used according to

the work of [268], named BALO approach 2, adapted with the presented cost_function, and two-step binarization.

ALO has very competitive results in terms of improved exploration, local optima avoidance, exploitation, and convergence [267]. It addresses two problems of BVPL which are the risks to fall in a local optimum, and exploration. These are some reasons for including BALO's solutions in the matrices of teams of BVPL.

One strategy mentioned for hybridization was the integrative approach, where one algorithm is considered a subordinate or embedded part of another [12]. This is an approach applied for this variant of hybridization between BVPL and BALO.

2.4.2.1 The mathematical formulation of Ant Lion Optimizer

Mathematically, in ALO are calculated the following equations.

The radius of ants' random walks hyper-sphere is decreased adaptively, and is calculated by Eq. 2.43, Eq. 2.44, and Eq. 2.45.

$$c^t = \frac{c^t}{I} \tag{2.43}$$

$$d^t = \frac{d^t}{I} \tag{2.44}$$

$$I = 10^{\omega} \frac{t}{T}, \tag{2.45}$$

where c^t is the minimum of all variables at t[th] iteration, d^t is the maximum of all variables at t[th] iteration. I is calculated as in Eq.2.45 where t is the current iteration, T is the maximum number of iterations, and ω is a constant defined based on the current iteration. This constant can adjust the accuracy level of exploitation.

Roulette wheel selection is used for selecting an ant lion randomly. RW1 represents the attraction of an ant by the elite ant lion that is represented by a random walk around the elite continuous-valued ant lion with suitable step size. And it can be represented by stochastic mutation around a selected ant lion with suitable mutation rate around the binary valued elite ant lion in the binary version as given in Eq. 2.46. RW2 represents the attraction by the other ant lions and is performed by applying stochastic mutation around an ant lion in the binary mode that is selected by the roulette wheel selection

method. RW1 and RW2 are binary vectors representing the effect of elite ant lion and a random selected ant lion.

$$x_{out}^{d} = \begin{cases} x_{in}^{d} & if\ rand1 \geq r \\ rand2 & otherwise \end{cases} \tag{2.46}$$

where x_{out}^{d} is the d dimension value for the output vector from mutation, x_{in}^{d} is the input vector to be mutated rand1, rand2 are two random numbers drawn from uniform distribution in the range [0, 1], and r is the mutation rate. It worth mentioning that r is linearly decremented with iteration number ranging from 0.9 to 0 according as shown in Eq. 2.47.

$$r = 0.9 + \frac{-0.9*(i-1)}{IterMax-1} \tag{2.47}$$

where r is the mutation rate at iteration i and, IterMax is the total number of iterations to run the optimization. The used crossover is simple stochastic crossover that switches between the two-input vector with same probability as given in Eq. 2.48.

$$x^{d} = \begin{cases} x_{1}^{d} & if\ (rand) \geq 0.5 \\ x_{2}^{d} & otherwise \end{cases} \tag{2.48}$$

Random walks are all calculated based on the Eq. 2.49.

$$X(t) = [0, cumsum(2W(t_1) - 1);\ cumsum(2W(t_2) - 1); \ldots;\ cumsum(2W(t_T) - 1)] \tag{2.49}$$

where cumsum calculates the cumulative sum, T is the maximum number of iterations, t shows the step of random walk, and W(t) is a stochastic function defined in Eq. 2.50.

$$W(t) = \begin{cases} 1 & if\ (rand) > 0.5 \\ 0 & otherwise \end{cases} \tag{2.50}$$

where t shows the step of random walk and rand is a random number generated with uniform distribution in the interval of [0, 1]. In order to keep the random walks inside the search space, they are normalized using a min-max normalization. The pseudocode of the binary ALO is presented as follows:

Algorithm 2. BALO pseudocode [268]

```
Input: N - number of antlions, n - number of ants, max_iterations =
100, nruns=20
Output: Optimal ant lion binary position, best, worst, and standard
deviation, average of selected features, average accuracy
1. Initialize a population of n ants' positions at random ∈ 0, 1
```

```
2. Initialize a population of n ant lions' positions at random ∈ 0, 1
3. Calculate the fitness of all ants and ant lions using the
cost_function
4. Find the fittest ant lion
For nruns = 1 to nruns
t=1
5.   While t < maxiter
         Calculate the radius of the ant's random walk
    For each antj do
           Select an ant lion at random using roulette wheel
selection (ant_lionRW)
Apply random walk around ant_lionRW given the current random walk
radius; called x1
Apply random walk around xelite given the current random walk radius;
called x2.
Apply the two-step binarization in solutions x1, x2 to output RW1, RW2
Perform crossover between RW1, RW2 and set the new position of antj
to the output of crossover.
   End For
             Calculate the fitness of all ants using the cost
function
             Replace an ant lion with its corresponding ant it if
becomes fitter
             Update the elite
    t = t + 1
    End While
End For
6. Produce the elite ant lion xelite and its fitness.
```

2.4.2.2 The new hybrid Binary Volleyball Premier League and Antlion Optimizer

The learning phase of VPL creates an extensive searching range for the algorithm. The main advantage of the VPL algorithm comes from the learning phase, making all teams follow the top three teams. However, the learning phase has the largest effect on the performance of the VPL algorithm, and this phase can lead to the VPL getting stuck in an optimal local solution [265]. In order to improve BVPL, and taking into consideration the advantages of BALO, the learning phase of BVPL is improved using the method of generating the best solutions using BALO algorithm. In the event that the BALO learning phase generates a better solution, the team with better fitness will be updated on the team table. So BALO improves the searching area of BVPL. In this approach, the probability of the fitness function f_i is calculated as in Eq. (2.51). Based

on the value of $Prob_i$ the current team can update its behavior using the BALO operators, or else the traditional process in BVPL.

$$Prob_i = \frac{f_i}{\sum f_i} \tag{2.51}$$

The hybrid metaheuristic BVPL_BALO is presented in a pseudocode form in Algorithm 3 below.

Algorithm 3. The proposed hybrid metaheuristic BVPL_BALO

```
Input: iteration = 0, parameters, max_iter, nruns
Output: average fitness, standard deviation of fitness,
average of selected features, average accuracy
1. Initialization
2. Create an Occurrence List (first element = all 0 team with Cost 1)
For nruns = 1 to nruns
t =1; maxiter = 100
  While t < maxiter
         3. Generate a league schedule
         For i=1: (N-1)
     Best team =Select Best team according to Cost function
// ---Two-step binarization is applied each time for converting a
continuous team to a binary one ---//
// --- Cost function is applied each time that the fitness of the
team needs to be calculated -//
             For (each match in schedule table of week i)
4. Apply Competition procedure between team A, and B
           5. Determine winner and loser teams
           6. Apply different strategies for winner
                       and loser teams
           Update Best team
           7. Calculate the probability (Probi) (2.51)
           If (Probi >rand)
                  8. Apply BALO
                If (Team$fitness>New_team$fitness){
                     Team=New_Team
                End if
           Else
                  9. Apply learning phase BVPL
           End If
             End For
     i=i+1
         End For
    10. Apply Promotion and relegation process
    11. Apply season transfer process
    t = t + 1
    End While
```

```
End For
```

Figure 2.10 illustrates in a flowchart the proposed hybrid metaheuristic BVPL_BALO. In this figure, the hybrid metaheuristic uses the same steps as BVPL until the learning phase. The difference comes in improving the solutions of the learning phase using BALO or BVPL in generating new solutions based on a probability, and an if condition. The metaheuristic BALO is applied when the probability is greater than a generated random number between 0 and 1.

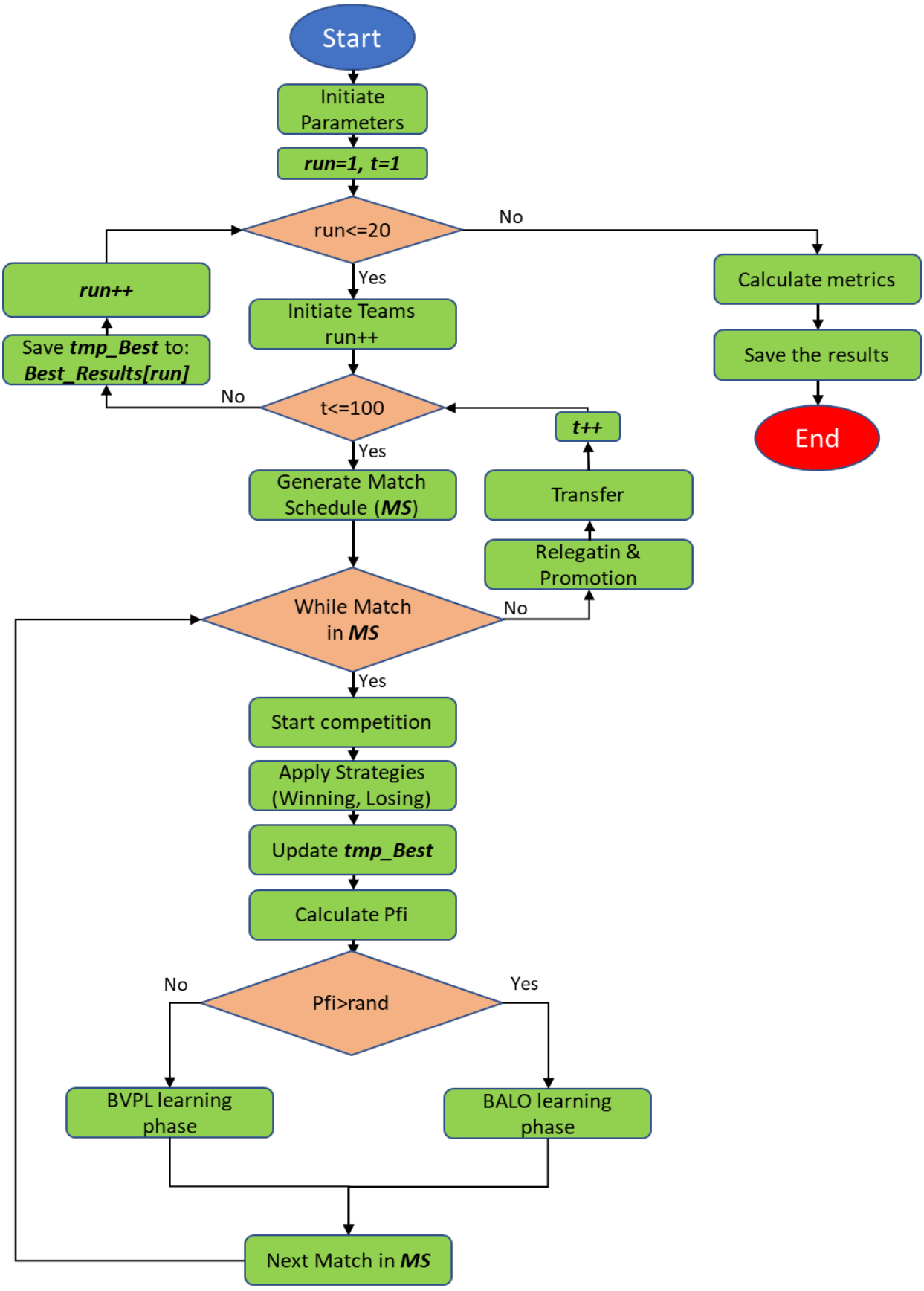


Figure 2.10. Flowchart of BVPL_BALO

2.5 Improving efficiency of the hybrid Binary Volleyball Premier League and Antlion Optimizer metaheuristic algorithm

A drawback which is observed when using VPL is its largest execution time. The complexity of VPL is dependent on the number of populations or teams: active + passive (2n), number of dimensions of the dataset (dim), the number of iterations (T), fitness function f*i*, and is formulated according to Eq. (2.52) [265].

$$O(VPL) = (O(T(2n^2))) + O(n * dim) * O(f_i) \quad (2.52)$$

The equation shows that as the number of populations and dimensions increases, the complexity of VPL also increases. Besides it, when the proposed BVPL is applied in feature selection, the execution time is increased when using the cost function because this calculation is affected by the complexity of k-NN classification algorithm, and the number of times the classification algorithm is executed.

2.5.1 Integrating the occurrence list in the cost function

For solving the drawback of the largest execution time required by the hybrid metaheuristic which normally it is inherited by the BVPL metaheuristic algorithm, a new technique is integrated in the proposed hybrid algorithm in order to improve the execution time of BVPL. This approach stores in a list, named here "occurrence list", the binary positions and the fitness of the generated teams from the previous iterations. This new technique restricts the necessity of recalculating the fitness for the exact team, hence allowing the fitness value to be extracted directly from the "occurrence list". The following pseudocode presents the calculation of the new improved cost function by adding this list.

```
Cost function (Team) {
---- Check if Team is in Occurrence List
If (Team is in Occurrence List)
           Get Cost from Occurrence List
     Else
---- Cost calculation
Calculate Team Accuracy using k-NN
error=1-Accuracy
alpha=0.99
Cost=alpha*error+(1-alpha) *(length (Selected Features) / dimensions)
Add Team and Cost to Occurrence List
```

```
End If
Return Cost
}
```

This pseudocode is integrated in the proposed hybrid BVPL_BALO metaheuristic in order to evaluate its effect on reducing the execution time of BVPL_BALO. The results of this implementation will be tested on the high-dimensional dataset D5 together with other metaheuristics.

2.5.2 A new method combining cosine similarity and metaheuristic method

In this subsection, it is presented a new method which has the aim to improve the efficiency of the proposed hybrid metaheuristic, named shortly CS_BVPL_BALO. The proposed method contains two phases for reducing the features in the feature selection process. In the first phase, an algorithm is proposed that will rank the features according to their importance, considering the similarity of each row of the dataset with the average values for all the features of the dataset. The cosine similarity equation will be calculated for measuring this similarity. Each feature of the dataset will be removed one by one, and a recalculation of the similarity between each row and the other average feature row values will be applied using cosine similarity again. In the end, the average difference between the original average values with all the features of the dataset and the average values when each feature is removed is calculated, and the final score is provided. Once ranked, a metaheuristic receives the selected features as input. In summary, the general steps followed of the proposed method are given in Figure 2.11:

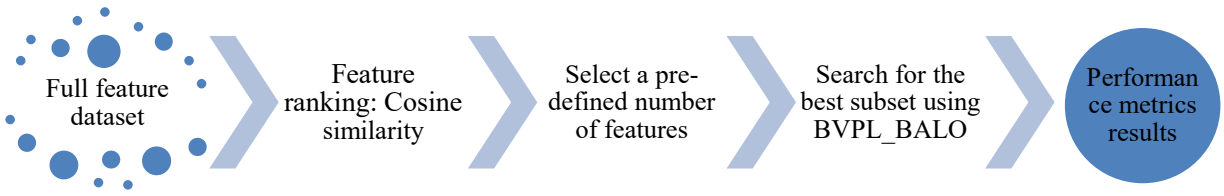


Figure 2.11. The steps of the proposed method CS_BVPL_BALO

Algorithm 4 provides a more detailed description of the method, representing the two phases with a total of 6 sequential steps and the corresponding equations.

Algorithm 4. Proposed method: CS_BVPL_BALO

Phase 1: Features ranking algorithm

Begin

Input: Dataset $P_{rows\ x\ dimensions}$

Step 1. Determine the mean feature value for each column of P. The mean values are stored in a row vector $M_{1\ x\ dimension}$.

Step 2. Calculate the distances between each row of P and row M. The similarity between the dataset and the mean vector, using the cosine similarity is determined by Eq. 2.53. The distances are stored in column vector $H_{rows\ x\ 1}$.

$$similarity = \cos(\theta) = \frac{PxM}{||P||x||M||} \frac{\sum_{i=1}^{n} P_i M_i}{\sqrt{\sum_{i=1}^{n} P_i^2}\ \sqrt{\sum_{i=1}^{n} M_i^2}} \quad (2.53)$$

Step 3. Generate the first dataset, P1, which is the dataset P without the first feature. Calculate the mean of each feature and store in vector M_1. Calculate the distances between each row of P_1 and row M_1 using Eq. 2.53. Store the distances in vector column H_1. Repeat the process for each feature in the original dataset. Finally, the column vectors H_1, H_2, …, $H_{dimension}$ are generated.

Step 4. Calculate the mean difference between vector H of Step 2 and each other vector $H_{dimension}$ as in Eq. 2.54.

For j = 1 to dim:

$$DIFmean[j] = \sum_{i=1}^{dim} |H_i - Hj_i|/dim \quad (2.54)$$

Step 5. Sort the features according to the vector DIFmean values in decreasing order. The feature that causes the largest distance from the original vector is considered the most important.

Output: A ranked list of features which allows to select a reduced dataset with only the most important features.

Phase 2: Apply BVPL_BALO algorithm

Step 6. A pre-defined percent of features of Step 5 are provided as an input to a selected metaheuristic algorithm, in this case is BVPL_BALO.

Final output: average, and standard deviation of fitness, average accuracy, number of selected features, and execution time.

End

Some advantages of using this method are:

- The combination of the two phases presents a new approach for reducing the dimensionality of the data combining a ranking features method and a FS metaheuristic algorithm.
- Implementation of the first phase, which is *Feature ranking algorithm* guarantee that a desired percent of important and identifiable features can be extracted from a large list of features, and given as an input to a metaheuristic algorithm.
- It would be easy to integrate other metaheuristic algorithms in place of the BVPL_BALO one.
- High-dimensional datasets can benefit from its reduction in dimensionality, but datasets with fewer features can also benefit from its use.

Regarding disadvantages of using this method, some of them are:

- The process of ranking the features using Phase 1 could be longer in time because calculating the similarity using cosine similarity is affected by the dimensions of the input dataset. However, this process is conducted only once, and the ranking features could be accessed anytime.
- This method does not guarantee that an extracted number of features is always effective for each given dataset.
- The proposed method does not guarantee the preservation of the predictive model's accuracy in all the datasets.
- The number of dimensions in the dataset and the complexity of the metaheuristic algorithm influence the complexity of this method.

2.6 Chapter conclusions

This chapter presents a group of novel metaheuristic algorithms and methods that are employed for the first time to address the feature selection problem with the focus on predicting Parkinson’s. The main goal is to find the best set of features which predicts with a higher accuracy Parkinson's on each dataset. This will be done using metaheuristic optimization algorithms and classification machine learning algorithms to judge the quality of the metaheuristics' solutions. This chapter addresses the goals by completing tasks 1–7.

Initially, an analytical review is performed to comprehend and assess the prevalence of metaheuristic usage on specific public datasets. This review examines the classifiers, performance metrics, resampling methods, statistical tests used to compare the metaheuristics, and the optimal combination that yielded the most favorable outcomes. Furthermore, due to the evident popularity of filter and wrapper methods, four filter and wrapper methods are utilized to evaluate the performance of the subset of features. This evaluation is done using a Generalized Simulated Annealing heuristic algorithm to optimize the parameters of the classifiers. From this comparative analysis, it will be identified the most resultative methods and the effect of Generalized Simulated Annealing in hyperparameter optimization. Next, a highly effective binary Volleyball Premier League algorithm is suggested to be used on feature selection for picking the most significant features in order to predict Parkinson's with a high accuracy and minimal feature size. The comparison with other metaheuristics and performance indicators will confirm its superiority in feature selection. Next, in BVPL, are incorporated two strategies to enhance its effectivity. Firstly, we incorporate an opposition-based learning technique into BVPL to expand its search area, minimize the risk of reaching the local minimum, and enhance the final solution. Secondly, the binary antlion optimizer is used in combination with BVPL to boost the learning phase and improve the exploitation phase of BVPL, proposing a hybrid BVPL_BALO metaheuristic that significantly improves BVPL's effectivity.

When used in feature selection, BVPL integrates fitness into nearly every step of the metaheuristic, resulting in a significant time requirement for its calculation. Thus, two enhancements are suggested to reduce the execution time of BVPL and to improve its efficiency. In BVPL_BALO, a procedure is implemented that generates an occurrence list, storing the team fitness generated in each iteration, thereby reducing the number of fitness calculations required to generate the same solution. The second enhancement involves a method that integrates a feature ranking algorithm, which utilizes cosine similarity to rank the features according to their significance. As a result, the proposed hybrid metaheuristic selects the significant features from the feature ranking algorithm's defined input. Both proposals significantly enhance the execution time of BVPL_BALO, a feature that BVPL inherited.

3. Experimental results and findings

This chapter provide the results of the experiments in order to validate the novel proposed algorithms, methods, and improvements described in each subsection of Chapter 2.

3.1 Statistics of using metaheuristics in feature selection Parkinson-based

This section outlines the summarized information regarding the actual research and trend in using metaheuristics on predicting Parkinson based on machine learning algorithms. Researchers have utilized a variety of approaches within the field of metaheuristics, including improved iterations of the initial metaheuristic algorithms, chaotic algorithms, and combinations of these methods. The primary focus of researchers has been to identify optimal pairs of metaheuristics and supervised machine learning algorithms for predicting individuals with Parkinson. Figure 3.1 illustrate the availability of diverse metaheuristics applied in the domain of Parkinson's.

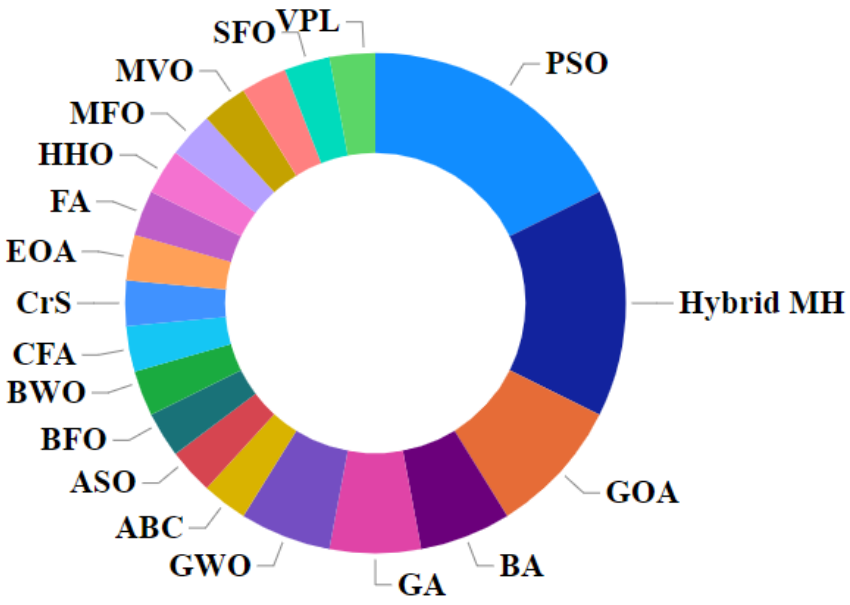


Figure 3.1. Distribution of metaheuristics

PSO was the most frequently used method, appearing six times, followed by five instances of hybrid methods and three instances of GOA. Furthermore, sixteen different metaheuristics have been implemented only once. Typically, researchers evaluate the effectiveness of one metaheuristic by comparing it to another, using different datasets and measures.

Figure. 3.2 illustrates the frequency of utilization of each machine learning algorithm in the selected papers.

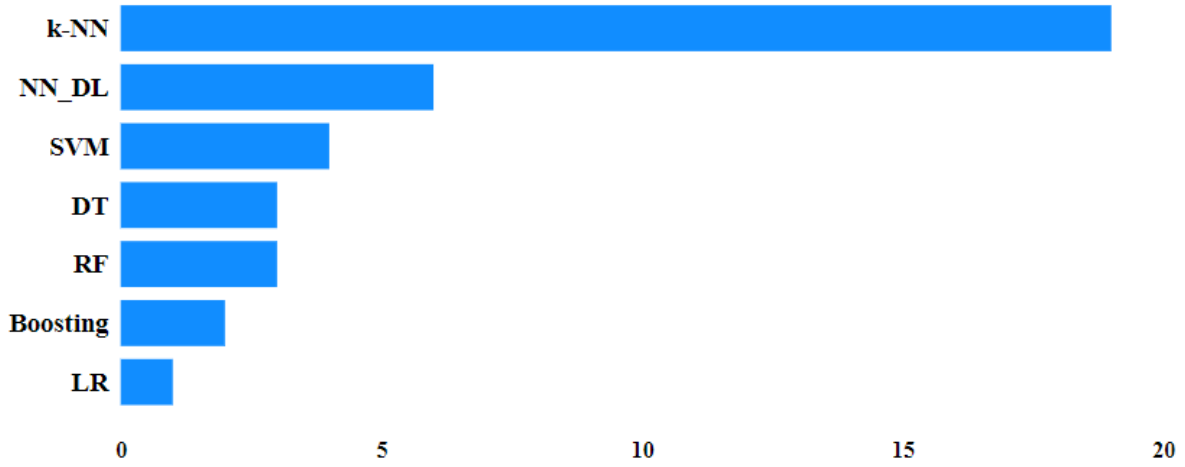


Figure 3.2. The frequency of usage of the supervised learning algorithms

For the wrapper-based approach, nineteen out of thirty-four papers commonly use K-nearest neighbor and its variants, with k = 5 being the most popular. Six publications primarily use neural networks (NN) and deep learning (DL) algorithms. Other frequently used methods are SVM (4 times), RF, and DT in 3 publications. In addition, there are research papers in which metaheuristics, such as GOA and MVO, are used both for evaluating the performance of the selected features and for hyper parameter optimization.

Regarding performance metrics, Figure 3.3 report the usage of average accuracy of the classifier in nearly all of the papers (97.06%). Next in line are the average feature size (76.47%) and computation time (32.35%). Only 23.53% of all publications appear to utilize the F1-score. On the other hand, 14.71% of publications report precision and specificity. The misclassification error rate, negative predictive value, mean squared error, and geometric mean indicate infrequent usage, with a rate of 11.76%.

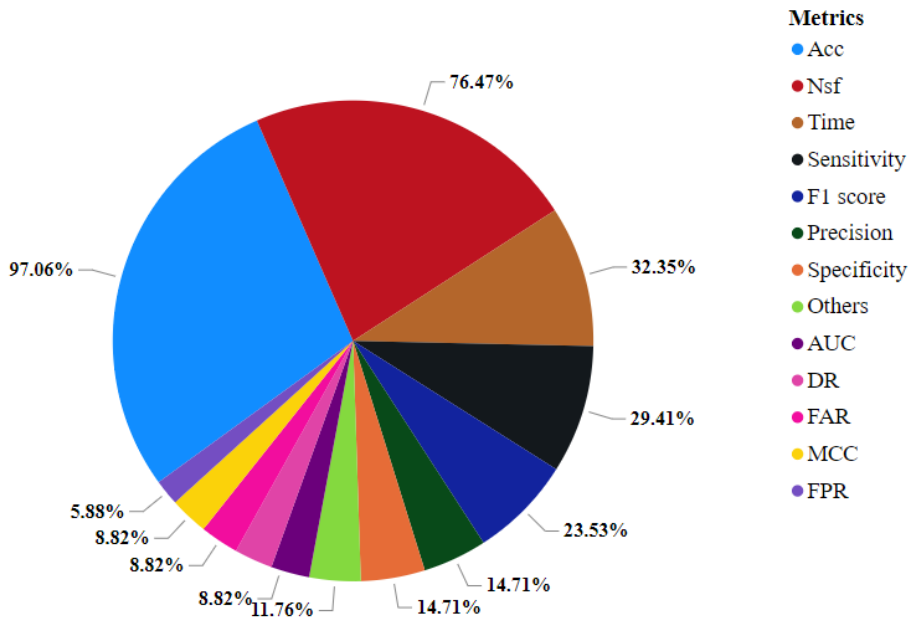


Figure 3.3. The percentage of usage of each metrics

Because FS is an optimization problem, the predominant objective function typically involves computing the classifier's accuracy or error alongside the number of selected features, which occurred on sixteen occasions. This is the same fitness function applied here on the thesis. Additionally, there are alternative formulas that employ diverse weighting parameter values, denoted as alpha, and define the optimization problem as either a minimization or a maximization. The variable alpha typically assumes values within the range of 0 to 1. Some of the studies included in the review fail to provide a clear indication of the value of alpha. Furthermore, some publications state that the classifier's accuracy determines the subset's assessment, but they fail to fully illustrate this for the readers. In addition, various authors have proposed distinct forms of objective functions.

To reduce the problem of overfitting in ML, different resampling methods are proposed, such as cross-validation. K-fold CV, where k = 10, is frequently used (14 times). However, resampling methods are not always considered for the Parkinson datasets, and sometimes they are not explicitly mentioned in the papers. Regarding the utilization of parametric or non-parametric tests, the Wilcoxon sum-rank test is the most frequently employed (8 instances). Subsequently, the Friedman test and Wilcoxon signed-rank test are utilized twice each.

Public datasets, such as voice, speech, and handwriting, were primarily detected in publications. With 23 publications, the D1 dataset is the most frequently used, followed by the D5 dataset with 10 publications, the D7 dataset six times, the D2 spiral and meander five times, and the D3 spiral and meander only once. The use of gait and handwriting datasets is less common.

For some specific occasions, the combination of the metaheuristic and the supervised learning algorithm gave higher Parkinson predictions. Some of the best results for each dataset are listed in continuity. For the D5 dataset, the best accuracy was given by a combination of the Fuzzy Monarch Butterfly Optimization Algorithm + Levy Flight Cuckoo Search Algorithm + Adaptive FA combined with a fuzzy convolution bi-directional long short-term memory deep learning algorithm. The accuracy of acc = 98.77% was taken using a combination of features (Mel frequency cepstral coefficients features + Wavelet features + Concat (baseline, vocal fold, and time frequency features)). Regarding the D1 dataset, almost all the methods produced accuracy greater than 90%. In particular, GOA + SVM produced an accuracy of 100% for six features of this dataset. Besides that, two average accuracies of 99.62% (average features = 2.15) and 99.19% (average features = 12.9) were generated by using an enhanced black widow optimization algorithm and PSO, respectively. When D2 and D3 datasets are applied, an optimized Crow search algorithm combined with k-NN, DT, and RF shows acc = 100%. For the last dataset, D7, a modified GWO-RF, modified GWO-DT, a modified GOA-RF, and improved Sailfish Optimization -bidirectional gated recurrent unit neural network generated an accuracy of 100%.

3.2 Results of comparative analysis of filter, and wrapper methods

The methodology in the Figure 3.4 presents a unified view of the overall stages which are followed to apply each filter, and wrapper method, the three used classifier algorithms, and how are conducted the evaluations of the final subsets. Firstly, filter, wrapper and GA methods will be applied which will generate their best subsets. After the evaluation of the subsets, default and optimized parameters of the learning models will be applied in the same classifiers, to compare the new accuracy with the accuracy of the full features and with the default parameters. GSA optimizer has been used for selecting optimal parameters of the algorithms in order to improve the accuracy. Then

the new parameters will be used to test the new accuracy of each classifier, and to evaluate the difference.

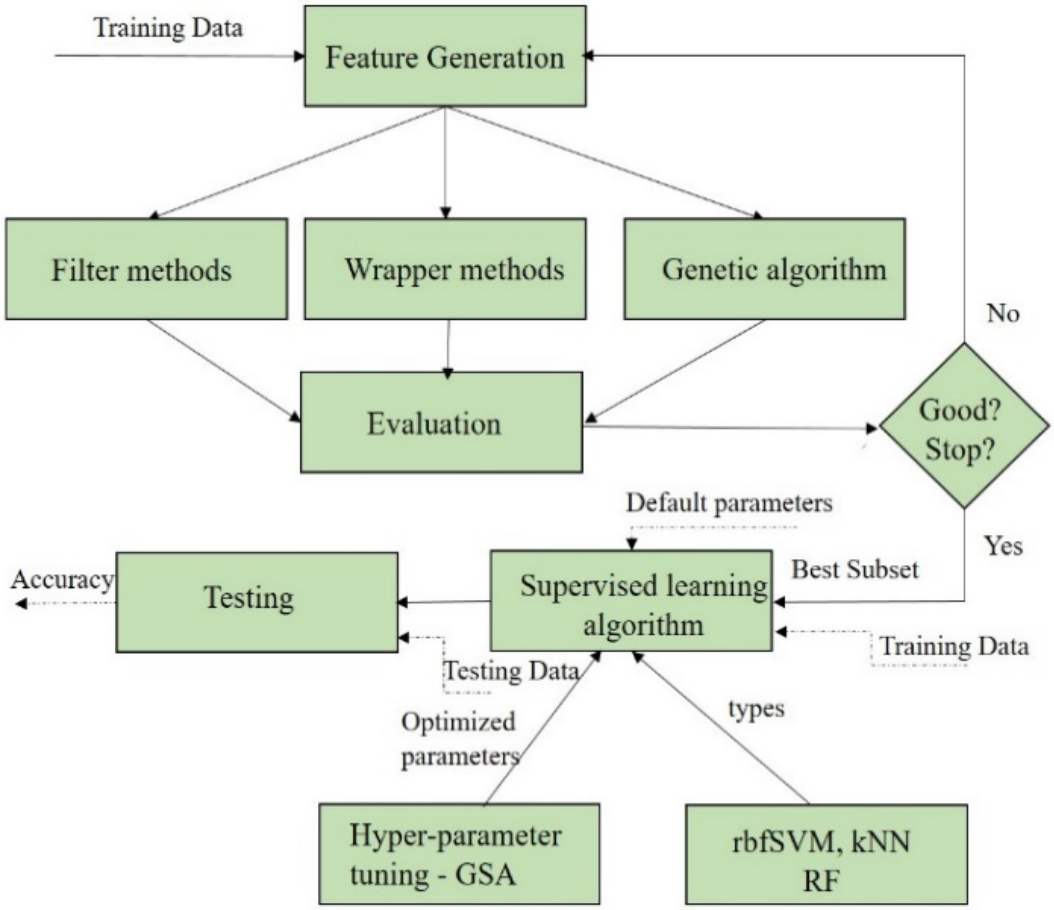


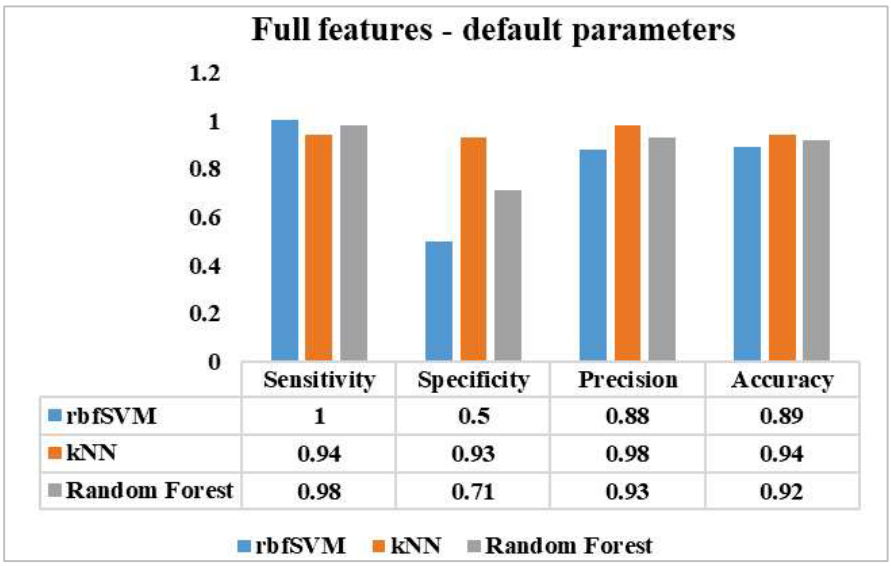
Figure 3.4. The methodology of the comparative analysis

The considered dataset for the analysis is the D1 dataset. All the 22 features of the dataset are numeric. The data were firstly normalized between range 0 and 1 using a min-max normalization. Status variable is transformed in a factor variable with two levels: “yes” when patients have Parkinson and “no” otherwise. The positive class is defined the level yes. There are no missing values in the data. All the methods described above are implemented and tested using mlr package in R [261]. The ratio between training and test dataset is always considered 70:30. Cross-validation is the resampling procedure used to evaluate machine learning models on a limited data sample. It is chosen the 10-fold cross-validation repeated 3 times because of the limited size of the dataset. Hyper-parameter optimization or model selection is the process of choosing a set of hyper-parameters for a ML algorithm. It ensures that the model does

not overfit its data by tuning [269]. It is necessary to specify the search space, the optimization algorithm, an evaluation method, i.e., a resampling strategy and a performance measure.

3.2.1 Results for full features

Initially, were chosen some well-known and largely used classifiers to do a preliminary control of the accuracy that they would classify the dataset. Logistic Regression, Neural networks, and Naïve Bayes were excluded from the analysis because of the low accuracy, specifically 83%, 85%, 68%. Figure 3.5 summarizes the results of the three mentioned classifiers (k-NN, rbfSVM, and RF) in terms of sensitivity, specificity, precision, and accuracy without using GSA. True Positive Rate (Sensitivity) corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points. True Negative Rate (Specificity) corresponds to the proportion of negative data points that are correctly considered as negative, with respect to all negative data points. Precision is the number of correct positive results divided by the number of positive results predicted by the classifier.



	Sensitivity	Specificity	Precision	Accuracy
rbfSVM	1	0.5	0.88	0.89
kNN	0.94	0.93	0.98	0.94
Random Forest	0.98	0.71	0.93	0.92

Figure 3.5. Performance measures of the three classifiers with default parameters.

K-NN has predicted more instances correctly in the Parkinson data, with an accuracy of 94% for the full features dataset, than two other classifiers.

The parameters of each classifier, the search space and the new optimized parameters with the GSA algorithm are showed in Table 3.1.

Table 3.1. Information of the parameters for hyper-parameter tuning

Classifier	Default parameters	Maximum values	Search space	Optimized parameters
K-NN	k = 1; l = 0	1:inf; 0:inf	k=1:4; l=0:1	k=2;l=0.514
Random Forest	ntree=500; mtry-no default values	1:inf; 1:inf	ntree=1:100; mtry = 1:10	mtry=5; ntree=19
rbfSVM	cost=1; gamma = 0	cost=0:inf; gamma=0:inf	cost=1:10; gamma =0:20	cost=4; gamma = 0.129

The new performance measures are given in the Figure 3.6 after applying GSA for the full features dataset.

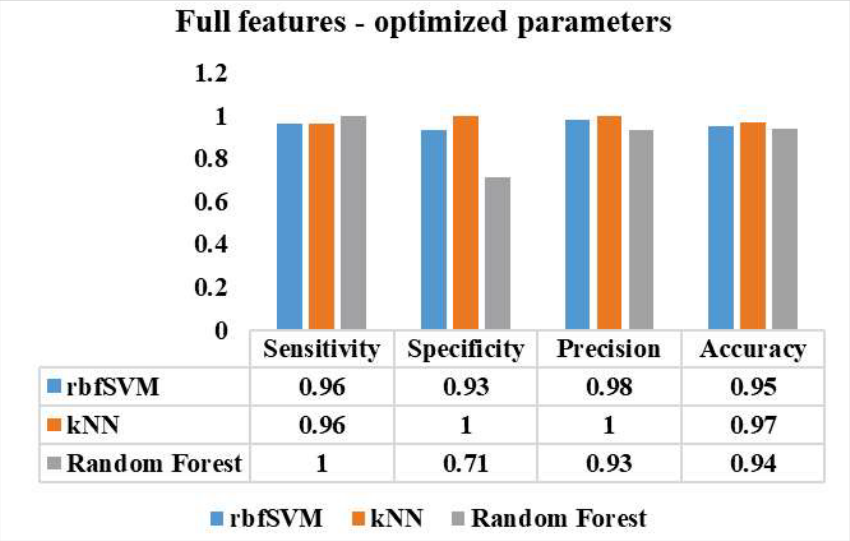


Figure 3.6. Performance measures of the three classifiers with optimized parameters.

In case of k-NN, rbfSVM, and RF the difference in accuracy is increased 3%, 6%, and 2% respectively. The measures when using k-NN are improved totally after using GSA for hyper-parameter tuning.

3.2.2 Filter Methods Results for Default and Optimized Parameters

In the filter methods, it is necessary to select a pre-defined number of features, in order to apply these features to the mentioned classifiers. There are selected the 25% of the top scoring features. The results for each filter method in conjunction with each classifier are summarized in Table 3.2. It can be observed that the accuracy of the subset generated by JMI versus the dataset with full features has had an improvement in case of k-NN and rbfSVM, whereas for RF there was an increase in the percent of the correctly classified observations for the healthy people and a slight increase in the precision, but with an unchanged accuracy. The subset of mRmR gives better accuracy (92%) for RF against the two others. The subset generated by JMI when used k-NN shows the best results.

Table 3.2. Performance measures for the filter methods

	Without using GSA					Using GSA			
k-NN	Sensitivity	Specificity	Precision	Accuracy	k-NN	Sensitivity	Specificity	Precision	Accuracy
FF	0.94	0.93	0.98	0.94	FF	0.96	1	1	0.97
IG	0.82	0.71	0.91	0.8	IG	0.84	0.86	0.96	0.85
GR	0.82	0.79	0.93	0.82	GR	0.92	0.86	0.96	0.91
JMI	0.98	1	1	0.98	JMI	0.98	1	1	0.98
mRmR	0.9	0.93	0.98	0.91	mRmR	0.96	0.79	0.94	0.92
rbfSVM	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**	**rbfSVM**	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**
FF	1	0.5	0.88	0.89	FF	0.96	0.93	0.98	0.95
IG	0.96	0.5	0.88	0.86	IG	0.92	0.71	0.92	0.88
GR	1	0.43	0.86	0.88	GR	0.84	0.86	0.96	0.85
JMI	1	0.64	0.91	0.92	JMI	1	0.93	0.98	0.98
mRmR	1	0.57	0.89	0.91	mRmR	0.98	0.79	0.94	0.94
RF	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**	**RF**	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**
FF	0.98	0.71	0.93	0.92	FF	1	0.71	0.93	0.94
IG	0.96	0.71	0.92	0.91	IG	0.86	0.57	0.88	0.8
GR	0.96	0.71	0.92	0.91	GR	0.9	0.64	0.9	0.85
JMI	0.96	0.79	0.94	0.92	JMI	0.98	0.86	0.96	0.95
mRmR	0.98	0.71	0.93	0.92	mRmR	0.98	0.86	0.96	0.95

The main idea is to compare how does GSA affects the performance of each classifier for each feature selection methods. The search space is the same for each of the methods. After it is used GSA, JMI method gives an improvement in the accuracy compared with the full features dataset for each classifier. It is seen that GSA doesn't

affect and improve the accuracy of JMI filter method but it increases with 5%, 9% and 1 % the accuracies of IG, GR, and mRmR for k-NN algorithm. GSA improves all the performance measures of the subset generated by JMI method (the difference is 6%) for rbfSVM. There is an increase in accuracy with 2%, 3% respectively for IG, mRmR, and a 3% decrease for GR. The subset generated from JMI and mRmR has an increased accuracy when used GSA in case of RF. In totally, among the filter methods, regardless of the classifier, the subset of JMI gives the best accuracy after using optimization with GSA.

3.2.3 Wrapper Methods Results for Default and Optimized Parameters

Regarding the wrapper methods, for forward, and backward search, it is used parameter alpha which shows the minimal required value of improvement difference for a forward/adding step. In this case, the value alpha is equal to 0.02. About RS and GA, the argument chosen was number of iterations, and the computation were executed for 100 iterations/500 iterations respectively. Following is presented Table 3.3 with the performance measures when it is applied or not the GSA algorithm after each subset created by wrapper methods, and GA.

Table 3.3. Performance measures for the wrapper methods

	Without using GSA					Using GSA			
k-NN	Sensitivity	Specificity	Precision	Accuracy	k-NN	Sensitivity	Specificity	Precision	Accuracy
FF	0.94	0.93	0.98	0.94	FF	0.96	1	1	0.97
SFS	0.92	0.86	0.96	0.91	SFS	0.88	0.79	0.94	0.86
SBS	0.94	0.93	0.98	0.94	SBS	0.94	1	1	0.95
RS	0.94	0.93	0.98	0.94	RS	0.94	1	1	0.95
GA	0.94	1	1	0.95	GA	0.94	1	1	0.95
rbfSVM	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**	**rbfSVM**	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**
FF	1	0.5	0.88	0.89	FF	0.96	0.93	0.98	0.95
SFS	0.96	0.57	0.89	0.88	SFS	0.92	0.57	0.89	0.85
SBS	1	0.57	0.89	0.91	SBS	0.98	0.79	0.94	0.94
RS	1	0.5	0.88	0.89	RS	0.96	0.93	0.98	0.95
GA	1	0.57	0.89	0.91	GA	0.98	0.79	0.94	0.94
RF	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**	**RF**	**Sensitivity**	**Specificity**	**Precision**	**Accuracy**
FF	0.98	0.71	0.93	0.92	FF	1	0.71	0.93	0.94
SFS	0.96	0.86	0.96	0.94	SFS	0.94	0.86	0.96	0.92
SBS	0.98	0.71	0.93	0.92	SBS	0.98	0.71	0.93	0.92
RS	0.98	0.79	0.94	0.94	RS	0.98	0.79	0.94	0.94
GA	0.94	0.93	0.98	0.94	GA	0.92	0.93	0.98	0.92

Performing GA for generating a new subset of features came about with an improvement on all the measures when GSA is not used. SBS and GA gives the same % when used rbfSVM as a classifier but there is a difference in the number of features selected, 12 and 10 reciprocally. In regards to Random Forest, SFS, RS and GA gives the same accuracy but GA has also a better precision. The combination GA with k-NN classifier achieves the best accuracy. In the right of Table 3.3 is summarized how the performance of each classifier changes with each subset generated by each wrapper method and GA when is applied hyper-parameter tuning. There is no difference when optimizing the parameters of the k-NN algorithm for the subset of GA comparing with the default parameters of the learner, and there is a slight increase for SBS and RS in accuracy. The only decrease of measures is with SFS filter method (difference in acc = 5 %). The classifier rbfSVM has an increase in accuracy with 3%, 6%, 3% respectively for SBS, RS and GA, with default and optimized parameters, whereas for SFS there is a decrease with 3 %. When comparing wrapper methods for RF, we can state that the accuracies of the SBS and RS has not been changing whereas for GA and SFS has decreased with 2%. In overall, none of the subsets generated by wrapper methods or GA find a subset with a high accuracy than the dataset with the full features, but SBS, RS, GA with k-NN and RS with rbfSVM generates the higher accuracies after the optimization of learning parameters.

From all the experiments, it is concluded that in most of the cases, GSA increased the accuracy of the subsets generated. GSA helps in achieving better performance measures for each of the three classifiers for the full features dataset. From the three of them, k-NN has the highest performance with an accuracy 97% against the others. In regards with the filter methods, for the default parameters of the k-NN learner algorithm, the subset generated by JMI achieves the best results (acc = 98%). The optimal parameters defined by GSA achieves the best results with JMI + k-NN classifier, and also JMI combined with rbfSVM. There are some differences between wrapper methods and GA, when are used default and optimized parameters. Without using GSA, the combination GA and k-NN achieves the best results (acc = 95%), and additionally for rbfSVM and RF, GA gives better results than the full features dataset, respectively, acc = 91 % and acc = 94%. After using GSA, GA, RS and SBS with k-NN and RS with rbfSVM generates the higher accuracies with acc = 95%. When comparing the classifiers, k-NN

and rbfSVM gives an improvement in measures for three from the four methods, in wrapper methods.

3.3 Results from Binary Volleyball Premier League algorithm in Feature Selection

In this paragraph are developed two experiments for testing the compatibility of BVPL in the feature selection problem. In the first one, BVPL is compared with a developed binary PSO metaheuristic algorithm based on one Parkinson's dataset, using the machine learning classifier k-NN as part of the evaluation of the fitness function. In the second experiment, there were enlarged the domain of the metaheuristics whom BVPL is compared, and also 9 other Parkinson datasets were used using again the k-NN classifier.

3.3.1 Experiment 1

In this first experiment, S2, and V4 transfer functions and respectively the standard and complement methods are applied as part of the two-step binarization steps. The fitness function is calculated as in Eq. (1.6). The metaheuristics BVPL and BPSO are used as a search method to investigate the features region as well as to minimize the fitness function. The prediction is based on the D1 dataset. In the methodology presented in the previous paragraph k-NN has achieved an accuracy of 94 % on predicting PD in the same dataset [16]. As a result, the accuracy generated by k-NN it will be included as part of the fitness value for choosing the best team, in other words, the best solution. The general, and the parameters of both metaheuristics are presented in Table 3.4:

Table 3.4. Experiment 1 parameters

Parameter	Value
Iterations	100
Runs	30
Alpha	0.99
No. particles_BPSO	10
ω_{min} – minimum inertia weight	0.4
ω_{max} – maximum inertia weight	0.9
V_{max} – maximum speed	6

c_1, c_2 – cognitive, and social factor	2
δ_{pr}	0.15
δ_{st}	0.5
No. of players	8
No. of teams in a league_BVPL	10

Usually, the subsets generated by the FS method are inputs for classification algorithms of machine learning, and the best subset is chosen. Table 3.5 presents the best minimum (Min_Fit), maximum fitness (Max_Fit), the average number of features (Avg_Feat), average (Avg_Best), and standard deviation (SD_Best) of best solutions achieved in all runs for the four algorithms: BPSO_V (BPSO V-shaped), BPSO_S (B-PSO S-shaped), BVPL_V (BVPL V-shaped), and BVPL_S (BVPL_S S-shaped). The best minimum is achieved from both the BVPL_S and the BVPL_V algorithms, and it is observed that the same optimum value is obtained. Both BVPL variants choose a small number of features compared to BPSO.

Table 3.5. Summary of the metrics in 30 runs

Algorithm	Evaluated metrics				
	Min_Fit	***Max_Fit***	***Avg_Feat***	***Avg_Best***	***SD_Best***
BPSO_V	0.00227	0.005	8.7	0.00395	0.00072
BPSO_S	0.0027	0.005	7.7	0.00351	0.00055
BVPL_V	**0.00091**	**0.00272**	2.73	**0.00135**	**0.00054**
BVPL_S	**0.00091**	**0.00172**	2.73	**0.00182**	**0.00289**

The results from this preliminary experiment shows some promising solutions on using BVPL for generating an optimal subset of features for different input datasets. These results served as indices to extent the calculations in a larger number of datasets, and in other popular metaheuristics.

3.3.2 Experiment 2

In this subsection, a more detailed comparison of BVPL for its efficiency and effectivity compared to other MHOAs its analyzed, and concluded. This experiment includes the 10 datasets related to PD, summarized in Table 3.6. This table describes the long name of the datasets, a short identification of them, their size, and the number of categorical classes in the target variable. The number in the brackets on the third

column shows the final number of columns selected after removing columns or rows with missing values, or ID columns.

Table 3.6. Summarized information about Parkinson datasets

Dataset name	ID	Dimension	Class
Parkinson	D1	195x23 (23)	2
HandPD spiral	D2_S	368x16 (13)	2
HandPD meander	D2_M	368x16 (13)	2
NewHandPD spiral	D3_S	264x16 (13)	2
NewHandPD meander	D3_M	264x16 (13)	2
Early biomarkers of PD based on natural connected speech	D4	130x65 (27)	3
Parkinson's Disease Classification speech-based	D5	756x754 (754)	2
Replicated acoustic features Parkinson	D6	240x48 (46)	2
Parkinson dataset with Multiple Types of Sound Recordings	D7	1040x29 (27)	2
Gait Data Arm Swing	D8	148x58 (55)	2

Moreover, different assessments are employed to determine the most suitable transfer function that aligns with BVPL in each of the mentioned datasets. In Table 3.7 are presented the name of the metaheuristics, references about each algorithm, and the parameters of them. The selected algorithms are: ACO, ABC, ALO, Atom Search Optimization (ASO), Bat Algorithm (BA), DE, Dragon Fly (DF), Firefly Algorithm (FA), Grey Wolf Optimization (GWO), Harris Hawk Optimization (HHO), Moth Flame Optimization (MFO), PSO, Salp Swarm (SSA), Tree Growth Algorithm (TGA), Whale Optimization Algorithm (WOA), Equilibrium Optimizer Algorithm (EOA), GA, Sine-Cosine Algorithm (SCA), Teaching Learning-Based Optimization (TLBO), and Grasshopper Optimization Algorithm (GOA) algorithms. All the metaheuristics codes have been programmed in R language, and adapted for FS from the author. The two-step binarization method has not been applied to GA, DE, and ACO as they provide themselves binary outcomes. As part fitness function for evaluations, it was used again k-NN classifier accuracy with a Euclidean distance metric and k-neighbor = 5 to measure the quality of the subset of solutions.

Table 3.7. Parameters for the experiment 2

Algorithm	Reference	Parameters
General	-	nRuns = 20; maxiter = 100, population = 6; alpha_cost = 0.99, k=5-fold
BVPL	-	fall_rate=0.15, transport_rate = 0.5, β=2, b from β to 0
ACO	[270]	τ =1, eta= 1, α = 1, β= 0.1, ρ = 0.2.
ABC	[271]	Acceleration Coefficient a =1
ALO	[268]	-
ASO	[272]	V_{max} = 6, ε = 0.001, Depth weight α = 50, multiplier weight β = 0.2
BA	[273]	Loudness A = 0.25, pulse rate r = 0.1, Q_{min}=0, Q_{max}=2
DE	[274]	Crossover probability CR = 0.9
DF	[275]	D_{max}= 6
FA	[276]	Light Absorption Coefficient γ = 1, Attraction Coefficient β_0 = 2, Mutation Coefficient α = 0.2, Mutation Coefficient Damping Ratio alpha_damp = 0.98
GWO	[277]	α linearly decreases from 2 to 0, C1, C2, and C3 are random numbers
HHO	[251]	β = 1.5
MFO	[278]	a linearly decreases from -1 to -2
PSO	[279]	Cognitive factor C1 =2, Social factor C2 = 2, W_{max} = 0.9, W_{min} = 0.4, V_{max} = 6
SSA	[280]	C2, C3 = random number]0,1[
TGA	[281]	Number of trees in first group N_1= 3, Number of trees in second group N_2 = 5, Number of trees in fourth group N_4 = 3, Tree reduction rate τ = 0.8, Parameter controls nearest tree λ = 0.5.
WOA	[282]	a decreases linearly from 2 to 0, a_2 linearly decreases from -1 to -2, r_1, r_2, p are random numbers in interval (0,1), b =1
EOA	[283]	Thres = 0.5, V = 1, a_1 =2, a_2=1, GP = 0.5
GA	[284]	Crossover rate CR = 0.8, mutation rate MR = 0.3
SCA	[285]	r_1, decreases linearly from α to 0, α= 2, r_2, r_3, r_4, are random numbers,
TLBO	[286]	-
GOA	[287]	c_{max}=1, c_{min}==0.00004

3.3.2.1 Results from the S-shaped and V-shaped Transfer Function

This subsection presents the optimal outcomes attained by BVPL utilizing eight transfer functions for each dataset. The objective was to identify the most prominent TF for the BVPL based on metrics as: average and standard deviation of fitness, average accuracy, and the average number of selected features. The criteria of selection of the best TF

were according two conditions. First one is the minimum average fitness achieved for each dataset, and secondly when the average fitness is equal (for example D4), the subsequent criterion considered is the maximum average accuracy. Table 3.8 shows the average fitness applied on each dataset, for the eight TF. The bold and italic values highlight the smallest fitness.

Table 3.8. The results of average fitness for each dataset, and transfer function

Dataset	S1	S2	S3	S4	V1	V2	V3	V4
D1	0.06027	0.05929	0.06037	0.06228	0.05749	0.05832	***0.05338***	0.05421
D2_S	0.06508	0.06822	0.07338	0.06686	0.06768	0.06512	***0.06385***	0.06690
D2_M	0.05992	***0.05716***	0.06554	0.06438	0.06054	0.06112	0.05798	0.05856
D3_S	0.15848	0.15530	***0.14256***	0.14607	0.17481	0.17176	0.16378	0.15534
D3_M	0.14306	0.14194	***0.13584***	0.14511	0.15267	0.15205	0.14599	0.14123
D4	0.3491	***0.34154***	0.35054	0.34283	0.35271	0.34762	0.34258	***0.34154***
D5	0.08854	0.08592	***0.08522***	0.08755	0.09008	0.08931	0.08968	0.08599
D6	0.11497	0.12096	0.12025	0.11752	0.11551	0.11371	0.11914	***0.10983***
D7	0.32395	***0.3226***	0.32349	0.32566	0.32869	0.32774	0.32760	0.32313
D8	0.16934	0.17679	0.18291	0.17965	0.16723	0.16270	***0.15493***	0.16745

Table 3.9 represents the minimum standard deviation of the fitness function, where the lower standard deviation is emphasized.

Table 3.9. The standard deviation of fitness for each dataset, and transfer function

Dataset	S1	S2	S3	S4	V1	V2	V3	V4
D1	0.02302	0.02300	0.02496	0.02819	0.02146	0.02226	***0.01874***	0.02274
D2_S	0.01840	0.02083	0.025240	0.02082	0.01864	0.01794	***0.01782***	0.01861
D2_M	0.01598	0.01763	0.02238	0.01637	0.01514	0.01645	0.01515	***0.01388***
D3_S	0.03406	0.03564	0.03281	0.02990	0.03616	***0.02799***	0.03449	0.03104
D3_M	0.02132	0.0225	***0.02114***	0.02431	0.02308	0.02489	0.02989	0.02509
D4	0.03609	0.03427	***0.03182***	0.03448	0.03616	0.03317	0.03419	0.03223
D5	0.01093	0.01111	0.01227	0.01105	0.01286	0.01204	***0.01051***	0.01301
D6	0.02274	0.02440	0.02716	0.03290	***0.01905***	0.03097	0.02417	0.02630
D7	0.01469	0.01530	0.01461	0.01394	0.01299	0.01254	***0.01089***	0.01138
D8	0.03893	***0.03386***	0.03548	0.03908	0.03589	0.03888	0.03445	0.03755

Table 3.10 shows the average accuracy of each dataset. The bold, and italics values highlight the maximum accuracy achieved for each TF.

Table 3.10. The average accuracy for each dataset, and transfer function

Dataset	S1	S2	S3	S4	V1	V2	V3	V4

D1	0.94040	0.94149	0.94068	0.93879	0.94301	0.94215	***0.94723***	0.94669
D2_S	0.93597	0.93311	0.92781	0.93448	0.93344	0.93620	***0.93724***	0.93445
D2_M	0.94174	***0.94433***	0.93570	0.93720	0.94057	0.94037	0.94354	0.94291
D3_S	0.84258	0.84557	***0.85924***	0.85561	0.82582	0.82890	0.83709	0.84620
D3_M	0.85802	0.85950	***0.86557***	0.85587	0.84789	0.84857	0.85506	0.86000
D4	0.64852	***0.65643***	0.64757	0.65530	0.64487	0.65018	0.65526	0.65627
D5	0.91173	0.91482	***0.91593***	0.91374	0.90995	0.91083	0.91060	0.91458
D6	0.88458	0.87919	0.88054	0.88351	0.88417	0.88620	0.88058	***0.89040***
D7	0.67397	***0.67564***	0.67502	0.67289	0.66911	0.67010	0.67034	0.67477
D8	0.82955	0.82282	0.81723	0.82066	0.83184	0.83644	***0.84429***	0.83193

Table 3.11 contains the average size of features for each TF. It is observed that for TF V1 is achieved the most minimum size of features.

Table 3.11. The average number of features for each dataset, and transfer function

Dataset	S1	S2	S3	S4	V1	V2	V3	V4
D1	2.8	3	3.6	3.7	2.35	***2.3***	2.5	3.15
D2_S	***2.05***	2.4	2.3	2.4	2.15	2.35	***2.05***	2.4
D2_M	2.7	2.45	2.25	2.65	***2.05***	2.5	2.5	2.45
D3_S	3.15	2.9	3.85	3.75	***2.85***	***2.85***	3	3.7
D3_M	3	3.4	3.3	2.9	***2.5***	2.55	3	3.15
D4	***2.95***	3.65	4.25	4.1	***2.95***	3.35	3.35	3.25
D5	87.1	119.2	149.75	161.95	***69.75***	77.65	87.8	107
D6	***3.15***	6.1	8.95	9.9	3.8	4.7	4.1	5.95
D7	3.05	3.85	4.55	4.75	***2.9***	2.95	3.2	3
D8	***3.2***	7.45	10.65	11.35	4.05	4.2	4.2	5.75

The successful transfer functions for each dataset are as follows: D1 (V3), D2_S (V3), D2_M (S2), D3_S (S3), D3_M (S3), D4 (S2), D5 (S3), D6 (V4), D7 (S2), and D8 (V3). The selected TF are utilized for the subsequent experiments in the other metaheuristics.

3.3.2.2 Comparison of Binary Volleyball Premier League and metaheuristics

The results from the metrics for the MHOAs are presented in Tables 3.12 – 3.16. The optimal outcomes are shown through the utilization of both italics and bold formatting. In these tables, the short names are referred to the metrics as average fitness (f_{avg}), the standard deviation of the fitness (f_{sd}), average accuracy (acc_{avg}), and the average number of features ($feat_{avg}$).

In reference to D1 (Table 3.12), it can be shown that ACO outperforms BVPL in all metrics, with the exception of the average number of features. BVPL is among the

third-best algorithms after ACO and GA. According to the metrics data presented in Table 3.12 for the D2_S dataset, it is evident that the BVPL algorithm ranks as the second most effective approach, surpassed only by the ACO Algorithm. The ACO Algorithm demonstrates superior performance in terms of average fitness and accuracy. BVPL algorithm exhibits a high level of rivalry in terms of accuracy when compared to TGA, WOA, and GA. There is a slight distinction between BVPL and FA, as well as TLBO, in terms of the number of features.

Table 3.12. The results of the metrics for D1(left) and D2_S (right)

MHOA	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$
BVPL	0.05338	0.01874	0.94723	***2.5***	0.06385	0.01782	0.93724	2.05
ACO	***0.04030***	0.02004	***0.96379***	9.95	***0.05837***	0.01429	***0.94495***	4.75
ABC	0.11009	0.02708	0.89052	2.95	0.12738	0.04765	0.87294	2.05
ALO	0.05949	0.02450	0.94310	6.75	0.08581	0.04358	0.91606	2.8
ASO	0.06754	0.02601	0.93621	9.65	0.07619	0.017780	0.92569	3.15
BA	0.07099	0.03129	0.93362	11.6	0.09661	0.02855	0.90734	5.85
DE	0.05733	0.02612	0.94741	11.6	0.07337	0.01636	0.93119	6.3
DF	0.08042	0.02547	0.92414	11.7	0.08423	0.02352	0.92064	6.8
FA	0.10744	0.03267	0.89310	3.55	0.12738	0.05275	0.87294	***1.9***
GWO	0.05761	0.02221	0.94741	12.2	0.07440	0.01922	0.93028	6.45
HHO	0.06503	0.02657	0.93707	6	0.08095	0.02206	0.92156	3.95
MFO	0.07499	0.02574	0.92586	3.5	0.09121	0.04623	0.90963	2.1
PSO	0.07726	0.02155	0.92759	12.25	0.08802	0.02536	0.91606	5.9
SSA	0.10635	0.03638	0.89483	3.05	0.17589	0.04722	0.82431	2.05
TGA	0.05684	0.02242	0.94828	12.4	0.06544	0.01638	0.93853	5.5
WOA	0.05721	0.02530	0.94483	5.7	0.06641	0.01579	0.93532	2.85
EOA	0.08080	0.03317	0.91983	9.7	0.11483	0.05546	0.88578	6.05
GA	0.04904	0.01659	0.95517	10.25	0.06713	0.01844	0.93670	5.35
SCA	0.07658	0.01947	0.92414	2.7	0.09888	0.03708	0.90180	2.1
TLBO	0.09030	0.02952	0.91035	3.4	0.12556	0.05023	0.87477	***1.9***
GOA	0.09256	0.02409	0.90862	4.6	0.10967	0.04740	0.89174	3

In relation to the D2_M dataset, as illustrated in Table 3.13, it can be established that BVPL gives the greatest average fitness, along with average accuracy and the selected features. The ACO remains highly competitive. According to the data presented in Table 3.13 (D3_S), BVPL Algorithm exhibits substantially better outcomes in terms of average fitness and accuracy compared to ACO. Their results are better than those of the other MHOAs.

Table 3.13. The results of the metrics for D2_M (left) and D3_S (right)

MHOA	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$

BVPL	***0.05716***	0.01763	***0.94433***	***2.45***	***0.14256***	0.03281	***0.85924***	***3.85***
ACO	0.05977	0.02083	0.94358	4.85	0.14590	0.03043	0.85823	6.8
ABC	0.10143	0.03725	0.90138	4.65	0.20743	0.02453	0.79557	5.4
ALO	0.07313	0.01890	0.93211	7.15	0.16586	0.03436	0.83987	8.6
ASO	0.06607	0.02793	0.93532	***2.45***	0.17314	0.03374	0.82848	4
BA	0.08402	0.02626	0.92064	6.55	0.17735	0.02515	0.82595	6.05
DE	0.07990	0.02709	0.92523	7.05	0.16929	0.03239	0.83481	6.9
DF	0.06998	0.01959	0.93440	6.05	0.18040	0.03475	0.82279	5.95
FA	0.11725	0.04133	0.88578	5	0.22355	0.04155	0.77911	5.85
GWO	0.07577	0.02597	0.92890	6.45	0.15396	0.03695	0.85063	7.3
HHO	0.07436	0.02302	0.92982	5.85	0.15467	0.03317	0.84873	5.9
MFO	0.11023	0.05208	0.89312	5.3	0.17004	0.03198	0.83354	6.3
PSO	0.08836	0.03218	0.91606	6.3	0.20972	0.04757	0.79367	6.55
SSA	0.15769	0.04766	0.84312	4.35	0.29933	0.06355	0.70063	5.5
TGA	0.06899	0.01860	0.93486	5.4	0.15542	0.03411	0.84937	7.55
WOA	0.06738	0.01862	0.93716	6.2	0.15609	0.02941	0.84747	6.1
EOA	0.08653	0.04672	0.91697	5.4	0.17861	0.04447	0.82468	4.45
GA	0.06915	0.02361	0.93486	5.6	0.15684	0.03629	0.84684	6.25
SCA	0.10139	0.04792	0.90184	4.45	0.17618	0.03727	0.82722	5.8
TLBO	0.08632	0.02717	0.91697	4.95	0.18345	0.04716	0.81962	5.85
GOA	0.10469	0.04931	0.89817	4.65	0.18884	0.04317	0.81392s	5.55

According to the findings presented in Table 3.14 of D3_M, the SCA method demonstrates superior performance in terms of accuracy and fitness. According to the ranking, ACO is considered the second most favorable alternative, followed by BVPL. The results from the D4 dataset, which are shown in the right of Table 3.14, show that BVPL produce better results than the other MHOAs on all of the criteria that have been looked at. ACO is the second-best one, and the others are far away from this result.

Table 3.14. The results of the metrics for D3_M (left) and D4 (right)

MHOA	**f_{avg}**	**f_{sd}**	**acc_{avg}**	**$feat_{avg}$**	**f_{avg}**	**f_{sd}**	**acc_{avg}**	**$feat_{avg}$**
BVPL	0.13584	0.02114	0.86557	***3.3***	***0.34154***	0.03427	***0.65643***	3.65
ACO	0.12869	0.02737	0.87532	6.45	0.37252	0.04392	0.62821	12
ABC	0.18324	0.03017	0.81962	5.45	0.481	0.05230	0.51795	9.55
ALO	0.16381	0.02369	0.84177	8.85	0.42017	0.03796	0.58333	19.7
ASO	0.16077	0.03347	0.84051	3.45	0.43267	0.04637	0.56410	***2.95***
BA	0.18696	0.04335	0.81646	6.3	0.43515	0.05336	0.56539	12.7
DE	0.16645	0.03761	0.83797	7.25	0.42367	0.05521	0.57821	15.85
DF	0.16327	0.02266	0.84051	6.45	0.43883	0.04639	0.56154	12.35
FA	0.22121	0.05310	0.78101	5.3	0.49485	0.04527	0.50384	9.5
GWO	0.15166	0.03789	0.85317	7.55	0.416	0.04499	0.58590	15.7
HHO	0.14385	0.02511	0.86013	6.45	0.40635	0.04903	0.59359	10.4
MFO	0.15421	0.03190	0.84937	6.1	0.41321	0.03936	0.58590	8.45
PSO	0.19677	0.03463	0.80696	6.8	0.49506	0.05337	0.50513	13.35

SSA	0.25526	0.06306	0.74494	4.7	0.46537	0.05410	0.53205	10.15
TGA	0.14953	0.02683	0.85443	6.5	0.40173	0.03892	0.6	14.9
WOA	0.14573	0.02331	0.85823	6.45	0.39673	0.03614	0.60513	15.1
EOA	0.16766	0.02287	0.83544	5.2	0.40215	0.04229	0.59744	10.9
GA	0.14707	0.02602	0.85633	5.8	0.43825	0.04543	0.56154	10.85
SCA	***0.11151***	*0.04781*	***0.89220***	5.25	0.40267	0.04265	0.59744	11.4
TLBO	0.18228	0.04361	0.82025	5.2	0.43529	0.03697	0.56410	9.75
GOA	0.17113	0.03209	0.83165	5.35	0.43435	0.04199	0.56410	7.3

The results from the seventh dataset, D5 (Table 3.15), provide further confirmation that BVPL outperforms the other MHOAs. The competition between ACO, BA, and WOA is evident across various indicators. In the D6 dataset, as presented in the right of Table 3.15, it can be observed that BVPL presents greater performance in terms of average fitness. However, ACO demonstrates higher average accuracy. ALO reveals strong concurrence with ACO in terms of metrics.

Table 3.15. The results of the metrics for D5 (left) and D6 (right)

MHOA	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$
BVPL	***0.08522***	0.01227	***0.91593***	149.8	***0.10983***	0.02630	0.89040	5.95
ACO	0.09819	0.01492	0.90553	351.7	0.11227	0.01997	***0.89097***	19.7
ABC	0.12731	0.01741	0.87589	331.6	0.16207	0.02344	0.83819	8.2
ALO	0.10986	0.01564	0.89712	605.4	0.11520	0.01930	0.8875	16.9
ASO	0.10473	0.01292	0.89602	***134.6***	0.15249	0.02125	0.84792	8.7
BA	0.09687	0.01310	0.90708	367.7	0.15248	0.02947	0.85069	21
DE	0.10996	0.01631	0.89513	462.3	0.13757	0.02791	0.86667	25.1
DF	0.11513	0.01214	0.88872	373.3	0.17192	0.02570	0.83125	21.9
FA	0.12204	0.01559	0.88120	332.9	0.16833	0.02649	0.83194	8.8
GWO	0.09813	0.01367	0.90774	511.9	0.13015	0.02325	0.87431	25.7
HHO	0.10991	0.01650	0.89381	359.9	0.13889	0.02635	0.86181	9.4
MFO	0.10492	0.01664	0.89845	330	0.12981	0.01735	0.87083	8.7
PSO	0.12457	0.01712	0.87920	374.7	0.16834	0.02626	0.83472	21.2
SSA	0.11879	0.01712	0.88252	336.6	0.15796	0.02849	0.84306	8.2
TGA	0.10641	0.01481	0.89867	459.3	0.13511	0.01844	0.86944	26.4
WOA	0.10327	0.01346	0.90155	436.9	0.13913	0.02421	0.86042	4.25
EOA	0.10829	0.01556	0.89513	295.1	0.12934	0.01938	0.87083	16.45
GA	0.10979	0.01378	0.89381	350.5	0.13516	0.02037	0.86806	20.4
SCA	0.10739	0.01431	0.89624	351.3	0.12829	0.02698	0.87153	***3.4***
TLBO	0.11524	0.01484	0.88805	331.9	0.14520	0.02107	0.85486	6.8
GOA	0.11153	0.01648	0.89071	251.2	0.15037	0.02646	0.85070	11.5

Table 3.16 represents the results for the D7, and D8 dataset. It can be observed that BVPL indicates weak efficacy, while ACO yields superior outcomes in terms of average fitness and accuracy. The ALO, GWO, HHO, TGA, and WOA exhibit significant competition with the ACO. In the last dataset (Table 3.16), it can be observed that BVPL implies better performance across all measures, with the exception of the number of features, where SCA displays the most positive outcomes. For the same dataset, ACO and ALO indicate a considerable level of similarity.

Table 3.16. The results of the metrics for D7 (left) and D8 (right)

MHOA	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$	f_{avg}	f_{sd}	acc_{avg}	$feat_{avg}$
BVPL	0.32260	0.01530	0.67564	***3.85***	***0.15493***	0.03445	***0.84429***	4.2
ACO	***0.29192***	0.01316	***0.71026***	13.5	0.16540	0.04240	0.8375	24.6
ABC	0.33930	0.01734	0.66122	10	0.24681	0.05044	0.75227	8.1
ALO	0.30902	0.01684	0.69615	21.35	0.16177	0.05892	0.83977	17.1
ASO	0.31488	0.01947	0.68478	7.3	0.22156	0.05046	0.77727	5.7
BA	0.31183	0.02068	0.69022	13.4	0.21408	0.05738	0.78864	26.1
DE	0.31067	0.01824	0.69311	17.8	0.20351	0.05208	0.8	29.8
DF	0.31903	0.01359	0.68301	13.6	0.25707	0.05711	0.74546	27.4
FA	0.34018	0.01799	0.66010	9.6	0.24349	0.05011	0.75568	8.7
GWO	0.30059	0.01698	0.70272	16.4	0.19273	0.04837	0.81136	32.3
HHO	0.30341	0.01564	0.69792	11.3	0.21054	0.05312	0.78864	6.95
MFO	0.31164	0.01737	0.68958	11.3	0.17367	0.04461	0.82614	8.4
PSO	0.32716	0.02074	0.67468	13.3	0.25237	0.05098	0.75	26.3
SSA	0.33484	0.02285	0.66555	9.9	0.24825	0.05944	0.75114	8.3
TGA	0.30115	0.00596	0.70401	21.1	0.21772	0.01143	0.78523	27.5
WOA	0.29696	0.01591	0.70657	16.8	0.18185	0.04795	0.81705	3.9
EOA	0.31606	0.01780	0.68510	9.7	0.18704	0.04309	0.8125	21.4
GA	0.31019	0.01461	0.69183	13.3	0.20466	0.04393	0.79773	23.8
SCA	0.31166	0.01995	0.68942	12.2	0.18071	0.04426	0.81818	***3.35***
TLBO	0.31445	0.01843	0.68638	10.3	0.22857	0.03077	0.77046	7.1
GOA	0.32323	0.01769	0.67772	10.9	0.22477	0.04204	0.775	10.9

In general, it can be observed that BVPL produces a smaller number of features compared to the other methods, particularly in the cases of D1, D2_M, D3_S, D3_M, and D7. It is important to note that BVPL has a predetermined minimum number of features, with the maximum being half of the total number of features in each dataset.

3.3.2.3 Convergences curves and statistical difference

The convergence curves can visually illustrate the variations in the performance of all the MHOAs across different criteria. Figures 3.7 and 3.8 illustrate the convergence curves that correspond to the average fitness observed throughout each iteration. Each graph shown represents a distinct dataset. It can be observed that among the three datasets, namely D2_M, D4, and D5, the BVPL algorithm exhibits a quicker convergence speed compared to the other metaheuristics. Moreover, in the cases of D2_S, D3_M, D3_S, D6, and D8, the level of competitiveness of BVPL is notably high. Additionally, it is observed that BVPL in D8, D6, and D3_S exhibit a faster rate of convergence compared to the other MHAs, after the 75th iteration. SSA reveals a major shift in convergence across all datasets, surpassing the other algorithms in terms of fitness values. The ACO algorithm exhibits a notable convergence speed when searching for the global optimum in datasets D1, D2_S, and D7.

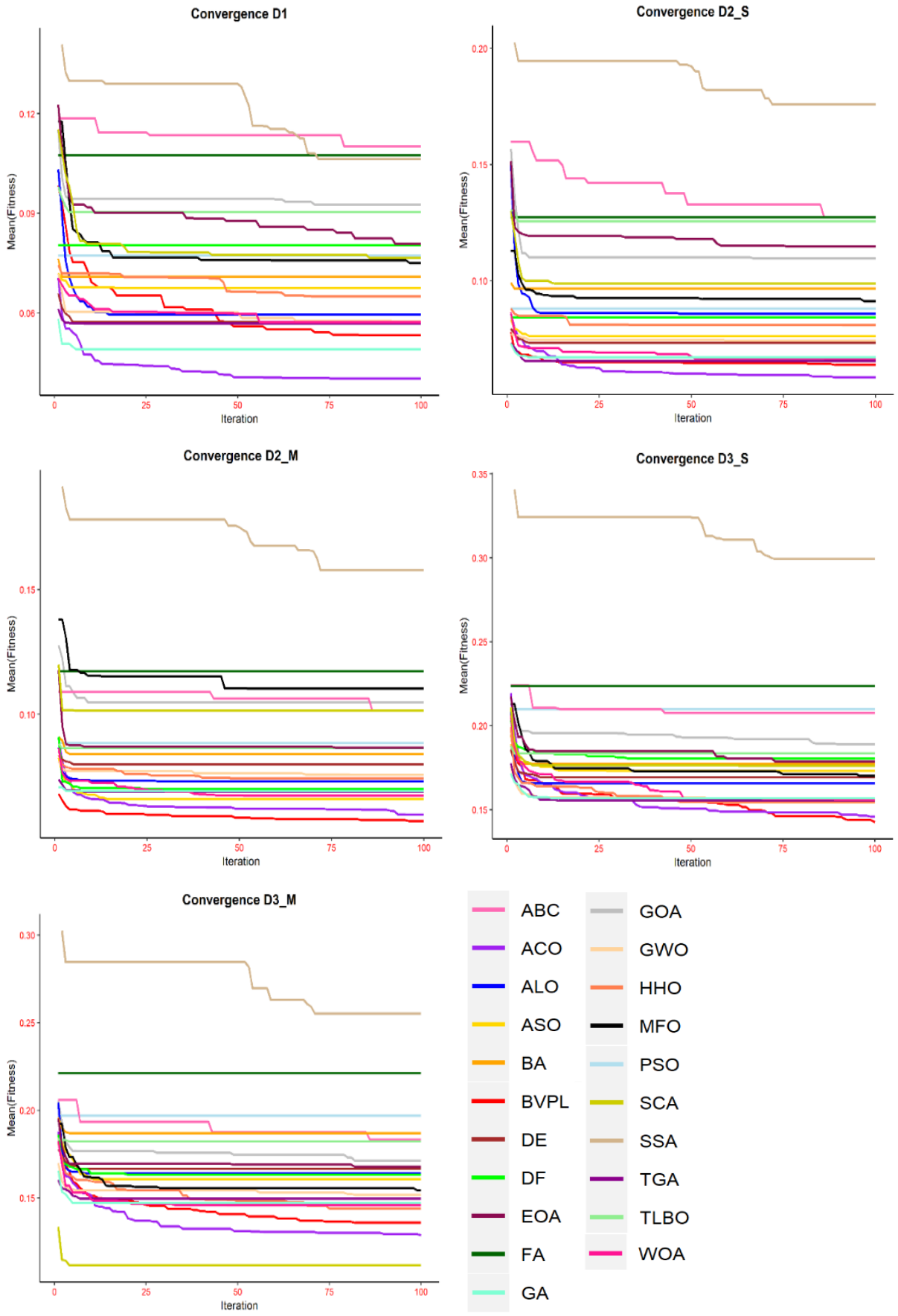


Figure 3.7. The convergence curves of BVPL versus the other MHOAs for the first five datasets

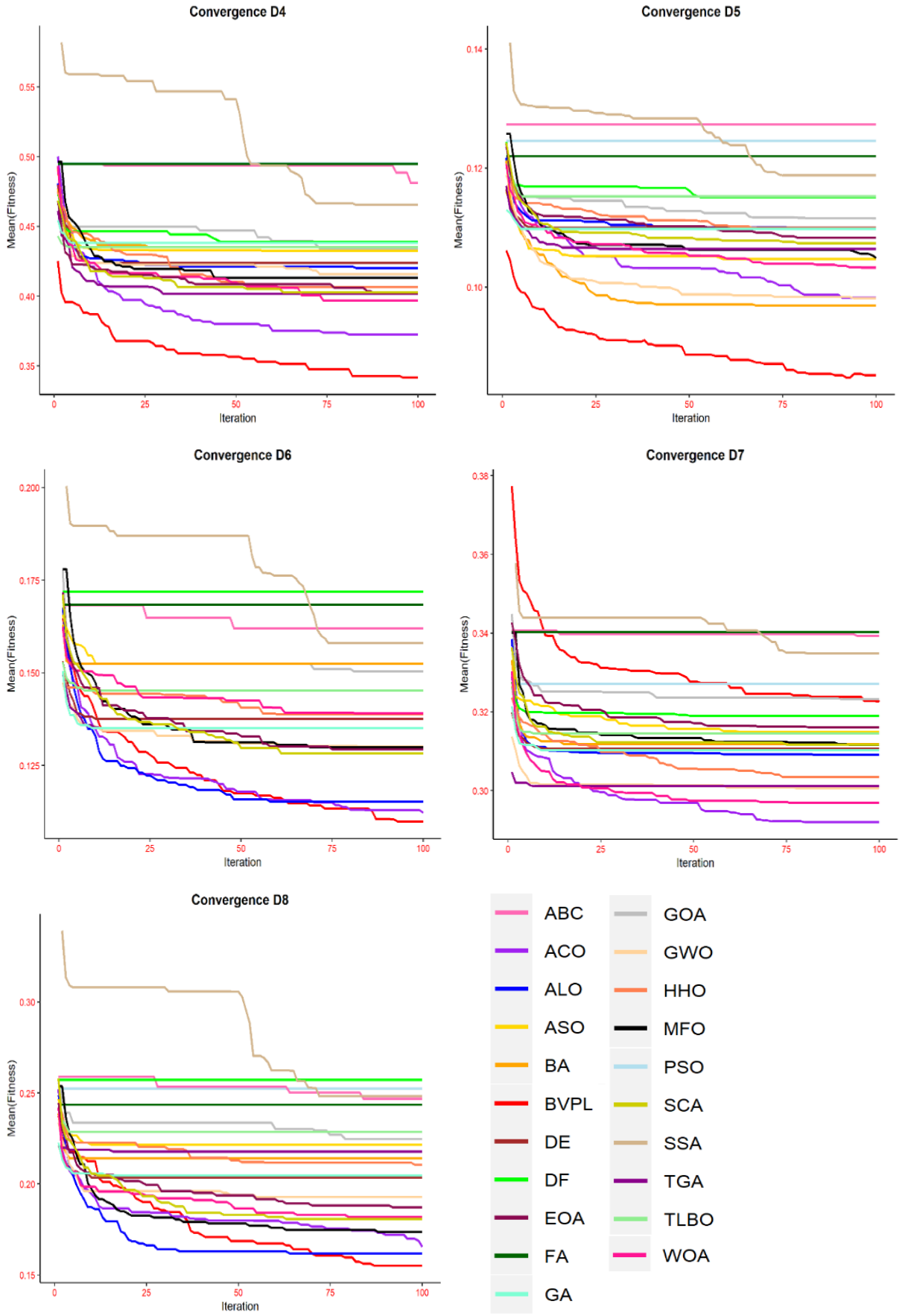


Figure 3.8. The convergence curves of BVPL versus the other MHOAs for the last five datasets

Statistical tests are utilized to assess the significance of the solutions provided by BVPL_BALO in comparison to those produced by other MHOAs. Here the significance is measured combining the parametric t-test and the non-parametric Wilcoxon sum-rank test. To assess the outcomes of each MHOA, the minimum fitness values achieved in each run was recorded. To evaluate the effectiveness of the hybrid approach, two first controls are employed, the Shapiro test to assess the normality of the fitness values for BVPL_BALO with each MHOAs, followed by the Levene's test to examine the homogeneity of the variances. The independent t-test is employed when the variances are homogenous and the fitness values are normally distributed, otherwise the Wilcoxon sum-rank test is utilized. The null hypothesis being assessed is that there is no difference in means (medians) in terms of optimum fitness between the hybrid metaheuristic and the other MHOAs. The proposed algorithm's fitness differs substantially from the compared methods if the p-value is less than 0.05.

Tables 3.17 and 3.18, present the p-values resulting from the use of t-tests or Wilcoxon sum-rank tests. The bolded values show the p-values below 0.05. These tests have been conducted to examine the difference in average fitness between BVPL and the remaining 20 MHOAs. If the p-value is less than 0.05, it can be concluded that there is a significant difference in the average fitness of BVPL when compared to the other algorithm. In the event that BVPL demonstrates superior performance, it will be represented by the symbol "+". Conversely, if BVPL exhibits equivalent performance, it will be signified by the symbol "=". Lastly, if BVPL demonstrates inferior performance, it will be indicated by the symbol "– ". The performance of BVPL surpasses that of 17 MHOAs, exhibiting superior results in over 50% of the datasets.

Table 3.17. The p-value results of BVPL against the 10 first metaheuristics

Dataset	GOA	PSO	ABC	ACO	ALO	ASO	BA	DE	DF	FA
D1	**1.57E-06**	**6.19E-04**	**6.14E-09**	**3.96E-02**	**3.82E-01**	5.68E-02	**1.77E-02**	5.86E-01	**5.18E-04**	**1.28E-06**
D2_S	**1.67E-04**	**1.36E-03**	**1.59E-06**	2.91E-01	**3.33E-02**	**3.44E-02**	**1.51E-04**	8.63E-02	**3.88E-03**	**5.40E-05**
D2_M	**3.65E-05**	**4.54E-04**	**8.16E-05**	4.90E-01	**2.42E-03**	3.92E-01	**1.28E-04**	**6.79E-04**	**7.32E-03**	**1.65E-06**
D3_S	**5.20E-04**	**9.69E-06**	**2.91E-08**	7.41E-01	**3.45E-02**	**6.08E-03**	**6.03E-04**	1.34E-02	**1.07E-03**	**5.26E-08**
D3_M	**3.17E-04**	**1.52E-07**	**1.78E-06**	3.61E-01	**3.42E-04**	**3.69E-02**	**2.90E-05**	**2.66E-03**	**3.19E-04**	**5.54E-07**

D4	**4.17E-09**	**2.70E-12**	**1.86E-11**	**1.77E-02**	**3.81E-08**	**3.13E-08**	**3.47E-06**	**3.48E-06**	**7.72E-09**	**6.74E-08**
D5	**1.75E-06**	**8.47E-10**	**2.41E-10**	**4.78E-03**	**2.83E-06**	**1.84E-05**	**6.11E-03**	**4.37E-06**	**2.43E-09**	**6.98E-10**
D6	**2.05E-05**	**2.17E-08**	**8.26E-08**	7.42E-01	4.66E-01	**2.01E-06**	**7.37E-05**	**2.52E-03**	**4.47E-09**	**2.38E-08**
D7	6.75E-01	4.33E-01	**2.58E-03**	**5.19E-08**	**1.12E-02**	1.72E-01	6.97E-02	**3.12E-02**	4.41E-01	**1.98E-03**
D8	**1.44E-06**	**3.91E-08**	**1.06E-07**	3.97E-01	7.25E-01	**2.56E-05**	**1.29E-04**	**1.43E-03**	**1.09E-07**	**1.96E-07**
+/-/=	9/1/0	9/1/0	10/0/0	4/6/0	8/2/0	7/3/0	9/1/0	7/3/0	9/1/0	10/0/0

The performance of BVPL does not appear to be superior to ACO, except in the case of four specific datasets. Additionally, when considering the performance of WOA and GA, they are found to be equally superior to BVPL

.

Table 3.18. The p-value results of BVPL against the 10 last metaheuristics

Dataset	GWO	HHO	MFO	TGA	TLBO	WOA	EOA	GA	SCA	SSA
D1	5.19E-01	1.32E-01	**1.87E-03**	2.17E-01	**4.39E-05**	3.63E-01	**3.08E-03**	9.24E-01	**4.69E-04**	**2.04E-06**
D2_S	7.95E-02	**1.22E-02**	**3.82E-02**	4.65E-01	**6.54E-06**	6.33E-01	**9.32E-04**	5.70E-01	**2.31E-04**	**3.36E-07**
D2_M	**4.65E-03**	**5.07E-03**	**3.87E-05**	**1.27E-02**	**3.89E-04**	**1.23E-02**	**2.89E-03**	5.81E-02	**1.08E-04**	**4.43E-07**
D3_S	3.09E-01	2.53E-01	**1.08E-02**	**2.32E-01**	**3.12E-03**	1.78E-01	**6.14E-03**	2.00E-01	**4.44E-03**	**8.29E-08**
D3_M	2.28E-01	2.82E-01	6.77E-02	9.58E-02	**1.28E-04**	1.68E-01	**5.09E-05**	1.43E-01	**2.39E-02**	**5.16E-07**
D4	**1.03E-06**	**2.73E-05**	**3.92E-07**	**7.60E-06**	**4.60E-10**	**1.53E-05**	**5.45E-06**	**6.14E-09**	**1.49E-05**	**6.77E-10**
D5	**3.25E-03**	**5.16E-06**	**1.46E-04**	**1.79E-05**	**3.24E-08**	**7.76E-05**	**7.94E-06**	**6.86E-07**	**6.23E-06**	**2.86E-08**
D6	**1.36E-02**	**1.23E-03**	**3.18E-03**	**1.22E-03**	**3.75E-05**	**7.54E-04**	**6.80E-03**	**1.23E-03**	**3.96E-02**	**2.38E-06**
D7	**1.15E-04**	**3.56E-04**	**4.10E-02**	**9.28E-06**	1.37E-01	**7.27E-06**	2.21E-01	**1.25E-02**	5.98E-02	5.48E-02
D8	**7.40E-03**	**4.19E-04**	1.46E-01	**4.76E-08**	**1.76E-08**	1.29E-01	**1.33E-02**	**3.17E-04**	**4.71E-02**	**2.67E-06**
+/-/=	6/4/0	7/3/0	8/2/0	7/3/0	9/1/0	5/5/0	9/1/0	5/5/0	9/1/0	9/1/0

To summarize, the metaheuristic BVPL in feature selection problem, has provided a higher accuracy in predicting Parkinson, in 10 different datasets, compared with a large list of metaheuristics. BVPL outperforms ACO in fitness and accuracy across five

datasets, while ACO outperforms in one. SCA outperforms ACO in one dataset, with the lowest values across all datasets. The BVPL algorithm demonstrates an acceptable speed of convergence and effectiveness in searching across a wide range of datasets, consistently ranking among the top three among other MHOAs, and superior in three of them.

3.4 Results on improving the effectivity of Binary Volleyball Premier League

Therefore, in this paragraph are presented the results from applying OBL into BVPL, and the new hybrid metaheuristic BVPL_BALO which are used to improve on the effectivity of BVPL in predicting Parkinson. The datasets, and experiment settings are the same as in the experiment 2.

3.4.1 Results for Opposition-based learning Binary Volleyball Premier League algorithm

Table 3.19 provides a summary of the four metrics, average features, standard deviation of fitness, average accuracy, and average number of selected features which will compare for BVPL against OBL_BVPL.

Table 3.19. Results BVPL against OBL_BVPL

Algorithm	**BVPL**				**OBL_BVPL**			
Dataset	$\mathbf{f_{avg}}$	$\mathbf{f_{sd}}$	$\mathbf{acc_{avg}}$	$\mathbf{feat_{avg}}$	$\mathbf{f_{avg}}$	$\mathbf{f_{sd}}$	$\mathbf{acc_{avg}}$	$\mathbf{feat_{avg}}$
D1	0.05338	0.01874	0.94723	2.5	0.01848	0.00539	0.98274	3.05
D2_S	0.06385	0.01782	0.93724	2.05	0.05601	0.00747	0.94654	3.7
D2_M	0.05716	0.01763	0.94433	2.45	0.05082	0.00864	0.95216	4.15
D3_S	0.14256	0.03281	0.85924	3.85	0.10058	0.01898	0.90241	4.75
D3_M	0.13584	0.02114	0.86557	3.3	0.09235	0.00385	0.91013	4.05
D4	0.34154	0.03427	0.65643	3.65	0.30865	0.01973	0.68971	3.8
D5	0.08522	0.01227	0.91593	149.8	0.08392	0.01209	0.91728	152.3
D6	0.10983	0.02630	0.89040	5.95	0.08794	0.01235	0.91249	5.85
D7	0.32260	0.01530	0.67564	3.85	0.28404	0.01429	0.71523	5.5
D8	0.15493	0.03445	0.84429	4.2	0.11701	0.02745	0.88298	6.3

The suggested technique exhibits significant enhancements in terms of average fitness and accuracy across all datasets. Incorporating the opposing approach leads to a

considerable increase in accuracy and a decrease in fitness. The observed improvement in accuracy ranges from 0.135% in D5 to 4.456% for D3_M. In relation to efficacy, this technique has demonstrated major relevance in the prediction of Parkinson's disease. Opposition-based learning can enhance the prediction of Parkinson above 90% in 7 out of 10 datasets.

3.4.2 Results from Binary Volleyball Premier League and Antlion Optimizer metaheuristic algorithm

In this subsection, the main goal is to apply and validate the hybrid metaheuristic algorithm BVPL_BALO in the list of the 10 Parkinson datasets, and to compare with some prominent metaheuristics which provided the better results in section 3.3.2. The metaheuristics are: BVPL, binary ALO, SCA, and ACO metaheuristic algorithm. This validation is important in order to offer a better solution of BVPL in predicting Parkinson with a higher accuracy. Table 3.20 contains information about the datasets, their dimension, the values of the target variable, and the utilized TF selected according the trials in subsection 3.3.2.

Table 3.20. The 10 Parkinson benchmark datasets

Dataset	Dimension	Class	TF
D1	195x23	2(1 = PD, 0 = HC)	V3
D2_S	368x16 (13)	2 (1 = HC, 2 = PD)	V3
D2_M	368x16 (13)	2 (1 = HC, 2 = PD)	S2
D3_S	264x16 (13)	2 (1 = HC, 2 = PD)	S3
D3_M	264x16 (13)	2 (1 = HC, 2 = PD)	S3
D4	130x65 (27)	3(PD=1, RBD[a] =2, HC= 0)	S2
D5	756x754	2 (1 = PD, 0 = HC)	S3
D6	240x48 (46)	2 (1 = PD, 0 = HC)	V4
D7	1040x29 (27)	2 (1 = PD, 0 = HC)	S2
D8	192x60 (148x 55)	2(1 = prodromal, 3 = PD)	V3

It is used again the k-nearest neighbor (k-NN) classifier using a Euclidean distance metric with a k-neighbor value of 5. The accuracy of the supervised learning algorithm was employed in a fitness function to evaluate the efficacy of the chosen feature subsets. The datasets are divided into training and testing datasets with the ratio 70:30. To reduce the overfitting problem, k-fold cross-validation with kfold = 5 was utilized.

A min-max normalization technique is employed to normalize all the features of the datasets within the range of 0 to 1. Table 3.21 presents the parameters of the other metaheuristics whom the hybrid is compared.

Table 3.21. The parameters of the metaheuristics

Algorithms	Parameters
General	nRuns = 20; maxiter = 100, NPop = 6; α = 0.99, k=5
BVPL	fall_rate=0.15, transport_rate = 0.5, β=2, b from β to 0
BACO [270]	tau =1, eta = 1, α = 1, β = 0.1, ρ= 0.2.
BALO [268]	-
BSCA [285]	r1 decreases linearly from 2 to 0, r2, r3, r4 ϵ [0,1]

3.4.2.1 Comparison of the hybrid metaheuristic vs other metaheuristics

Table 3.22 presents results of the performance metrics related to average fitness (avg(fit)), standard deviation of the fitness (sd(fit)), average accuracy (avg(acc)), and average of selected features (avg(feat)).

Table 3.22. The results of the metrics for the hybrid against other metaheuristics

Dataset	MHOA	Avg(fit)	Sd(fit)	Avg(acc)	Avg(feat)
D1	BVPL	0.053379	0.018743	0.947230	**2.5**
	BALO	0.059486	0.024495	0.943103	6.75
	BVPL_BALO	**0.014506**	**0.007155**	**0.987115**	3.85
	BACO	0.040299	0.020041	0.963793	9.95
	BSCA	0.076581	0.019466	0.924138	2.7
D2_S	BVPL	0.063845	0.017816	0.937235	**2.05**
	BALO	0.085814	0.043584	0.916055	2.8
	BVPL_BALO	**0.049251**	**0.008463**	**0.954587**	5.15
	BACO	0.058370	0.014289	0.944954	4.75
	BSCA	0.098881	0.037083	0.901803	2.1
D2_M	BVPL	0.057158	0.017634	0.944327	**2.45**
	BALO	0.073128	0.018895	0.932110	7.15
	BVPL_BALO	**0.040951**	**0.011221**	**0.962718**	4.85
	BACO	0.059774	0.020828	0.943578	4.85
	BSCA	0.101392	0.047918	0.901835	4.45
D3_S	BVPL	0.142560	0.032805	0.859241	**3.85**
	BALO	0.165859	0.034362	0.839873	8.6
	BVPL_BALO	**0.080607**	**0.000689**	**0.924051**	6.5
	BACO	0.145896	0.030426	0.858228	6.8
	BSCA	0.176182	0.037267	0.827215	5.8

D3_M	BVPL	0.135835	0.021138	0.865570	**3.3**
	BALO	0.163812	0.023694	0.841772	8.85
	BVPL_BALO	**0.071963**	**0.006467**	**0.931519**	5
	BACO	0.128687	0.027367	0.875316	6.45
	BSCA	0.111512	0.047812	0.892201	5.25
D4	BVPL	0.341538	0.034274	0.656430	**3.65**
	BALO	0.420173	0.037956	0.583333	19.7
	BVPL_BALO	**0.3095**	**0.026907**	**0.689763**	6.15
	BACO	0.372519	0.043923	0.628205	12
	BSCA	0.402673	0.042648	0.597436	11.4
D5	BVPL	**0.085218**	0.012270	**0.915930**	**149.75**
	BALO	0.109861	0.015638	0.897124	605.35
	BVPL_BALO	0.087538	**0.011656**	0.915503	292.65
	BACO	0.098193	0.014918	0.905530	351.7
	BSCA	0.107392	0.014310	0.896239	351.25
D6	BVPL	0.109825	0.026297	0.890401	5.95
	BALO	0.115197	0.019299	0.8875	16.9
	BVPL_BALO	**0.075396**	**0.018124**	**0.925683**	8.2
	BACO	0.112271	0.019972	0.890972	19.7
	BSCA	0.128288	0.026980	0.871528	**3.4**
D7	BVPL	0.322596	0.015304	0.675641	**3.85**
	BALO	0.309019	0.016842	0.696154	21.35
	BVPL_BALO	**0.269404**	**0.012301**	**0.735897**	20.65
	BACO	0.291923	0.013163	0.710256	13.45
	BSCA	0.311663	0.019952	0.689423	12.2
D8	BVPL	0.154931	0.034449	0.844290	4.2
	BALO	0.161773	0.058918	0.839773	17.1
	BVPL_BALO	**0.078278**	**0.030837**	**0.922727**	9.6
	BACO	0.165403	0.042396	0.8375	24.6
	BSCA	0.180713	0.044262	0.818182	**3.35**

[a.] The underlined and bold values show the better metrics

The data reveals that the hybrid approach exhibits superior performance compared to the other methods across 90% of the datasets, as seen by its higher values for maximum, average, and standard deviation of fitness, as well as average accuracy. In relation to the average number of features, it is seen that the hybrid approach exhibits an increase in the number of features, in contrast to BVPL, which yields a lower feature ratio in 80% of the datasets. The utilization of BVPL_BALO in the D5 dataset does not produce any noticeable improvement. The final accuracy obtained from the hybrid exceeds 90% in most of the datasets, which is an adequate result in predicting PD. However, for datasets D4 and D7, the accuracies remain somewhat low, despite the hybrid approach managing to increase them by approximately 3.33% and 6.03% respectively from BVPL. This means that these two datasets should be observed carefully in the future in order to improve their accuracy. Additionally, it has been

shown that the BACO algorithm outperforms BVPL in terms of accuracy for almost half of the datasets.

3.4.2.2 Convergence curves

For a complete idea of how fitness changes in each iteration of each run for the MHOAs, the respective convergence curves for the BACO, BALO, BVPL, BSCA, and BVPL_BALO are presented. The aim is to emphasize the trend of convergence in each iteration for each dataset. Figure 3.9, and 3.10 shows the graphical illustration of the convergence curves for the five metaheuristics together, where each color represents a different metaheuristic.

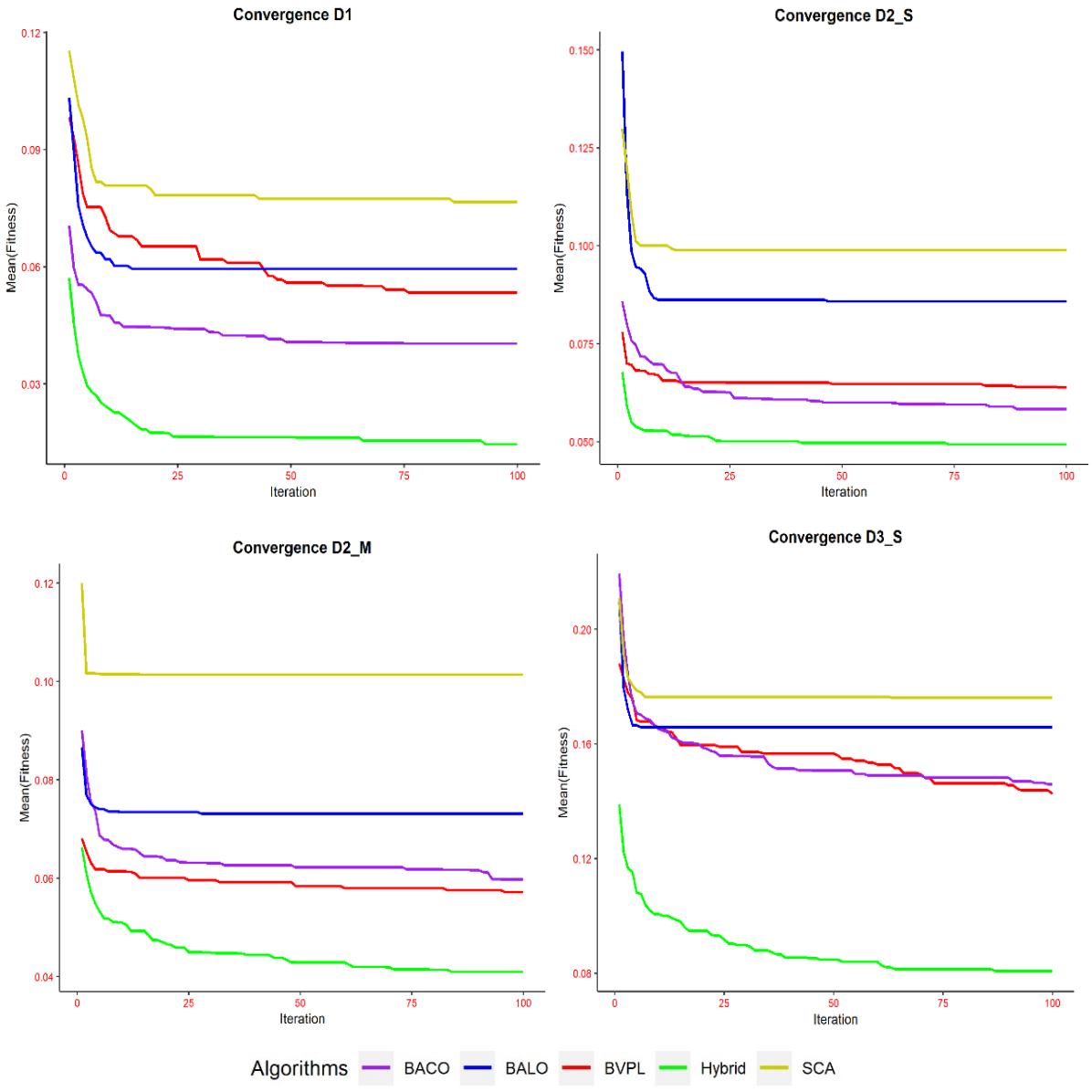


Figure 3.9. The convergence curves of the metaheuristics for the first 4 Parkinson's datasets

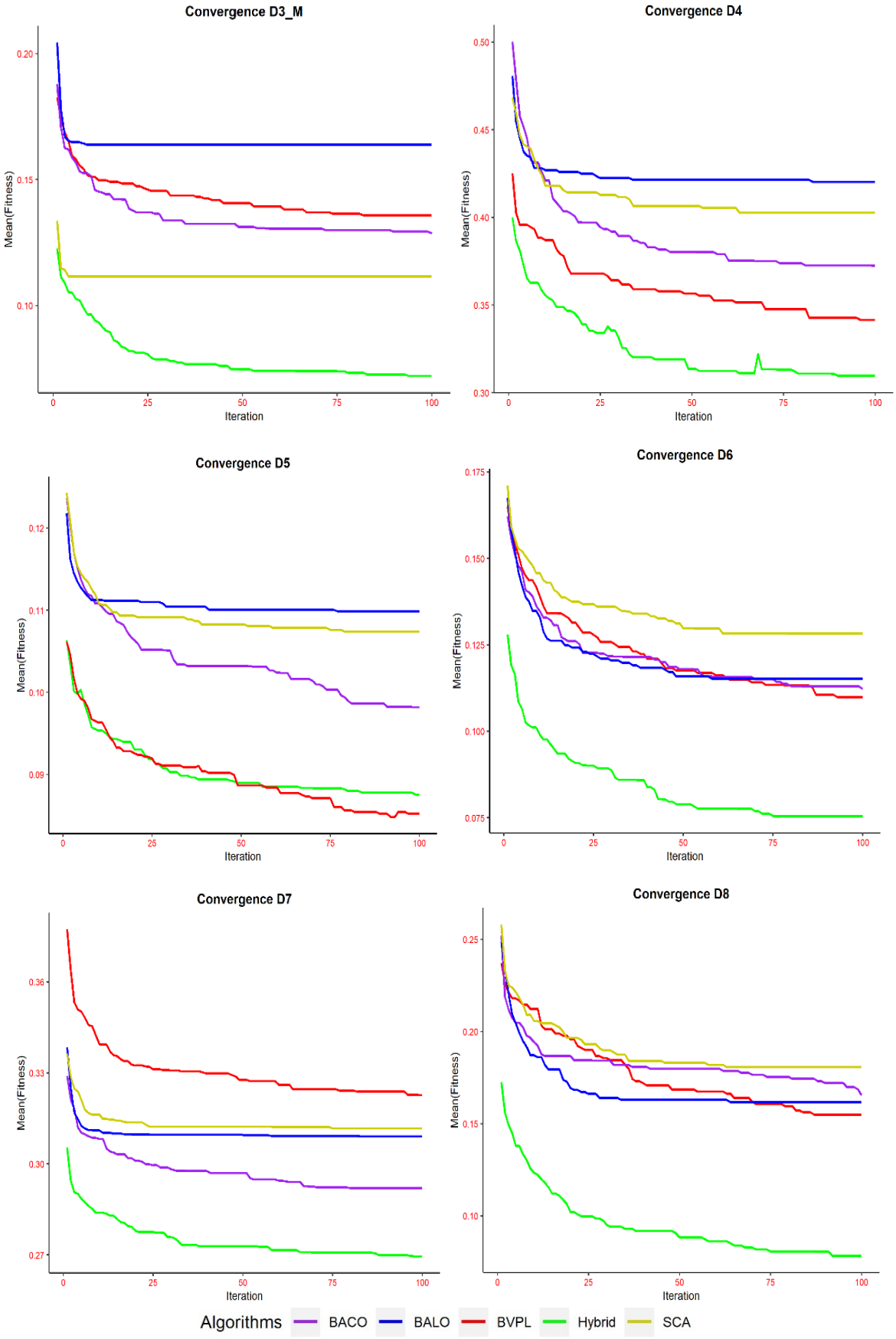


Figure 3.10. The convergence curves for the other Parkinson's datasets

Based on the data presented in the charts, it can be observed that the hybrid shows consistently lower fitness values in each iteration, except for the D5 dataset. This means that the algorithm has shown efficacy since the first iterations. Regarding the BVPL, it has provided the best convergence for one dataset and is the second best for 5 datasets out of 10. All the experiments show that the hybrid BVPL_BALO converges faster to the optimum for nine datasets.

3.4.2.3 Statistical tests results

Table 3.23 presents the provided p-values generated by the Wilcoxon sum-rank test or t-test for the comparison between the average fitness of BVPL_BALO and four other MHOAs.

Table 3.23. The p-value results

Dataset	BVPL	BACO	BALO	BSCA
D1	8.08E-07	3.79E-06	4.43E-08	4.37E-08
D2_S	1.10E-02	2.98E-02	8.38E-05	1.75E-06
D2_M	3.71E-04	4.72E-04	3.36E-06	3.98E-07
D3_S	3.33E-08	2.15E-07	3.40E-08	3.40E-08
D3_M	4.21E-08	1.10E-07	4.22E-08	1.51E-03
D4	3.39E-03	6.39E-06	6.34E-08	3.45E-07
D5	**5.44E-01**	1.64E-02	1.11E-05	2.60E-05
D6	2.98E-05	4.11E-07	5.95E-08	2.28E-08
D7	7.69E-08	2.11E-06	5.31E-10	3.62E-09
D8	4.21E-06	1.12E-08	1.36E-06	6.54E-10

The p-value < 0.05 tells that the mean difference of the proposed hybrid is statistically significant between the other MHOAs. The only case where BVPL_BALO doesn't show a significant difference with BVPL is for the D5 dataset, and is highlighted (p-value >0.05).

The newly proposed metaheuristic demonstrates superior performance compared to all other existing MHOAs across 90% of the datasets. The experimental results show that BVPL_BALO is more competitive in terms of converging faster to the optimum, and in predicting with a higher accuracy the PD. The convergence plots and statistical tests both support these results.

3.5 Results on improving the efficiency of the hybrid Binary Volleyball Premier League and Antlion Optimizer metaheuristic algorithm

3.5.1 Results after integrating the occurrence list in the cost function

In this subsection, is evaluated the effect that has integrating the occurrence list in the hybrid metaheuristic BVPL_BALO. A comparative analysis was conducted to assess the execution times of BVPL_BALO vs the other metaheuristics. The aim was to evaluate the reduction in time achieved by BVPL and to determine the magnitude of variations between the other algorithms. The D5 dataset has been incorporated for this test due to its higher number of dimensions. The final results are shown in Table 3.24.

Table 3.24. Execution time for D5 dataset

MHOAs	Time
BVPL	7.4888 days
BALO	2.8644 hours
BVPL_BALO	2.5463 days
BSCA	1.3866 hours
BACO	1.4925 hours

It is observed that the execution time of BVPL is significantly reduced when BVPL_BALO is employed. This represents a good approach for future evaluations of BVPL since its complexity is higher. The other metaheuristics offer more promising computational times in relation to BVPL when used in this particular dataset. The proposed improvement in the hybrid technique leads to a decrease of execution time of approximately 2.94 times; yet, this decrease remains unsatisfactory. Taking this in consideration, in the next subsection, another approach is proposed for reducing the execution time by using the combination of the cosine similarity, and metaheuristic BVPL_BALO are presented.

3.5.2 Results after using cosine similarity and the hybrid metaheuristic algorithm

The interpretation of the results is given with three different comparisons. Initially, the proposed technique is implemented on the high-dimensional Parkinson dataset, D5. Then it is applied on the other datasets in order to test the effect of the method, and

finally an observation of the similarity of the features between the proposed method, and when applying only the hybrid metaheuristic algorithm.

3.5.2.1 Results for D5 dataset

In the proposed method, the phase 1 concludes with ranking the features according to their importance, and were selected the top 2.5%, 5%, 10%, 15%, 20%, 25%, 30%, and 50% of features for extraction. The selection process for the percentage was driven by the goal of minimizing the execution time of BVPL_BALO. To achieve this, a brute force method was employed to identify the optimal percentage of features that would produce comparable results when using the entire dataset. The measurement results for each feature include the execution time, average (avg), standard deviation (sd) of fitness (fit), average accuracy (acc), and average number of selected features (feat). The new method compares metrics from the dataset with all features (D5_hybrid100%) to those from the dataset with fewer features. The size of rows and features in the dataset influences the initial ranking time, which is approximately 5 days. However, you only need to perform this procedure once, and you can choose different percentages without repeating the ranking. The results are shown in Table 3.25.

Table 3.25. The results of the metrics for each percent of selected features

Dataset	Performance metrics				
	Time(h)	***Avg(fit)***	***Sd(fit)***	***Avg(acc)***	***Avg(feat)***
D5_2.5%	1.2244 h	0.2149	0.0093	0.7852	4.05
D5_5%	6.3734 h	0.1637	0.0106	0.8374	10.1
D5_10%	9.7011 h	0.1325	0.0104	0.8685	17.8
D5_15%	12.2934 h	0.1204	0.0106	0.8805	23.55
D5_20%	13.0052 h	0.1141	0.0128	0.8870	32.65
D5_25%	16.7052 h	0.1027	0.0140	0.8985	40
D5_30%	17.8223 h	0.1024	0.0107	0.8987	48.4
D5_50%	**26.7905 h**	**0.0863**	**0.0129**	**0.9150**	**81.35**
D5_hybrid100%	61.1112 h	0.0875	0.0117	0.9155	292.65

The calculations clearly show that extracting 50% of the features yields metrics that are exactly the same as when the BVPL_BALO uses the full feature (original) dataset as input. At the same time, the execution time was reduced by 56.16% and the number of selected features minimized by 72.2%.

3.5.2.2 Results for the other datasets

The identical suggested method was implemented on the remaining datasets, but with a focus on only 50% of the most significant features, since it provided better results for the D5 dataset. This allows for a comparison between this subset of features (referred to as D$_50%) and the complete set of features (D$_Hybrid100%). Table 3.26 summarizes the experiment's findings.

Table 3.26. The metrics for 50% and 100% of the features

Dataset	Performance metrics				
	Time(m)	***Avg(fit)***	***Sd(fit)***	***Avg(acc)***	***Avg(feat)***
D1_50%	11.7123	0.0251	0.0093	0.9793	5.05
D1_Hybrid100%	63.924	0.0145	0.0072	0.9871	3.85
D2_S50%	2.9290	0.0851	0	0.9174	2
D2_S_Hybrid100%	11.1710	0.0493	0.0085	0.9546	5.15
D2_M50%	3.2653	0.0836	0.0012	0.9229	4.4
D2_M_Hybrid100%	11.0229	0.0410	0.0112	0.9627	4.85
D3_S50%	3.8261	0.1804	0	0.8228	3
D3_S_Hybrid100%	16.1578	0.0806	0.0007	0.9241	6.5
D3_M50%	3.3871	0.2289	0	0.7721	2
D3_M_Hybrid100%	12.2148	0.0720	0.0065	0.9315	5
D4_50%	41.0500	0.3407	0.0307	0.6590	4.05
D4_Hybrid100%	279.594	0.3095	0.0269	0.6898	6.15
D6_50%	136.719	0.1011	0.0152	0.8997	4
D6_Hybrid100%	415.478	0.0754	0.0181	0.9257	8.2
D7_50%	20.8601	0.2923	0.0128	0.7090	5.4
D7_Hybrid100%	247.531	0.2694	0.0123	0.7359	20.65
D8_50%	253.810	0.1030	0.0180	0.8977	4.7
D8_Hybrid100%	425.641	0.0783	0.0308	0.9227	9.6

In the D1 dataset, the results provided by the extracted 50% of the features are very similar to the case of 100% of the features. The average accuracy decreases by 0.8%, the number of features increases by approximately 31.17%, and the time decreases by 81.68%. The D2_S dataset exhibits a decrease in average accuracy of 3.9%, a decrease in the number of features of about 61.17%, and a decrease in time of 73.78%. The D2_M dataset exhibits a decrease in average accuracy of 4.13%, a reduction in features of about 9.28%, and a decrease in time of 70.38%.

The D3_S dataset shows a 10.96% decrease in average accuracy, a 53.85% decline in the number of features, and a 76.32% decrease in time. In the D3_M dataset, the

average accuracy has decreased by 17.11%, the number of features has fallen by 60%, and the time has decreased by 72.27%. Both of these datasets experience a significant decrease in the number of features and execution time, but their accuracy is greatly impacted.

The D4 dataset shows a decrease in average accuracy by 4.47%, a reduction of features by 34.15%, and a decrease in time by 85.32%. Within the D6 dataset, there is a decrease in accuracy by 2.81%, a reduction in the number of features by 51.22%, and a decrease in time by 67.09%. The D7 dataset shows a 3.66% decrease in average accuracy, a 73.85% reduction in features, and a 91.57% decrease in time. The D8 dataset shows a 2.71% decrease in average accuracy, a 51.04% drop in the number of features, and a 40.37% decrease in time.

The proposed strategy significantly reduces the execution time in all datasets, with a maximum impact of 4.47% on the accuracy of the selected number of features for seven datasets. A change of approximately 10.96% and 17.11% significantly influences the accuracy in the remaining two datasets, while improving the execution time and selected features. An explanation could be that some datasets require a larger number of features in order to achieve a satisfactory level of accuracy. Generally, the results show that choosing 50% of the features is not always a guarantee that the results will be the same as for the full feature dataset.

3.5.2.3 Results on similarity

Lastly, at this experiment, 50% of the ordered features extracted using cosine similarity are compared to the top 50% of the most selected features when the hybrid metaheuristic is applied to the whole feature dataset. The objective is to examine whether there is a convergence of features between both methods. Table 3.27 contains the results of this comparison. The first column stores the similarity of the features, the second stores the dimensions of the datasets, and the third one compares the accuracy difference between BVPL_BALO with 100% of the features vs 50% of the features.

Table 3.27. The similarity for each dataset

Dataset	Similarity		
	% Similarity	***Dimensions***	***Difference in accuracy***
D1	45.5%	23	0.8%
D2_S	50%	13	3.9%
D2_M	33.3%	13	4.13%
D3_S	50%	13	10.96%
D3_M	33.3%	13	17.11%
D4	53.8%	27	4.47%
D5	52.9%	754	0.05%
D6	48.9%	46	2.81%
D7	53.8%	27	3.66%
D8	48.1%	55	2.71%

This comparison aims to illustrate how accuracy changes with the number of dimensions and the similarity of features. The similarity calculations were presented in the first column using the idea of Eq. (2.55).

$$\%\ similarity = \frac{number\ of\ similar\ features}{(dim/2)} \tag{2.55}$$

The majority of the datasets have a similarity range of 45.5 % to 53.8% among their features from the two comparisons, except for two datasets that have a similarity of 33.3%. There is no evident association between dimensions or similarities that affects the difference in accuracy. This approach appears to be useful in certain datasets, such as D1 and D5 dataset.

The proposed method offers significant advantages in terms of reducing the execution time, with a range of improvement from 40.37% to a maximum of 91.57%. Additionally, it reduces the number of features within a range of 9.28% to 73.85%. However, there is a trade-off in terms of accuracy, ranging from a minimal decrease of 0.05% to a significant decrease of 17.11%. Moreover, when using cosine similarity, the ranking of the top 50% significant features generally produces an average similarity range of approximately 47%, as opposed to solely utilizing BVPL_BALO in the full features dataset.

The similarity approach does not guarantee that the features will be identical, or nearly identical, in the case of using only the hybrid metaheuristic as a feature selection method. The user can alter the percentage of extracted features to determine the number of crucial features that are important to the dataset.

The suggested method can be advantageous in scenarios where prioritizing execution time is more crucial than maintaining relatively high accuracy. Our findings show that although the dataset has a major impact on accuracy loss, most datasets nevertheless produce results with acceptable accuracy. These two approaches are not interchangeable but provide two perspectives for choosing the most significant features based on two criteria: execution time and accuracy.

The proposed method can be applied on other low-to-high dimensional datasets, in the same way and its application is not affected by the input dataset. The first phase of ranking the features could also be combined with other metaheuristics besides BVPL_BALO.

3.6 Chapter conclusions

In this chapter, the novel metaheuristics algorithms, and methods used in the FS problem for forecasting Parkinson presented on Chapter 2, are applied and validated. The experiments were performed in a PC with an Intel (R) Core (TM) i5-8365U CPU @ 1.60 GHz and 1.90 GHz, 16 GB of RAM. All the codes are written and executed in RStudio environment.

- First, a literature review was conducted about the use of metaheuristics in predicting Parkinson based on seven public datasets. This was achieved by examining a total of thirty-four scientific articles. The papers are assessed in several categories, including metaheuristic algorithms, supervised learning methods in machine learning, outcome metrics, fitness calculation, resampling methods, statistics tests, and results related to accuracy and number of features. The results indicate the frequent use of the particle swarm optimization algorithm, the k-nearest neighbor classifier, the accuracy metric, and the D1 dataset. 10-fold cross-validation and the Wilcoxon sum-rank test are the most frequently used. The structure of publications reveals that even with identical data, variations in metaheuristics, classifiers, hyper-parameter optimization, performance indicators, and fitness evaluation can yield seemingly superior results.

- Second, it is proposed a comparative assessment employing three distinct filter methods, three wrapper methods, and GA to identify the most crucial features for Parkinson prediction in D1 dataset. The evaluation of the subset was based on three classifiers: k-NN, radial basis function SVM, and RF. This comparative analysis also included the GSA to optimize the parameters of each of the three classifiers. Using GSA or not, Joint Mutual Information and k-NN provided the best accuracy (98%). The combination of GA and k-NN achieves the best results without using GSA (acc = 95%). Moreover, k-NN gives a better prediction of accuracy = 97%, sensitivity = 96%, specificity 100%, and precision 100% when using GSA and k-NN for the dataset with the full features.
- Next, a binary VPL metaheuristic algorithm for FS is proposed for the first-time using two-step binarization (S-shaped and V-shaped functions, complement, and standard method). Initially, it is applied to the same Parkinson dataset, D1, and the results are compared with the binary PSO. This model generates very low fitness values for minimum, maximum, average, and standard deviation, as well as a lower number of features. This is a first attempt to evaluate the suitability of binary VPL in FS. Additional experiments must be provided.
- The next section presents a detailed comparative analysis to estimate the effectiveness and efficiency of BVPL. This analysis includes 20 MHOAs, 10 Parkinson datasets, 5 S-shaped and 5 V-shaped TF, an analysis of convergence, and a statistically measured change in fitness. This second experiment demonstrated BVPL's strong competitiveness with most of the MHOAs, with binary ACO being the most competitive with him. BVPL's convergence speed increases with the number of iterations, and in three datasets, it converges more quickly than the others. In four datasets, it predicts Parkinson with an accuracy greater than 90%. The experiment revealed the need to improve the accuracy of BVPL's Parkinson prediction and address its lengthy execution time, particularly for the high-dimensional Parkinson dataset D5 (754 features).
- There are two proposed improvements to the BVPL's effectivity in FS. The first enhancement to BVPL involves integrating OBL solutions as a technique to search for a better solution than the one BVPL ultimately provides. In this way,

incorporating OBL improved the accuracy of predicting Parkinson above 90% in 7 out of 10 datasets. The next improvement is integrating a binary ALO into the learning phase of BVPL in order to select the better solution provided by each of them. If BVPL doesn't find a better optimum, it uses BALO to enhance the quality of the solutions it generates. The conditions of the experiment have not changed, which confirms that the hybrid BVPL_BALO performs better than all other MHOAs (binary VPL, ALO, ACO, and SCA) in 90% of the datasets. The experimental results demonstrate that BVPL_BALO is more competitive in terms of converging faster to the optimum and predicting the Parkinson with higher accuracy. The convergence plots and statistical tests both support these results.

- Apart from effectivity, there are also two proposed improvements to the efficiency of the BVPL on FS related to execution time. Most BVPL phases include the calculation of team costs, which leads to an increase in BVPL execution time, particularly in high-dimensional datasets. Therefore, the proposed hybrid metaheuristic BVPL_BALO first integrates it into an occurrence list, storing the teams and their corresponding cost function values. The proposed improvement in the hybrid technique leads to a decrease in execution time of approximately 2.94 times compared to BVPL for D5 dataset. The second one concentrates on enhancing the BVPL_BALO's execution time in the 10 Parkinson datasets, primarily targeting the high-dimensional dataset D5. It employs a method that integrates cosine similarity to assess the significance of each feature in the input dataset and subsequently ranks them. After that, in the second phase, the user pre-defines a percentage of selected features and gives them as an input to the hybrid BVPL_BALO, which will select from them the most important features. This solution offers significant advantages in terms of reducing the execution time for future computations, with a range of improvement from 40.37% to a maximum of 91.57%. Additionally, it reduces the number of features within a range of 9.28% to 73.85%. However, there is a trade-off in terms of accuracy, ranging from a minimal decrease of 0.05% to a significant decrease of 17.11%. Moreover, when using cosine similarity, the

ranking of the top 50% significant features generally produces an average similarity range of approximately 47%, as opposed to solely utilizing BVPL_BALO in the full features dataset.

4. Conclusions - a summary of the results obtained

Conclusions of the thesis

This thesis has examined, analyzed, and suggested novel algorithms, techniques, and methods to address the feature selection issue based on metaheuristic optimization algorithms, with a specific emphasis on predicting Parkinson's. The thesis provides a concise overview of the significance of employing metaheuristic optimization techniques for feature selection, integrating them with machine learning, and extending this field with novel optimization strategies.

Firstly, the dissertation surveys and analyzes the trend of using metaheuristics for feature selection based on Parkinson's, with the results confirming the popularity of combining metaheuristic optimization algorithms with machine learning algorithms. Moreover, a comparative analysis is conducted in order to assess the importance and suitability of integrating filter and wrapper methods, including the genetic algorithm, together with integrating the heuristic generalized simulated annealing for hyper parameter optimization. The results confirmed that each feature selection method has its importance in feature selection, and choosing them is dependent on different conditions. Hyper parameter optimization is very helpful in improving the accuracy of predicting Parkinson’s.

Improved strategies for solving the feature selection problem focusing only on metaheuristics are proposed using a metaheuristic optimization algorithm called the "binary Volleyball Premier League Optimization” algorithm. The "binary Volleyball Premier League Optimization" algorithm, not being proposed before in feature selection, has undergone a redesign to effectively tackle this binary problem. This algorithm replicates the competitive nature of a volleyball game during a league, providing a range of steps to identify optimal solutions in each phase, thereby expanding the search area and enhancing the exploration capability. This metaheuristic redirects improving other existing solutions found so far using the values of the three best teams in the learning phase which affects the exploitation phases of BVPL. The binary VPL algorithm is compared with a considerable number of metaheuristics and on

different Parkinson's datasets and has given very good results in accuracy for predicting PD, a high convergence speed to find the global optimum, and better results in fitness and the number of selected features than the majority of the other metaheuristics on most of the datasets.

Two improvements are proposed for enhancing the accuracy, the optimal solution, convergence, and exploitation of the binary volleyball Premier League algorithm, in other words its effectivity. The first one is the integration of an opposite-based learning technique, where in each iteration of the binary volleyball Premier League algorithm is evaluated the opposite solution of the final optimal solution. These two solutions are compared, and if the opposite solution provides a lower cost function evaluation, then it is retained as the best one and used in the next iteration. OBL solutions contribute to achieving a better optimum, faster convergence on the optimum, and a predictive capability of more than 90% for Parkinson in 70% of the datasets. The second improvement on effectivity for binary Volleyball Premier League, is a new hybrid metaheuristic algorithm which employs the Antlion Optimizer algorithm in the binary volleyball Premier League in order to generate a hybrid BVPL_BALO, which enhances the learning phase of the binary VPL. ALO has very competitive results in terms of improved exploration, local optima avoidance, exploitation, and convergence. Since the learning phase significantly impacts the performance of the VPL algorithm, BALO's method of generating optimal solutions enhances the learning phase of BVPL. If BALO algorithm produces a superior solution, the team table will be updated to reflect the team's higher fitness. As a result, BALO improves BVPL's final solution. The experimental results show that BVPL_BALO is more competitive in terms of converging faster to the optimum and predicting Parkinson with higher accuracy.

One notable disadvantage of binary VPL is its longer execution time. This thesis proposes two improvements using BVPL_BALO for improving its efficiency. The first is to store the results of each generated team's fitness in a list called the occurrence list. If the same solution is generated in different runs or iterations, the fitness is extracted from this list, and it is not recalculated. The second method involves two phases. Firstly, it ranks the features based on their significance, using cosine similarity as a measure for the distance between the dataset's rows after removing each feature individually. Next, the hybrid metaheuristic is applied to a defined number of features

for feature selection. Both approaches improve the execution time by a considerable amount.

Limitations of the study

This thesis explores potential enhancements to the VPL metaheuristic algorithm first employed in FS. While this thesis work shows progress in using metaheuristics for feature selection, it did not fully overcome some limitations.

- A notable shortcoming of binary VPL is the extended duration of execution, requiring around 7.4888 days for a high-dimensional dataset (754x756). This thesis introduces two notable enhancements that significantly decrease the time required for the task, specifically 61.1112 hours and 26.7905 hours. The computer's processor and CPU time primarily influence this longer duration. Thus, enhancing the outcomes can be achieved by utilizing a more robust computer and implementing parallelization techniques for the binary VPL or other combinations of feature selection methods.
- The binary VPL algorithm required a large execution time; therefore, the number of independent runs was 20, and the number of iterations was 100, but larger values could enforce the stability of the final solutions.
- Moreover, for the reasons mentioned in the previous paragraph, only one classifier, k-nearest neighbor, is used for evaluating the quality of the final solutions, and in the future, other supervised learning ML algorithms could be used for evaluating the effectiveness of the binary VPL.
- In terms of metrics, the fitness function we use solely relies on the accuracy metric, which is influenced by imbalanced datasets. Other metrics, like F-score, can be integrated in the future in the fitness function, or other suggested fitness functions could be integrated.

Future research

The implementation of the binary VPL and other modifications have produced positive and helpful outcomes in the feature selection problem. There has been a significant advancement in forecasting Parkinson's integrating metaheuristics for selecting the

optimal subset of features. There are some directions that could be followed for future research:

- The proposed binary VPL, hybrid BVPL_BALO, and other improvements of it includes a lot of random generated numbers, and solutions as chaotic maps, and others can improve more the generated solutions, and the proposed features, as it is shown in the work of [264].
- The proposed metaheuristic algorithm, BVPL, and other improvements could be used in other data related with Parkinson to identify important features with higher accuracy.
- The proposed methods and algorithms are tested only on Parkinson's data but it is not limited its application on other fields. Various sectors such as banking, healthcare, genetics, climatology, marketing, e-commerce, and network traffic, which collect substantial amounts of data, can be used to demonstrate the efficacy of this model.
- Even the innovative methods, and algorithms reduce the execution time, especially for datasets with medium to high dimensions, it is critical to carefully consider the limitations of BVPL that result from the large number of steps in VPL and fitness evaluations required in most of the phases of BVPL which is strongly affected by the number of features of the input dataset. Hence, it is advisable to explore alternative combinations of feature selection, and feature importance methods, in conjunction with BVPL to reduce the largest execution time.

Thesis contributions

1. Analysis of the wide usage of metaheuristic optimization algorithms in feature selection for data processing combined with machine learning methods, with a special focus on predicting Parkinson's.
2. A comparative analysis for evaluating different feature selection methods (filter and wrapper) on predicting Parkinson's evaluating the subsets using three classification machine learning algorithms, and considering optimizing their parameters by a generalized Simulated Annealing heuristic algorithm.
3. Proposed novel and effective Binary Volleyball Premier League algorithm in feature selection which predicts with a higher accuracy Parkinson's, and a faster convergence speed compared to most other metaheuristic optimization algorithms.
4. Proposed integration of an "opposition-based learning" technique in the Binary Volleyball Premier League algorithm that improves its exploration abilities, and effectivity in predicting Parkinson's with a higher accuracy.
5. A new proposed hybrid metaheuristic of Binary Volleyball Premier League algorithm and Antlion Optimizer algorithm which aims to search for new optimal solution, and to improve the exploitation of Binary Volleyball Premier League algorithm considering BALO advantages. The hybrid metaheuristic improves the predictability of Parkinson's, and contributes in a more effective Binary Volleyball Premier League metaheuristic algorithm.
6. Proposed procedure to decrease the execution time of the proposed hybrid metaheuristic Binary Volleyball Premier League algorithm and Antlion Optimizer named "occurrence list" that improves its efficiency by avoiding redundant calculations of the fitness function.
7. Proposed efficient method to reduce the dimensionality of the data and to select the most relevant features by incorporating two algorithms: a feature ranking one based on cosine similarity, and the hybrid metaheuristic Binary Volleyball Premier League algorithm and Antlion optimizer algorithm in a most efficient time.

List of publications related to the thesis

1. Naka, E. K., Guliashki V. G. (2021), “Optimization Techniques in Data Management: A Survey”, 7th International Conference on Computing and Data Engineering (ICCDE2021) (ACM Digital Library), January 15-17, 2021, Phuket, Thailand, pp. 8-13, ISBN:9781450388450; doi: 10.1145/3456172.3456214.
2. Naka, K. E., Guliashki V. G., Marinova G. I., (2021) “A Comparative Analysis of Different Feature Selection Methods on Parkinson Data”, 2021 15th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS 2021), (IEEE, Scopus), October 20-22, 2021, Niš, Serbia, pp. 366-371, doi:10.1109/TELSIKS52058.2021.9606398.
3. Naka E., Guliashki V., (2022), “B-VPL: “A Binary Volleyball Premier League optimization algorithm for Feature Selection”, 2022 29-th International Conference on Systems, Signals and Image Processing (IWSSIP 2022), (IEEE, Scopus), June 01 - 03, 2022, Sofia, Bulgaria, pp. 1-4, doi: 10.1109/IWSSIP55020.2022.9854424.
4. Naka K. E., “Review of Metaheuristic Algorithms in Feature Selection based on Parkinson Disease”, 2023 24th International Conference on Control Systems and Computer Science (CSCS), (IEEE, Scopus), Bucharest, Romania, 24-26 May 2023, pp. 221-228, doi: 10.1109/CSCS59211.2023.00042.
5. Naka E., “A Competitive Parkinson-Based Binary Volleyball Premier League Metaheuristic Algorithm for Feature Selection” Cybernetics Information Technologies, vol.23, no. 4 (Nov 2023), SJR 2022 (0.46) Q2, IF 2022 (1.2), Print ISSN: 1311-9702; Online ISSN: 1314-4081, doi:10.2478/cait-2023-0038
6. Naka E., “An efficient hybrid volleyball premier league and antlion metaheuristic algorithm for feature selection”, 2023, 8th IEEE International Conference “Big Data, Knowledge and Control Systems Engineering” - BdKCSE’2023, (IEEE, Scopus), 02-03.11.2023, Sofia, Bulgaria, pp. 1-8, doi: 10.1109/BdKCSE59280.2023.10339732
7. Naka E., "A feature importance method based on cosine similarity and metaheuristic algorithm," in *IEEE International Conference on Artificial Intelligence in Engineering and Technology*, Kota Kinabalu, Malaysia, 2024 (accepted to be presented on 26-28 August 2024) Status: to appear, Indexed in: IEEE Xplore

Citations

- Naka, E. K., Guliashki V. G. (2021), "Optimization Techniques in Data Management: A Survey", 7th International Conference on Computing and Data Engineering (ICCDE2021) (ACM Digital Library), January 15-17, 2021, Phuket, Thailand, pp. 8-13, ISBN:9781450388450; doi: 10.1145/3456172.3456214.

Cited by:

1. Borissova, D., Danev, V., Garvanova, M., Garvanov, I., Yoshinov, R. (2022). Key Indicators to Measure Student Performance in IoT and Their Teamwork Ability. In: Auer, M.E., Tsiatsos, T. (eds) New Realities, Mobile Systems and Applications. IMCL 2021. Lecture Notes in Networks and Systems, vol 411. Springer, Cham. https://doi.org/10.1007/978-3-030-96296-8_64
2. Balabanov T. (2021). Solving Multi-Objective Problems by Means of Single Objective Solver. Problems of Engineering Cybernetics and Robotics, vol. 76, pp. 63-70, p-ISSN: 2738-7356; e-ISSN: 2738-7364 https://doi.org/10.7546/PECR.76.21.05
3. Shah, S. T. U. (2024). Optimizing Data Warehouse Implementation on Azure: A Comparative Analysis of Efficient Data Warehousing Strategies on Azure, Master Thesis, VAMK University of Applied Sciences, Vaasa, Finland.
4. Виктор Кънчев Данев (2023). ПРОЕКТИРАНЕ НА "УМНИ КЪЩИ" ПОД ОТВОРЕНА СИСТЕМА OPENHAB, ДИСЕРТАЦИЯ, ИНСТИТУТ ПО ИНФОРМАЦИОННИ И КОМУНИКАЦИОННИ ТЕХНОЛОГИИ, БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ, София, България.

- Naka, K. E., Guliashki V. G., Marinova G. I., (2021) "A Comparative Analysis of Different Feature Selection Methods on Parkinson Data", 2021 15th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS 2021), (IEEE, Scopus), October 20-22, 2021, Niš, Serbia, pp. 366-371, doi:10.1109/TELSIKS52058.2021.9606398.

Cited by:

1. U. H. Jaid and A. K. Abdulhassan, "Fuzzy-Based Ensemble Feature Selection for Automated Estimation of Speaker Height and Age Using Vocal Characteristics," in *IEEE Access*, vol. 11, pp. 77895-77905, 2023, doi: 10.1109/ACCESS.2023.3298697
2. K. Stoyanova and T. Balabanov, "A combination of Broyden-Fletcher-Goldfarb-Shanno (BFGS) and bisection method for solving portfolio optimization problems," *2022 International Conference on Engineering and Emerging Technologies (ICEET)*, Kuala Lumpur, Malaysia, 2022, pp. 1-3, doi: 10.1109/ICEET56468.2022.10007369
3. J. Chen, H. Du, Y. Liu and D. Li, "Solving the Algebraic Loop Problem in FMI Specification," *2023 9th International Conference on Mechanical and Electronics Engineering (ICMEE)*, Xi'an, China, 2023, pp. 162-167, doi: 10.1109/ICMEE59781.2023.10525405.

- Naka E., Guliashki V., (2022), "B-VPL: "A Binary Volleyball Premier League optimization algorithm for Feature Selection", 2022 29-th International Conference on Systems, Signals and Image Processing (IWSSIP 2022), (IEEE, Scopus), June 01 - 03, 2022, Sofia, Bulgaria, pp. 1-4, doi: 10.1109/IWSSIP55020.2022.9854424.

Cited by:

1. A. Gjecka and M. Fetaji, "Literature Review On Metaheuristics Techniques In The Health Care Industry," *2023 12th Mediterranean Conference on Embedded Computing (MECO)*, Budva, Montenegro, 2023, pp. 1-8, doi: 10.1109/MECO58584.2023.10155079.
2. Mansourah Aljohani, Yousry AbdulAzeem, Hossam Magdy Balaha, Mahmoud Badawy, Mostafa A Elhosseini, Advancing feature ranking with hybrid feature ranking weighted majority model: a weighted majority voting strategy enhanced by the Harris hawks optimizer, *Journal of Computational Design and Engineering*, Volume 11, Issue 3, June 2024, Pages 308–325, https://doi.org/10.1093/jcde/qwae051

Declaration of Originality

I declare that

1. This dissertation contains original research results obtained by me with the support and assistance of my supervisor. The presented dissertation work has been prepared by me.

2. Results that have been obtained, described and/or published by other scientists are duly and extensively cited in the bibliography. In the dissertation, no foreign texts are used directly or indirectly or, if parts of them are used, they are referred to / quoted and no part of my dissertation infringes the copyright of institution or person.

3. No part of my dissertation work has been presented in this form in the same or in another university, educational and/or scientific institution for the award of educational or scientific degree.

In the event of a discrepancy with the circumstances declared by me according to points 1, 2 and 3 of this declaration, I bear responsibility in accordance with the law and normative documents of IICT.

Signature: ……..

Bibliography

[1] B. Marr, "How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read," Bernard Marr & Co., [Online]. Available: https://bernardmarr.com/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/.

[2] M. Amstrong, "Global Data Creation is About to Explode," Statista Digital Economy Compass 2019, 16 April 2019. [Online]. Available: https://www.statista.com/chart/17727/global-data-creation-forecasts/. [Accessed 28 May 2024].

[3] S. R. Department, "Per capita health expenditure in selected countries in 2022," Statista, 22 May 2024. [Online]. Available: https://www.statista.com/statistics/236541/per-capita-health-expenditure-by-country/. [Accessed 28 May 2024].

[4] J. Degenhard, "Per capita consumer spending on healthcare worldwide from 2014 to 2029," Statista, 30 January 2024. [Online]. Available: https://www.statista.com/forecasts/1161416/healthcare-consumer-spending-per-capita-forecast-in-the-world. [Accessed 30 January 2024].

[5] G. 2. N. S. D. Collaborators, "Global, regional, and national burden of disorders affecting the nervous system, 1990–2021: a systematic analysis for the Global Burden of Disease Study 2021," vol. 23, no. 4, pp. 344-381, 2024.

[6] Statista Research Department, "Parkinson's disease: projected worldwide increase in prevalence," 01 October 2010. [Online]. Available: https://www.statista.com/statistics/215459/projected-worldwide-increase-in-prevalence-of-parkinsons-diseas/. [Accessed 28 May 2024].

[7] T. Dokeroglu, A. Deniz and H. E. Kiziloz, "A comprehensive survey on recent metaheuristics for feature selection," Neurocomputing, vol. 494, pp. 269-296, 2022.

[8] C. Bin, H. Jiarong and W. Yadong, "The minimum feature subset selection problem," *Journal of Computer Science and Technology,* vol. 12, pp. 145-153, 1997.

[9] D. Molina, J. Poyatos, J. Del Ser, S. García, A. Hussain and F. Herrera, "Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration Versus Algorithmic Behavior, Critical Analysis Recommendations," *Cognitive Computation,* vol. 12, pp. 897-939, 2020.

[10] K. Hussain, M. N. M. Salleh, S. Cheng and Y. Shi, "Metaheuristic research: a comprehensive survey," *Artificial Intelligence Review,* vol. 52, p. 2191–2233, 2019.

[11] A. E. Ezugwu, A. K. Shukla, R. Nath, A. Akinyelu, J. Agushaka, H. Chiroma and P. K. Muhuri, "Metaheuristics: a comprehensive overview and classification along with bibliometric analysis," *Artificial Intelligence Review,* vol. 54, pp. 4237-4316, 2021.

[12] K. Rajwar, K. Deep and S. Das, "An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges," *Artificial Intelligence Review,* vol. 56, pp. 13187-13257, 2023.

[13] L. Velasco, H. Guerrero and A. Hospitaler, "A Literature Review and Critical Analysis of Metaheuristics Recently Developed," *Archives of Computational Methods in Engineering,* vol. 31, pp. 125-146, 2024.

[14] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE*

Transactions on Evolutionary Computation*,* vol. 1, no. 1, pp. 67-82, 1997.

[15] E. K. Naka and V. G. Guliashki, "Optimization Techniques in Data Management: A Survey," in *ICCDE '21: Proceedings of the 2021 7th International Conference on Computing and Data Engineering*, Phuket Thailand, 2021.

[16] E. K. Naka, V. G. Guliashki and G. I. Marinova, "A Comparative Analysis of Different Feature Selection Methods on a Parkinson Data," in *2021 15th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS)*, Nis, Serbia, 2021.

[17] E. Naka and V. Guliashki, "B-VPL: A Binary Volleyball Premier League optimization algorithm for Feature Selection," in *2022 29th International Conference on Systems, Signals and Image Processing (IWSSIP)*, Sofia, Bulgaria, 2022.

[18] E. K. Naka, "Review of Metaheuristic Algorithms in Feature Selection based on Parkinson Disease," in *2023 24th International Conference on Control Systems and Computer Science (CSCS)*, Bucharest, Romania, 2023.

[19] E. Naka, "An Efficient Hybrid Volleyball Premier League and Antlion Metaheuristic Algorithm for Feature Selection," in *2023 International Conference on Big Data, Knowledge and Control Systems Engineering (BdKCSE)*, Sofia, Bulgaria,, 2023.

[20] E. Naka, "A Competitive Parkinson-Based Binary Volleyball Premier League Metaheuristic Algorithm for Feature Selection," *Cybernetics and Information Technologies,* vol. 23, no. 4, pp. 91-109, 2023.

[21] E. Naka, "A Feature Importance Method based on Cosine Similarity and Metaheursitic Algorithm," in *IEEE International Conference on Artificial Intelligence in Engineering and Technology*, Kota Kinabalu, Malaysia, 2024.

[22] G. D. M. Community, DAMA-DMBOK - Data Management Body of Knowledge, New Jersey, USA: Technics Publications, 2017.

[23] Ishwarappa and J. Anuradha, "A Brief Introduction on Big Data 5Vs Characteristics and Hadoop Technology," *Procedia Computer* Science*,* vol. 48, pp. 319-324, 2015.

[24] K. Krishnan, "Introducing Big Data Technologies," in *Data Warehousing in the Age of Big Data*, Waltham, MA, Elsevier, 2013, pp. 45-99.

[25] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research and Development,* vol. 3, no. 3, pp. 210-229, 1959.

[26] R. A. Sarker and C. S. Newton, Optimization Modelling, A Practical Approach, CRC Press, 2008, pp. 17-18.

[27] X.-S. Yang, Engineering Optimization - An Introduction with Metaheuristic Applications, Hoboken, New Jersey: John Wiley & Sons, Inc.,, 2010, pp. 17-21.

[28] Y. E. Ioannidis, "Query optimization," *ACM Computing Surveys,* vol. 28, no. 1, pp. 121-123, 1996.

[29] R. Gomath and D. Sharmila, "A Novel Adaptive Cuckoo Search for Optimal Query Plan Generation," *The Scientific World Journal,* vol. 2014, pp. 1-7, 2014.

[30] M. Joshi and P. R. Srivastava, "Query Optimization: An Intelligent Hybrid Approach using Cuckoo and Tabu Search," *International Journal of Intelligent Information Technologies (IJIIT), IGI Global,,* vol. 9, no. 1, pp. 40-55, 2013.

[31] L. Golshanara, S. M. T. R. Rankoohi and H. Shah-Hosseini , "A multi-colony ant algorithm for optimizing join queries in distributed database systems," *Knowledge and Information*

System, vol. 39, pp. 175-206, 2014.

[32] M. Alamery, A. Faraahi, H. Haj Seyyed Javadi, S. Nourossana and H. Erfani, "Multi-Join Query Optimization Using the Bees Algorithm," in *Distributed Computing and Artificial Intelligence. Advances in Intelligent and Soft Computing*, vol. 79, Berlin, Springer, 2010, p. 449–457.

[33] H. Azgomi and M. K. Sohrabi, "A novel coral reefs optimization algorithm for materialized view," *Applied Intelligence,* vol. 49, p. 3965–3989, 2019.

[34] J. Loureiro and O. Belo , "A Discrete Particle Swarm Algorithm," in *In Proceedings of the Eighth International Conference on Enterprise Information Systems - DISI*, 2006.

[35] M. Ghobaei-Arani and A. Shahidinejad, "An efficient resource provisioning approach for analyzing cloud workloads - a metaheuristic-vased clustering approach," *The Journal of Supercomputing,* vol. 77, pp. 711-750, 2021.

[36] A. B. Mathew, "Data allocation optimization for query processing in graph databases using Lucene," *Computers and Electrical Engineering,* vol. 70, pp. 1019-1033, 2018.

[37] M. Woźniak, "Positioning Traffic in NoSQL Database Systems by the Use of Particle Swarm Algorithm," in *Proceedings of the XV Workshop "Dagli Oggetti agli Agenti" (WOA 2014)*, Catania, Italy, 2014.

[38] M. Woźniak, M. Gabryel, R. K. .Nowicki and B. A. Nowak , "An Application of Firefly Algorithm to Position Traffic in NoSQL Database Systems," in *Knowledge, Information and Creativity Support Systems. Advances in Intelligent Systems and Computing,*, vol. 416, Sprimger, Cham, 2016.

[39] Z.-C. Lin, "How Can Machine Learning and Optimization Help Each Other Better?," *Journal of the Operations Research Society of China,* vol. 8, pp. 341-351, 2020.

[40] S. Sun, Z. Cao, H. Zhu and J. Zhao, "A Survey of Optimization Methods From a Machine Learning Perspective," *IEEE Transactions on Cybernetics,* vol. 50, no. 8, pp. 3668 - 3681, 2020.

[41] E.-G. Talbi, "Machine Learning into Metaheuristics: A Survey and Taxonomy," *ACM Computing Surveys,* vol. 54, no. 6, pp. 1-31, 2021.

[42] L. Calvet, J. de Armas, D. Masip and A. A. Juan, "Learnheuristics: hybridizing metaheuristics with machine learning for optimization with dynamic inputs," *Open Mathematics,* vol. 15, no. 1, pp. 261-280, 2017.

[43] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan and E.-G. Talbi, "Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art," *European Journal of Operational Research,* vol. 296, no. 2, pp. 393-422, 2022.

[44] K. P. Bennett and E. Parrado-Hernandez, "The Interplay of Optimization and Machine Learning Research," *Journal of Machine* Learning *Research,* vol. 7, no. 46, pp. 1265-1281, 2006.

[45] A. Yaqoob, R. M. Aziz, N. K. Verma, P. Lalwani, A. Makrariya and P. Kumar, "A Review on Nature-Inspired Algorithms for Cancer Disease Prediction and Classification," *Mathematics,* vol. 11, no. 5, pp. 1-32, 2023.

[46] W. Jia, M. Sun, J. Lian and S. Hou, "Feature dimensionality reduction: a review," *Complex & Intelligent Systems,* vol. 8, p. 2663–2693, 2022.

[47] H. Liu and H. Motoda, "Perspectives of Feature Selection," in *Feature Selection for*

Knowledge Discovery and Data Mining . The Springer International Series in Engineering and Computer Science, vol. 454, Boston, MA, Springer, 1998, pp. 17-41.

[48] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *EEE Transactions on Knowledge and Data Engineering,* vol. 17, no. 4, pp. 491-502, 2005.

[49] N. Pudjihartono, T. Fadason, A. W. Kempa-Liehr and J. M. O'Sullivan, "A Review of Feature Selection Methods for Machine Learning-Based Disease Risk Prediction," *Frontiers in Bioinformatics,* vol. 2, pp. 1-17, 2022.

[50] B. Venkatesh and J. Anuradha, "A Review of Feature Selection and Its Methods," *Cybernetics and Information Technologies,* vol. 19, no. 1, pp. 3-26, 2019.

[51] B. Remeseiro and V. Bolon-Canedo, "A Review of Feature Selection Methods in Medical Applications.," *Computers in Biology and Medicine,* vol. 112, pp. 1-35, 2019.

[52] J. Li, K. Cheng, . S. Wang, F. Morstatter, R. P. Trevino, J. Tang and H. Liu, "Feature Selection: A Data Perspective," *ACM Computing Survey,* vol. 9, no. 4, pp. 1-45, 2017.

[53] Y. Li, T. Li and H. Liu, "Recent advances in feature selection and its applications," *Knowledge and Information Systems,* vol. 53, pp. 551-577, 2017.

[54] M. F. Kabir, T. Chen and S. A. Ludwig, "A performance analysis of dimensionality reduction algorithms in machine learning models for cancer prediction," *Healthcare Analytics,* vol. 3, 2023.

[55] J. Chen, L. Q. R. Ooi, T. W. K. Tan, S. Zhang, J. Li, C. L. Asplund, S. B. Eickhoff, D. Bzdok, A. J. Holmes and B. T. T. Yeo, "Relationship between prediction accuracy and feature importance reliability: An empirical and theoretical study," *Neuroimage,* vol. 274, pp. 1-13, 2023.

[56] M. Kuhn and K. Johnson, Feature Engineering and Selection: A Practical Approach for Predictive Models, Chapman & Hall/CRC, 2019.

[57] L. Breiman, "Random Forests," *MAchine Learning,* vol. 45, no. 1, 2001.

[58] I. Guyon, J. Weston, S. Barnhill and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning,* vol. 46, no. 1, p. 389–422, 2002.

[59] K. Kira and L. A. Rendell, "A Practical Approach to Feature Selection," in *In Proceedings of the Ninth International Workshop on Machine Learning (ML '92:)*, San Francisco, CA, USA, 1992.

[60] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics,* vol. 29, no. 5, pp. 1189-1232, 2001.

[61] A. Suebsing and N. Hiransakolwong, "Feature Selection Using Euclidean Distance and Cosine Similarity for Intrusion Detection Model," in *2009 First Asian Conference on Intelligent Information and Database Systems*, Dong hoi, Vietnam, 2009.

[62] V. K. Dubey and A. K. Saxena, "A Cosine-Similarity Mutual-Information Approach for Feature Selection on High Dimensional Datasets," *Journal of Information Technology Research (JITR),* vol. 10, no. 1, pp. 15-28, 2017.

[63] K. Zhang, Y. Liu, F. Mei, G. Sun and J. Jin, "IBGJO: Improved Binary Golden Jackal Optimization with Chaotic Tent Map and Cosine Similarity for Feature Selection," *Entropy,* vol. 25, no. 8, 2023.

[64] S. Radouche and C. Leghris, "New Network Selection Algorithm Based on Cosine Similarity Distance and PSO in Heterogeneous Wireless Networks," *Journal of Computer Networks and*

Communications, vol. 2021, pp. 1-11, 2021.

[65] W.-l. Xiang, Y.-z. Li, R.-c. He, M.-x. Gao and M.-g. An, "A novel artificial bee colony algorithm based on the cosine similarity," *Computers & Industrial Engineering,* vol. 115, pp. 54-68, 2018.

[66] "Parkinson's Statistics," Parkinson's Europe, [Online]. Available: https://parkinsonseurope.org/facts-and-figures/statistics/. [Accessed 17 April 2024].

[67] "Statistics," Parkinson's Foundation, [Online]. Available: https://www.parkinson.org/understanding-parkinsons/statistics. [Accessed 17 April 2024].

[68] Z. Ou, J. Pan, S. Tang, D. Duan, D. Yu, H. Nong and Z. Wang, "Global Trends in the Incidence, Prevalence, and Years Lived With Disability of Parkinson's Disease in 204 Countries/Territories From 1990 to 2019," *Frontiers in Public Health,* vol. 9, pp. 1-16, 2021.

[69] E. Tolosa, A. Garrido, S. W. Scholz and W. Poewe, "Challenges in the diagnosis of Parkinson's disease," *The Lancet Neurology,* vol. 20, no. 5, pp. 385-397, 2021.

[70] M. J. Armstrong and M. S. Okun, "Diagnosis and Treatment of Parkinson Disease - Areview," *JAMA,* vol. 323, no. 6, pp. 548-560, 2020.

[71] J. Mei, C. Desrosiers and J. Frasnelli, "Machine Learning for the Diagnosis of Parkinson's Disease: A Review of Literature," *Frontiers in Aging Neuroscience,* vol. 13, pp. 1-41, 2021.

[72] M. Sharma and P. Kaur, "A Comprehensive Analysis of Nature-Inspired Meta-Heuristic Techniques for Feature Selection Problem," *Archives of Computational Methods in Engineering,* vol. 28, pp. 1103-1127, 2021.

[73] M. A. Little, P. E. McSharry, E. J. Hunter, J. Spielman and L. O. Ramig, "Suitability of dysphonia measurements for telemonitoring of Parkinson's disease," *IEEE Transactions on Biomedical Engineering,* vol. 56, no. 4, pp. 015 - 1022, 2009.

[74] M. Little, "Oxford Parkinson's Disease Detection Dataset," UC Irvine Machine Learning Repository, 25 June 2008. [Online]. Available: https://archive.ics.uci.edu/dataset/174/parkinsons. [Accessed 01 March 2021].

[75] C. R. Pereira, D. R. Pereira, F. A. Silva, J. P. Masieiro, S. A. Weber, C. Hook and J. P. Papa, "A new computer vision-based approach to aid the diagnosis of Parkinson's disease," *Computer Methods and Programs in Biomedicine,* vol. 136, pp. 79-88, 2016.

[76] "HandPD dataset; NewHandPD dataset," [Online]. Available: https://wwwp.fc.unesp.br/~papa/pub/datasets/Handpd/. [Accessed 4 August 2022].

[77] C. R. Pereira, S. A. T. Weber, C. Hook, G. . H. Rosa and J. P. Papa, "Deep Learning-Aided Parkinson's Disease Diagnosis from Handwritten Dynamics," in *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, Sao Paulo, Brazil, 2016.

[78] J. Hlavnička, . R. Čmejla, T. Tykalová, K. Šonka, E. Růžička and J. Rusz, "Automated analysis of connected speech reveals early biomarkers of Parkinson's disease in patients with rapid eye movement sleep behaviour disorder," *Scientific Reports,* vol. 7, no. 12, 2017.

[79] "Early biomarkers of Parkinson's disease based on natural connected speech," UC Irvine Machine Learning Repository, [Online]. Available: https://archive.ics.uci.edu/dataset/392/early+biomarkers+of+parkinson+s+disease+based+on+natural+connected+speech. [Accessed 26 March 2022].

[80] C. O. Sakar, G. Serbes, A. Gunduz, H. C. Tunc, H. Nizam, B. E. Sakar, M. Tutuncu, T. Aydin, M. E. Isenkul and H. Apaydin, "A comparative analysis of speech signal processing algorithms for Parkinson's disease classification and the use of the tunable Q-factor wavelet

transform," *Applied Soft Computing Journal,* vol. 74, 2019.

[81] "Parkinson's Disease Classification," UC Irvine Machine Learning Repository. [Online]. [Accessed 14 August 2022].

[82] A. Tsanas, M. A. Little, P. E. McSharry and L. O. Ramig, "Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests," *IEEE Transactions on Biomedical Engineering,* vol. 57, no. 4, pp. 884-893, 2010.

[83] "Parkinson Dataset with replicated acoustic features," [Online]. Available: https://archive.ics.uci.edu/dataset/489/parkinson+dataset+with+replicated+acoustic+features. [Accessed 26 March 2022].

[84] B. E. Sakar, M. E. Isenkul, C. O. Sakar, A. Sertbas, F. Gurgen, S. Delil, H. Apaydin and O. Kursun, "Collection and Analysis of a Parkinson Speech Dataset with Multiple Types of Sound Recordings," *IEEE Journal of Biomedical and Health Informatics,* vol. 17, no. 4, pp. 828 - 834, 2013.

[85] "Parkinson's Speech with Multiple Types of Sound Recordings," [Online]. Available: https://archive.ics.uci.edu/dataset/301/parkinson+speech+dataset+with+multiple+types+of+sound+recordings. [Accessed 26 March 2022].

[86] Parkinson's Progression Markers Initiative, "Access Data," [Online]. Available: https://www.ppmi-info.org/access-data-specimens/download-data.

[87] K. Sörensen and F. W. Glover, "Metaheuristics," in *Encyclopedia of Operations Research and Management Science*, Springer, Boston, MA, 2013, p. 960–970.

[88] M. Sevaux, F. Glover and K. Sörensen, "A History of Metaheuristics," in *Handbook of Heuristics*, Springer, Cham, 2018, pp. 791-808.

[89] M. Rafael, M. Sevaux and K. Sörensen, "50 years of metaheuristics," *European Journal of Operational Research,* 2024.

[90] S. Bandaru and K. Deb, "Metaheuristic Techniques," in *Decision Sciences: Theory and Practice*, CRC Press, Taylor & Francis Group, pp. 693-750.

[91] P. Agrawal, H. F. Abutarboush, T. Ganesh and A. W. Mohamed, "Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019)," *IEEE Access,* vol. 9, pp. 26766-26791, 2021.

[92] S. Darvishpoor , A. Darvishpour, M. Escarcega and H. Mostafa, "Nature-Inspired Algorithms from Oceans to Space: A Comprehensive Review of Heuristic and Meta-Heuristic Optimization Algorithms and Their Potential Applications in Drones," *Drones,* vol. 7, no. 7, 2023.

[93] T. Dokeroglu, T. Kucukyilmaz and E.-G. Talbi, "Hyper-heuristics: A survey and taxonomy," *Computers & Industrial Engineering,* vol. 187, pp. 1-18, 2024.

[94] E. Wari and W. Zhu, "A survey on metaheuristics for optimization in food manufacturing industry," *Applied Soft Computing,* vol. 46, pp. 328-343, 2016.

[95] E. Cuevas, F. Fausto and A. Gonzalez, "Metaheuristics and Swarm Methods: A Discussion on Their Performance and Applications," in *New Advancements in Swarm Algorithms: Operators and Applications*, vol. 160, Intelligent Systems Reference Library, Springer, Cham, 2020, pp. 43-67.

[96] O. M. Alyasiri, Y.-N. Cheah, A. K. Abasi and O. M. Al-Janabi, "Wrapper and Hybrid Feature Selection Methods Using Metaheuristic Algorithms for English Text Classification: A Systematic Review," *IEEE Access,* vol. 10, pp. 39833-39852, 2022.

[97] E. O. Abiodun, A. Alabdulatif, O. I. Abiodun, M. Alawida, A. Alabdulatif and R. S. Alkhawaldeh, "A systematic review of emerging feature selection optimization methods for optimal text classification: the present state and prospective opportunities," *Neural Computing and Applications,* vol. 33, pp. 15091-15118, 2021.

[98] A. Soler-Dominguez, A. A. Juan and R. Kizys, "A Survey on Financial Applications of Metaheuristics," *ACM* Computing *Surveys,* vol. 50, no. 1, pp. 1-23, 2017.

[99] A. K. Shukla, . D. Tripathi, B. R. Reddy and D. Chandramohan , "A study on metaheuristics approaches for gene selection in microarray data: algorithms, applications and open challenges," *volutionary Intelligence,* vol. 13, pp. 309--329, 2020.

[100] H. Rezk, A. G. Olabi, T. Wilberforce and E. T. Sayed, "A Comprehensive Review and Application of Metaheuristics in Solving the Optimal Parameter Identification Problems," *Sustainability,* vol. 15, no. 7, 2023.

[101] K. Sörensen, "Metaheuristics—the metaphor exposed," *International Transactions In Operational Resaearch,* vol. 22, no. 1, pp. 3-18, 2013.

[102] M. Azizi, M. B. Shishehgarkhaneh, M. Basiri and R. C. Moehler, "Squid Game Optimizer (SGO): a novel metaheuristic algorithm," *Scientific Reports,* vol. 13, 2023.

[103] S. Ferahtia, H. Azeddine, H. Rezk, A. Djerioui, M. Machmoum, S. Motahhir and M. Ait-Ahmed, "Red-tailed hawk algorithm for numerical optimization and real-world problems," *Scientific Reports,* vol. 13, 2023.

[104] M. Azizi, U. Aickelin, H. A. Khorshidi and M. B. Shishehgarkhaneh , "Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization," *Scientific Reports,* vol. 13, no. 226, 2023.

[105] E. Trojovská, M. Dehghani and V. Leiva, "Drawer Algorithm: A New Metaheuristic Approach for Solving Optimization Problems in Engineering," *Biomimetics,* vol. 8, no. 2, 2023.

[106] H. Jia, H. Rao, C. Wen and S. Mirjalili , "Crayfish optimization algorithm," *Artificial Intelligence Review,* vol. 56, pp. 1919-1979, 2023.

[107] S. Mohapatra and P. Mohapatra, "American zebra optimization algorithm for global optimization problems," *Scientific Reports,* vol. 13, no. 5211, 2023.

[108] P. Trojovský and M. Dehghani , "A new bio-inspired metaheuristic algorithm for solving optimization problems based on walruses behavior," *Scientific Reports,* vol. 13, no. 8775, 2023.

[109] I. Matoušová, P. Trojovský, M. Dehghani, E. Trojovská and J. Kostra , "Mother optimization algorithm: a new human-based metaheuristic approach for solving engineering optimization," *Scientific Reports,* vol. 13, no. 10312, 2023.

[110] M. Dehghani , Z. Montazeri , . E. Trojovská and P. Trojovský, "Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowledge-Based Systems,* vol. 259, 2023.

[111] A. H. Rabie, . A. I. Saleh and N. A. Mansour , "Red piranha optimization (RPO): a natural inspired meta-heuristic algorithm for solving complex optimization problems," *Journal of Ambient Intelligence and Humanized Computing,* vol. 14, pp. 7621-7648, 2023.

[112] Z. Montazeri, T. Niknam, J. Aghaei, O. P. Malik, M. Dehghani and G. Dhiman, "Golf Optimization Algorithm: A New Game-Based Metaheuristic Algorithm and Its Application to Energy Commitment Problem Considering Resilience," *Biomimetics,* vol. 8, no. 5, 2023.

[113] A. Seyyedabbasi and F. Kiani, "Sand Cat swarm optimization: a nature-inspired algorithm to solve global optimization problems," *Engineering with Computers,* vol. 39, pp. 2627-2651, 2023.

[114] H. Givi, M. Dehghani and Š. Hubálovský, "Red Panda Optimization Algorithm: An Effective Bio-Inspired Metaheuristic Algorithm for Solving Engineering Optimization Problems," *IEEE Access,* vol. 11, pp. 57203 - 57227, 2023.

[115] M. Dehghani and P. Trojovský, "Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems," *Frontiers in Mechanical Engineering,* vol. 8, 2023.

[116] S. O. Oladejo, S. O. Ekwe, L. A. Akinyemi and S. A. Mirjalili, "The Deep Sleep Optimizer: A Human-Based Metaheuristic Approach," *IEEE,* vol. 11, pp. 83639 - 83665, 2023.

[117] M. Dehghani, G. Bektemyssova, Z. Montazeri, G. Shaikemelev, O. P. Malik and G. Dhiman, "Lyrebird Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 8, no. 6, 2023.

[118] J. Bai, Y. Li, M. Zheng, S. Khatir, B. Benaissa, L. Abualigah and M. A. Wahab, "A Sinh Cosh optimizer," *Knowledge-Based Systems,* vol. 282, 2023.

[119] P. Trojovský and M. Dehghani, "Migration Algorithm: A New Human-Based Metaheuristic Approach for Solving Optimization Problems," *Computer Modeling in Engineering & Sciences,* vol. 137, no. 2, pp. 1695-1730, 2023.

[120] H. Givi and M. Hubalovska, "Skill Optimization Algorithm: A New Human-Based Metaheuristic Technique," *Computers, Materials & Continua,* vol. 74, no. 1, pp. 179-202, 2023.

[121] K. Zolf, "Gold rush optimizer : a new population-based metaheuristic algorithm," *Operations Research and Decisions,* vol. 33, no. 1, pp. 113-150, 2023.

[122] M. Dehghani, E. Trojovská, P. Trojovský and O. P. Malik, "OOBO: A New Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 8, no. 6, pp. 1-48, 202.

[123] M. Azizi, S. Talatahari and A. H. Gandomi, "Fire Hawk Optimizer: a novel metaheuristic algorithm," *Artificial Intelligence Review,* vol. 56, pp. 287-363, 2023.

[124] A. A. Abdelhamid, S. K. Towfek, N. Khodadadi, A. A. Alhussan, D. S. Khafaga, M. M. Eid and A. Ibrahim, "Waterwheel Plant Algorithm: A Novel Metaheuristic Optimization Method," *Processes,* vol. 11, no. 5, 2023.

[125] J. O. Agushaka, A. E. Ezugwu and L. Abualigah, "Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer," *Neural Computing and Applications,* vol. 35, pp. 4099-4131, 2023.

[126] M. Abdel-Basset, R. Mohamed, M. Jameel and M. Abouhawwash, "Spider wasp optimizer: a novel meta-heuristic optimization algorithm," *Artificial Intelligence Review,* vol. 56, pp. 11675-11738, 2023.

[127] L. Deng and S. Liu, "Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design," *Expert Systems with Applications,* vol. 225, 2023.

[128] M. Dehghani, P. Trojovský and O. P. Malik, "Green Anaconda Optimization: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 8, no. 1, pp. 1-60, 2023.

[129] M. Abdel-Basset , R. Mohamed, M. Jameel and M. Abouhawwash, "Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design

problems," *Knowledge-Based Systems,* vol. 262, 2023.

[130] M. Abdel-Basset, D. El-Shahat, M. Jameel and M. Abouhawwash, "Young's double-slit experiment optimizer : A novel metaheuristic optimization algorithm for global and constraint optimization problems," *Computer Methods in Applied Mechanics and Engineering,* vol. 403, no. A, 2023.

[131] D. Li, S. Du, Y. Zhang and M. Zhao, "Dark Forest Algorithm: A Novel Metaheuristic Algorithm for Global Optimization Problems," *Computers, Materials & Continua,* vol. 75, no. 2, pp. 2775-2803, 2023.

[132] S. K. Mohammadi, D. Nazarpour and M. Beiraghi, "A novel metaheuristic algorithm inspired by COVID-19 for real parameter optimization," *Neural Computing and Applications,* vol. 35, pp. 10147-10196, 2023.

[133] X. Zhou, Y. Guo, Y. Yan, Y. Huang and Q. Xue, "Migration Search Algorithm: A Novel Nature-Inspired Metaheuristic Optimization Algorithm," *Journal of Network Intelligence,* vol. 8, no. 2, pp. 324-345, 2023.

[134] C. Min, J. Cui, L. Zhou, Q. Yin and Y. Wang, "Calico Salmon Migration Algorithm: A novel meta-heuristic optimization algorithm," *arXiv:2311.05971,* pp. 1-9, 2023.

[135] P. Trojovský and M. Dehghani, "Subtraction-Average-Based Optimizer: A New Swarm-Inspired Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 8, no. 2, 2023.

[136] M. J. Mahmoodabadi, M. Rasekh and M. Yahyapour, "Tree optimization algorithm (TOA): a novel metaheuristic approach for solving mathematical test functions and engineering problems," *Evolutionary Intelligence,* vol. 16, pp. 1325-1338, 2023.

[137] P. D. Kusuma and m. Kallista, "Quad Tournament Optimizer: A Novel Metaheuristic Based on Tournament Anong Four Strategies," *International Journal of Intelligent Engineering & Systems,* vol. 16, no. 2, pp. 268-278, 2023.

[138] Z. Guan, C. Ren, J. Niu, P. Wang and Y. Shang, "Great Wall Construction Algorithm: A novel meta-heuristic algorithm for engineer problems," *Expert Systems with Applications,* vol. 233, 2023.

[139] S. Zhao, T. Zhang, S. Ma and M. Wang, "Sea-horse optimizer: a novel nature-inspired meta-heuristic for global optimization problems," *Applied Intelligence,* vol. 53, pp. 11833-11860, 2023.

[140] I. Faridmehr, M. L. Nehdi, I. F. Davoudkhani and A. Poolad, "Mountaineering Team-Based Optimization: A Novel Human-Based Metaheuristic Algorithm," *Mathematics,* vol. 11, no. 5, 2023.

[141] M. Abdel-Basset, D. El-Shahat, M. Jameel and M. Abouhawwash, "Exponential distribution optimizer (EDO): a novel math-inspired algorithm for global optimization and engineering problems," *Artificial Intelligence Review,* vol. 56, pp. 9329-9400, 2023.

[142] Q. Zhang, H. Gao, Z.-H. Zhan, J. Li and H. Zhang, "Growth Optimizer: A powerful metaheuristic algorithm for solving continuous and discrete global optimization problems," *Knowledge-Based Systems,* vol. 261, 2023.

[143] C. M. Rahman, "Group learning algorithm: a new metaheuristic algorithm," *Neural Computing and Applications,* vol. 35, pp. 14013-14028, 2023.

[144] E. H. Houssein , D. Oliva, N. A. Samee, N. F. Mahmoud and M. M. Emam, "Liver Cancer Algorithm: A novel bio-inspired optimizer," *Computers in Biology and Medicine,* vol. 165,

2023.

[145] "Al-Biruni Earth Radius (BER) Metaheuristic Search Optimization Algorithm," *Computer Systems Science & Engineering,* vol. 45, no. 2, pp. 1917-1934, 2023.

[146] M. Abdel-Basset, R. Mohamed, S. A. A. Azeem, M. Jameel and M. Abouhawwash, "Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion," *Knowledge-Based Systems,* vol. 268, 2023.

[147] Z. Musa, Z. Ibrahim, M. I. Shapiai and Y. Tsuboi, "Cubature Kalman Optimizer: A Novel Metaheuristic Algorithm for Solving Numerical Optimization Problems," *Journal of Advanced Research in Applied Sciences and Engineering Technology,* vol. 33, no. 1, 2023.

[148] M. Dehghani, Z. Montazeri, G. Bektemyssova, O. P. Malik, G. Dhiman and A. E. M. Ahmed, "Kookaburra Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 8, no. 6, 2023.

[149] P. D. Kusuma and F. C. Hasibuan, "Swarm Magnetic Optimizer: A New Optimizer that Adopts Magnetic Behaviour," *International Journal of Intelligent Engineering & Systems,* vol. 16, no. 4.

[150] M.-Y. Cheng and M. N. Sholeh, "Optical microscope algorithm: A new metaheuristic inspired by microscope magnification for solving engineering optimization problems," *Knowledge-Based Systems,* vol. 279, 2023.

[151] W. Zhao, L. Wang, Z. Zhang , S. Mirjalili, N. Khodadadi and Q. Ge, "Quadratic Interpolation Optimization (QIO): A new optimization algorithm based on generalized quadratic interpolation and its applications to real-world engineering problems," *Computer Methods in Applied Mechanics and Engineering,* vol. 417, no. A, 2023.

[152] I. Mashhad, M. Jalali, M. Yaghoobi and H. Tabatabaee, "Lotus effect optimization algorithm (LEA): a lotus nature-inspired algorithm for engineering design optimization," *The Journal of Supercomputing,* vol. 80, pp. 761-799, 2024.

[153] A. Taheri, K. RahimiZadeh, A. Beheshti, J. Baumbach, R. V. Rao, S. Mirjalili and A. H. Gandomi, "Partial reinforcement optimizer: An evolutionary optimization algorithm," *Expert Systems with Applications,* vol. 238/F, 2024.

[154] O. Al-Baik, S. Alomari, O. Alssayed, S. Gochhait, I. Leonova, U. Dutta, O. P. Malik, Z. Montazeri and M. Dehghani, "Pufferfish Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 9, no. 2, 2024.

[155] M. Abdel-Basset, R. Mohamed and M. Abouhawwash , "Crested Porcupine Optimizer: A new nature-inspired metaheuristic," *Knowledge-Based Systems,* vol. 284, 2024.

[156] M. A. Al-Betar, M. A. Awadallah, M. S. Braik, S. Makhadmeh and I. A. Doush , "Elk herd optimizer: a novel nature-inspired metaheuristic algorithm," *Artificial Intelligence Review,* vol. 57, 2024.

[157] M. Nemati, Y. Zandi and A. S. Agdas, "Application of a novel metaheuristic algorithm inspired by stadium spectators in global optimization problems," *Scientific Reports,* vol. 14, no. 1.

[158] S.-C. Chu, T.-T. Wang, A. R. Yildiz and J.-S. Pan, "Ship Rescue Optimization: A New Metaheuristic Algorithm for solving Engineering Problems," *Journal of Internet Technology,* vol. 25, no. 1, pp. 61-78, 2024.

[159] Z. Tian and M. Gai, "Football team training algorithm: A novel sport-inspired metaheuristic

optimization algorithm for global optimization," *Expert Systems with Applications,* vol. 254, 2024.

[160] E.-S. M. El-kenawy, N. Khodadadi, S. S. Mirjalili, A. A. Abdelhamid, M. M. Eid and A. Ibrahim, "Greylag Goose Optimization: Nature-inspired optimization algorithm," *Expert Systems with Applications,* vol. 238, no. E, 2024.

[161] X. Wang, V. Snášel, S. Mirjalili, J.-S. Pan, L. Kong and H. A. Shehadeh, "Artificial Protozoa Optimizer (APO): A novel bio-inspired metaheuristic algorithm for engineering optimization," *Knowledge-Based Systems,* vol. 295, 2024.

[162] B. Abdollahzadeh, . N. Khodadadi, S. Barshandeh, P. Trojovský, F. S. Gharehchopogh, E.-S. M. El-kenawy, L. Abualigah and S. Mirjalili , "Puma optimizer (PO): a novel metaheuristic optimization algorithm and its application in machine learning," *Cluster Computing,* pp. 5235-5283, 2024.

[163] A.-Q. Tian, F.-F. Liu and H.-X. Lv, "Snow Geese Algorithm: A novel migration-inspired meta-heuristic algorithm for constrained engineering optimization problems," *Applied Mathematical Modelling,* vol. 126, pp. 327-347, 2024.

[164] M. Han, Z. Du, K. F. Yuen, H. Zhu, Y. Li and Q. Yuan, "Walrus optimizer: A novel nature-inspired metaheuristic algorithm," *Expert Systems with Applications,* vol. 239, 2024.

[165] S. Thapliyal and N. Kumar, "Hyperbolic Sine Optimizer: a new metaheuristic algorithm for high performance computing to address computationally intensive tasks," *Cluster Computing,* 2024.

[166] K. M. Hosny, A. M. Khalid, W. Said, M. Elmezain and S. Mirjalili, "A novel metaheuristic based on object-oriented programming concepts for engineering optimization," *Alexandria Engineering Journal,* vol. 98, pp. 221-248, 2024.

[167] M. Ghasemi, M. Zare, A. Zahedi, . M.-A. Akbari, S. Mirjalili and L. Abualigah, "Geyser Inspired Algorithm: A New Geological-inspired Meta-heuristic for Real-parameter and Constrained Engineering Optimization," *Journal of Bionic Engineering,* vol. 21, pp. 374-408, 2024.

[168] R. Kundu, S. Chattopadhyay, S. Nag, M. A. Navarro and D. Oliva, "Prism refraction search: a novel physics-based metaheuristic algorithm," *The Journal of Supercomputing,* vol. 80, pp. 10746-10795, 2024.

[169] L. Ni, Y. Ping, N. Yao, J. Jiao and G. Wang, "Literature Research Optimizer: A New Human-Based Metaheuristic Algorithm for Optimization Problems," *Arabian Journal for Science and Engineering,* 2024.

[170] Z. Li, X. Gao, X. Huang, J. Gao, X. Yang and M.-J. Li, "Tactical unit algorithm: A novel metaheuristic algorithm for optimal loading distribution of chillers in energy optimization," *Applied Thermal Engineering,* vol. 238, 2024.

[171] H. Peraza-Vázquez, A. Peña-Delgado, M. Merino-Treviño, A. B. Morales-Cepeda and N. Sinha , "A novel metaheuristic inspired by horned lizard defense tactics," *Artificial Intelligence Review,* vol. 57, 2024.

[172] S. E. Khouni and T. Menacer , "Nizar optimization algorithm: a novel metaheuristic algorithm for global optimization and engineering applications," *The Journal of Supercomputing,* pp. 3229-3281, 2024.

[173] S. O. Oladejo, S. O. Ekwe and S. Mirjalili, "The Hiking Optimization Algorithm: A novel human-based metaheuristic approach," *Knowledge-Based Systems,* vol. In press, 2024.

[174] W. Zhao, L. Wang, Z. Zhang, H. Fan, J. Zhang, S. Mirjalili, N. Khodadadi and Q. Cao, "Electric eel foraging optimization: A new bio-inspired optimizer for engineering applications," *Expert Systems with Applications,* vol. 238, no. F, 2024.

[175] M. H. Amiri, N. M. Hashjin, M. Montazeri, S. Mirjalili and N. Khodadadi, "Hippopotamus optimization algorithm: a novel nature-inspired optimization algorithm," *Scientific Reports,* vol. 14, 2024.

[176] M. Abdel-Basset, R. Mohamed and M. Abouhawwash, "Crested Porcupine Optimizer: A new nature-inspired metaheuristic," *Knowledge-Based Systems,* vol. 284, 2024.

[177] M. Hubálovská, Š. Hubálovský and P. Trojovský, "Botox Optimization Algorithm: A New Human-Based Metaheuristic Algorithm for Solving Optimization Problems," *Biomimetics,* vol. 9, no. 3, 2024.

[178] N. Fang and Q. Cao, "Leaf in Wind Optimization: A New Metaheuristic Algorithm for Solving Optimization Problems," *IEEE Access,* pp. 56291 - 56308, 2024.

[179] R. K. Hamad and T. A. Rashid, "GOOSE algorithm: a powerful optimization tool for real-world engineering challenges and beyond," *Evolving Systems,* vol. 15, pp. 1249-1274, 2024.

[180] S. Zhou, Y. Shi, D. Wang, X. Xu, M. Xu and Y. Deng, "Election Optimizer Algorithm: A New Meta-Heuristic Optimization Algorithm for Solving Industrial Engineering Design Problems," *Mathematics,* vol. 12, no. 10, 2024.

[181] S. Fu, K. Li, H. Huang, C. Ma, Q. Fan and Y. Zhu , "Red-billed blue magpie optimizer: a novel metaheuristic algorithm for 2D/3D UAV path planning and engineering design problems," *Artificial Intelligence Review,* vol. 57, 2024.

[182] W. Liu and J. Wang, "A Brief Survey on Nature-Inspired Metaheuristics for Feature Selection in Classification in this Decade," in *2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*, Banff, AB, Canada, 2019.

[183] J. Barrera-García, F. Cisternas-Caneo, B. Crawford, M. G. Sánchez and R. Soto, "Feature Selection Problem and Metaheuristics: A Systematic Literature Review about Its Formulation, Evaluation and Applications," *Biomimetics,* vol. 9, no. 9, pp. 1-48, 2024.

[184] B. Crawford, R. Soto, G. Astorga, J. García, C. Castro and F. Paredes, "Putting Continuous Metaheuristics to Work in Binary Search Spaces," *Complexity,* vol. 2017, pp. 1-19, 2017.

[185] J. Kennedy and R. C. Eberhart, "A *discrete* binary versionof the particle swarm algorithm," in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, Orlando, FL, USA, 1997.

[186] S. Mirjalili *and* A. Lewis, "S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization, Swarm and Evolutionary Computation,," *Swarm and Evolutionary Computation,* vol. 9, pp. 1-14, 2013.

[187] Z. Beheshti, "A novel x-*shaped* binary particle swarm optimization," *Soft Computing,* vol. 25, pp. 3013-3042, 2021.

[188] S. Mirjalili, H. Zhang, S. Mirjalili, S. Chalup and N. Noman , "A Novel U-Shaped Transfer Function for Binary Particle Swarm Optimisation," in *Soft Computing for Problem Solving 2019 - Advances in Intelligent Systems and Computing ((AISC,volume 1138))*, Singapore, Springer, 2020, p. 241–259.

[189] S.-s. Guo, J.-s. Wang and M.-w. Guo, "Z-Shaped Transfer *Functions* for Binary Particle Swarm Optimization Algorithm," *Computational Intelligence and Neuroscience,* vol. 2020, 2020.

[190] L. Wang, X. Wang, J. Fu and L. Zhen, "A Novel Probability Binary Particle Swarm Optimization Algorithm and Its Application," *Journal of Software,* vol. 3, no. 9, pp. 28-35, 2008.

[191] J. Too, A. R. Abdullah and N. M. Saad, "A New Quadratic Binary Harris Hawk Optimization for Feature Selection," *Electronics,* vol. 8, pp. 1-27, 2019.

[192] J. Lemus-Romani, B. Crawford, F. Cisternas-Caneo, R. Soto and M. Becerra-Rozas, "Binarization of Metaheuristics: Is the Transfer Function Really Important?," *Electronics,* vol. 8, no. 5, pp. 1-47, 2023.

[193] B. Crawford, R. Soto, J. Lemus-Romani, M. Becerra-Rozas, J. M. Lanza-Gutiérrez, N. Caballé, M. Castillo, D. Tapia, F. Cisternas-Caneo, J. García, G. Astorga, C. Castro and J.-M. Rubio, "Q-Learnheuristics: Towards Data-Driven Balanced Metaheuristics," *Mathematics,* vol. 9, no. 16, 2021.

[194] R. A. Khurma, I. Aljarah, A. Sharieh, M. A. Elaziz, R. Damaševičius and T. Krilavičius , "A Review of the Modification Strategies of the Nature Inspired Algorithms for Feature Selection Problem," *Mathematics,* vol. 10, pp. 1-46, 2022.

[195] J. O. Agushaka and A. E. Ezugwu, "Initialisation Approaches for Population-Based Metaheuristic Algorithms: A Comprehensive Review," *Applied Sciences,* vol. 12, no. 2, 2022.

[196] B. Morales-Castañeda, D. Zaldívar, E. Cuevas, F. Fausto and A. Rodríguez, "A better balance in metaheuristic algorithms: Does it exist?," *Swarm and Evolutionary Computation,* vol. 54, 2020.

[197] R. Tang, S. Fong and N. Dey, "Metaheuristics and Chaos Theory," in *Chaos Theory*, InTech, 2018, pp. 181-196.

[198] H. R. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," in *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, Vienna, Austria, 2005.

[199] N. Nahar, F. Ara, M. A. I. Neloy, . A. Biswas, M. S. Hossain and K. Andersson, "Feature Selection Based Machine Learning to Improve Prediction of Parkinson Disease.," in *Brain Informatics - Lecture Notes in Computer Science ((LNAI,volume 12960))*, Springer, Cham., 2021, p. 496–508.

[200] A. K. Tiwari, "Machine Learning based approaches for prediction of Parkinson's Disease," *Machine Learning and Applications: An International Journal (MLAIJ),* vol. 3, no. 2, pp. 33-39, 2016.

[201] I. Nissar, D. R. Rizvi, S. Masood and A. N. Mir, "Voice-Based Detection of Parkinson's Disease through Ensemble Machine Learning Approach: A Performance Study," *EAI Endorsed Transactions on Pervasive Health and Technology,* vol. 5, no. 19, pp. 1-8, 2019.

[202] C. Bunterngchit and Y. Bunterngchit, "A Comparative Study of Machine Learning Models for Parkinson's Disease Detection," in *2022 International Conference on Decision Aid Sciences and Applications (DASA)*, Chiangrai, Thailand, 2022.

[203] A. Pasha and P. H. Latha, "Bio-inspired dimensionality reduction for Parkinson's disease (PD) classifcation," *Health Information Science and Systems,* vol. 8, no. 13, 2020.

[204] S. Shafiq, S. Ahmed, M. S. Kaiser, M. Mahmud, M. S. Hossain and K. Andersson, "Comprehensive Analysis of Nature-Inspired Algorithms for Parkinson's Disease Diagnosis," *IEEE Access,* vol. 11, pp. 1629-1653, 2023.

[205] G. R. Raidl, "A Unified View on Hybrid Metaheuristics. Lecture Notes in Computer Science ((LNTCS,volume 4030))," in *Hybrid* Metaheuristics., 2006, pp. 1-12.

[206] E.-G. Talbi, "Combining metaheuristics with mathematical programming, constraint programming and machine learning," *Annals of Operations Research,* vol. 240, pp. 171-215, 2016.

[207] O. A. Akinola, A. E. Ezugwu, O. N. Oyelade and J. O. Agushaka , "A hybrid binary dwarf mongoose optimization algorithm with simulated annealing for feature selection on high dimensional multi-class datasets," *Scientific Reports,* vol. 12, pp. 1-22, 2022.

[208] E.-G. Talbi, Metaheuristics: From design to implementation, Hoboken, New Jersey: John Wiley & Sons Inc,, 2009, pp. 57-67.

[209] M. Mafarja, A. Qasem, A. A. Heidari, I. Aljarah, H. Faris and S. Mirjalili, "Efficient Hybrid Nature-Inspired Binary Optimizers for Feature Selection," *Cognitive Computation,* vol. 12, pp. 150-175, 2020.

[210] M. M. Mafarja and S. Mirjalili, "Hybrid Whale Optimization Algorithm with simulated annealing for feature selection," *Neurocomputing,* vol. 260, pp. 302-312, 2017.

[211] N. Caballé-Cervigón , J. L. Castillo-Sequera , J. A. Gómez-Pulido, J. M. Gómez-Pulido and M. L. Polo-Luque, "Machine Learning Applied to Diagnosis of Human Diseases: A Systematic Review," *Applied Sciences,* vol. 1015, 2020.

[212] A. Rana, A. Dumka, R. Singh, M. K. Panda, N. Priyadarshi and B. Twala, "Imperative Role of Machine Learning Algorithm for Detection of Parkinson's Disease: Review, Challenges and Recommendations," *Diagnostics,* vol. 12, no. 8, 2022.

[213] J. Han, J. Pei and H. Tong, Data Mining - Concepts and Techniques, 4th edition, MA - USA: Morgan Kaufmann - Elsevier Inc., 2023, pp. 266-268.

[214] IBM, "What is the k-nearest neighbors (KNN) algorithm?," [Online]. Available: https://www.ibm.com/topics/knn. [Accessed 6 May 2024].

[215] B. E. Boser, I. M. Guyon and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *COLT '92: Proceedings of the fifth annual workshop on* Computational *learning theory*, 1992.

[216] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing,* vol. 408, pp. 189-215, 2020.

[217] IBM, "What is Random Forest?," [Online]. Available: https://www.ibm.com/topics/random-forest. [Accessed 8 May 2024].

[218] S. Gibson, B. Issac, L. Zhang and S. M. Jacob, "Detecting Spam Email With Machine Learning Optimized With Bio-Inspired Metaheuristic Algorithms," *IEEE Access,* vol. 8, pp. 187914-187932, 2020.

[219] B. Guo, J. Hu, W. Wu, Q. Peng and F. Wu, "The Tabu_Genetic Algorithm: A Novel Method for Hyper-Parameter Optimization of Learning Algorithms," *Electronics,* vol. 8, no. 5, pp. 1-19, 2019.

[220] M. Tayebi and S. E. Kafhali, "Performance analysis of metaheuristics based hyperparameters optimization for fraud transactions detection," *Evolutionary Intelligence,* vol. 17, pp. 921-939, 2024.

[221] H. T. Ibrahim, W. J. Mazher, O. N. Ucan and O. Bayat, "A grasshopper optimizer approach for feature selection and optimizing SVM parameters utilizing real biomedical data sets,"

Neural Computing and Applications, vol. 31, no. 10, p. 5965–5974, 2019.

[222] H. Faris, M. A. Hassonah, A. M. Al-Zoubi, S. Mirjalili and I. Aljarah , "A multi-verse optimizer approach for feature selection and optimizing SVM parameters based on a robust system architecture," *Neural Computing & Applications,* vol. 30, p. 2355–2369, 2018.

[223] R. Durgut, Y. Y. Baydilli and M. E. Aydin, "Feature Selection with Artificial Bee Colony Algorithms for Classifying Parkinson's Diseases," in *Proceedings of the 21st* EANN *(Engineering Applications of Neural Networks) 2020 Conference. EANN 2020. Proceedings of the International Neural Networks Society*, vol. 2, Springer, Cham, 2020, p. 338–351.

[224] H. Xie, L. Zhang, C. P. Lim, Y. Yu and H. Liu, "Feature Selection Using Enhanced Particle Swarm Optimisation for Classification Models," *Sensors,* vol. 21, no. 5, 2021.

[225] M. H. Nadimi-Shahraki, M. Banaie-Dezfouli, H. Zamani, S. Taghian and S. Mirjalili, "B-MFO: A Binary Moth-Flame Optimization for Feature Selection from Medical Datasets," *Computers,* vol. 10, no. 11, 2021.

[226] S. V. T. Dao, Z. Yu, L. V. Tran, P. N. K. Phan, T. T. M. Huynh and T. M. Le, "An Analysis of Vocal Features for Parkinson's Disease Classification Using Evolutionary Algorithms," *Diagnostics,* vol. 12, no. 8, 2022.

[227] J. Too and A. R. Abdullah , "Chaotic Atom Search Optimization for Feature Selection," *Arabian Journal for Science and Engineering,* vol. 45, pp. 6063-6079, 2020.

[228] L. Fang and X. Liang, "A Novel Method Based on Nonlinear Binary Grasshopper Whale Optimization Algorithm for Feature Selection," vol. 20, pp. 237-252, 2023.

[229] B. Sabeena, S. Sivakumari and D. M. Teressa, "Optimization-Based Ensemble Feature Selection Algorithm and Deep Learning Classifier for Parkinson's Disease," *Journal of Healthcare Engineering,* 2022.

[230] I. M. El-Hasnony, S. I. Barakat, M. Elhoseny and R. R. Mostafa, "Improved Feature Selection Model for Big Data Analytics," *IEEE Access,* vol. 8, pp. 66989 - 67004, 2020.

[231] F. Abukhodair, W. Alsaggaf, A. T. Jamal, S. Abdel-Khalek and R. F. Mansour, "An Intelligent Metaheuristic Binary Pigeon Optimization-Based Feature Selection and Big Data Classification in a MapReduce Environment," *Mathematics,* vol. 9, no. 20, 2021.

[232] P. Shrivastava, A. Shukla, P. Vepakomma, N. Bhansali and K. Verma, "A survey of nature-inspired algorithms for feature selection to identify Parkinson's disease," *Computer Methods and Programs in Biomedicine,* vol. 139, pp. 171-179, 2017.

[233] P. Sharma, S. Sundaram, M. Sharma, A. Sharma and D. Gupta, "Diagnosis of Parkinson's disease using modified grey wolf optimization," *Cognitive Systems Research,* vol. 54, pp. 100-115, 2019.

[234] A. Adamu, M. Abdullahi, S. B. Junaidu and I. H. Hassan, "An hybrid particle swarm optimization with crow search algorithm for feature selection," *Machine Learning with Applications,* vol. 6, pp. 1-13, 2021.

[235] H. Hichem, M. Elkamel, . M. Rafik, M. T. Mesaaoud and C. Ouahiba, "A new binary grasshopper optimization algorithm for feature selection problem," *Journal of King Saud University - Computer and Information Sciences,* vol. 34, no. 2, pp. 316-328, 2022.

[236] G. Hu, B. Du, X. Wang and G. Wei, "An enhanced black widow optimization algorithm for feature selection," *Knowledge-Based Systems,* vol. 235, pp. 1-26, 2022.

[237] D. A. Elmanakhly, M. M. Saleh and E. A. Rashed, "An Improved Equilibrium Optimizer Algorithm for Features Selection: Methods and Analysis," *IEEE Access,* vol. 9, pp. 120309 -

120327, 2021.

[238] D. Gupta, S. Sundaram, A. Khanna, A. E. Hassanien and V. H. C. de Albuquerque, "Improved diagnosis of Parkinson's disease using optimized crow search algorithm," Computers *& Electrical Engineering,* vol. 68, pp. 412-424, 2018.

[239] A. A. Bahaddad, M. Ragab, E. B. Ashary and E. M. Khalil, "Metaheuristics with Deep Learning-Enabled Parkinson's Disease Diagnosis and Classification Model," *Journal of Healthcare Engineering,* pp. 1-14, 2022.

[240] S. Sehgal, M. Agarwal, D. Gupta, S. Sundaram and A. Bashambu , "Optimized grass hopper algorithm for diagnosis of Parkinson's disease," *SN Applied Sciences,* vol. 2, 2020.

[241] S. Dash, A. Abraham, A. K. Luhach, J. Mizera-Pietraszko and J. J. Rodrigues, "Hybrid chaotic firefly decision making model for Parkinson's disease diagnosis," International *Journal of Distributed Sensor Networks,* vol. 16, no. 1, 2020.

[242] N. A. Al-Thanoon, O. S. Qasim and Z. Y. Algamal, "Improving nature-inspired algorithms for feature selection," *Journal of Ambient Intelligence and Humanized Computing,* vol. 13, pp. 3025-3035, 2022.

[243] Z. Cai, J. Gu, C. Wen, D. Zhao, C. Huang, H. Huang, C. Tong, J. Li and H. Chen, "An Intelligent Parkinson's Disease Diagnostic System Based on a Chaotic Bacterial Foraging Optimization Enhanced Fuzzy KNN Approach," *Computational and Mathematical Methods in Medicine,* pp. 1-24, 2018.

[244] D. Gupta, . A. Julka , S. Jain , T. Aggarwal, A. Khanna, N. Arunkumar and V. H. C. de Albuquerque, "Optimized cuttlefish algorithm for diagnosis of Parkinson's disease," *Cognitive Systems Research,* vol. 52, pp. 36-48, 2018.

[245] Q. Li, H. Chen, H. Huang, X. Zhao, Z. Cai, C. Tong, W. Liu and X. Tian, "An Enhanced Grey Wolf Optimization Based Feature Selection Wrapped Kernel Extreme Learning Machine for Medical Diagnosis," *Computational and Mathematical Methods in Medicine,* vol. 2017, no. 9512741, 2017.

[246] M. K. Shahsavari, H. Rashidi and H. R. Bakhsh, "Efficient classification of Parkinson's disease using extreme learning machine and hybrid particle swarm optimization," in *2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA)*, Qazvin, Iran, 2016.

[247] W.-L. Zuo, Z.-Y. Wang, . T. Liu and H.-L. Chen, "Effective detection of Parkinson's disease using an adaptive fuzzy k-nearest neighbor approach," *Biomedical Signal Processing and Control,* vol. 8, no. 4, pp. 364-373, 2013.

[248] D. Tomar, B. R. Prasad and S. Agarwal, "An efficient Parkinson disease diagnosis system based on Least Squares Twin Support Vector Machine and Particle Swarm Optimization," in *2014 9th International Conference on Industrial and Information Systems (ICIIS)*, Gwalior, India, 2014.

[249] Too, Jingwei; Abdullah , Abdul Rahim; Saad, Norhashimah Mohd;, "A New Co-Evolution Binary Particle Swarm Optimization with Multiple Inertia Weight Strategy for Feature Selection," *Informatics,* vol. 6, no. 2, 2019.

[250] A. P. Engelbrecht, J. Grobler and J. Langeveld, "Set based particle swarm optimization for the feature selection problem," *Engineering Applications of Artificial Intelligence,* vol. 85, pp. 324-336, 2019.

[251] J. Too, A. R. Abdullah and N. M. Saad, "A New Quadratic Binary Harris Hawk Optimization for Feature Selection," *Electronics,* vol. 8, no. 10, pp. 1-27, 2019.

[252] P. Romanski, L. Kotthoff and P. Schratz, "FSelector: Selecting Attributes - R package version 0.34," 2023. [Online]. Available: https://CRAN.R-project.org/package=FSelector.

[253] M. B. Kursa, "Praznik: High performance information-based feature selection," *SoftwareX,* vol. 16, 2021.

[254] H. Peng, F. Long and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 27, no. 8, pp. 1226 - 1238, 2005.

[255] J. H. Holland, "Genetic Algorithms," *Scientific American,* vol. 267, pp. 66-73, 1992.

[256] X.-S. Yang, Nature-Inspired Optimization Algorithms, Elsevier Inc., 2014.

[257] S. KIRKPATRICK, C. D. GELATT, Jr *and* M. P. VECCHI, "Optimization by Simulated Annealing," *Science,* vol. 220, no. 4598, pp. 671-680, 1983.

[258] H. Szu and R. Hartley, "Fast simulated annealing," *Physics Letters A,* vol. 122, no. 3-4, pp. 157-162, 1987.

[259] C. Tsallis and D. A. Stariolo, "Generalized simulated annealing," *Physica A: Statistical Mechanics and its Applications,* vol. 233, no. 1-2, pp. 395-406, 1996.

[260] Y. Xiang, S. Gubia, B. Suomela and J. Hoeng, "Generalized Simulated Annealing for Global Optimization: The GenSA Package," *The R Journal,* pp. 13-28, 2013.

[261] B. Bischl, M. Lang, L. Kotthoff, J. Schiffner, J. Richter, E. Studerus, G. Casalicchio and Z. M. Jones, "Machine Learning in R," 2021.

[262] R. Moghdani and K. Salimifard, "Volleyball Premier League Algorithm," *Applied Soft Computing,* vol. 64, pp. 161-185, 2018.

[263] T. Wolfanger, "TouRnament: Tools for Sports Competitions - R package version 0.2.5," 2019. [Online]. Available: https://CRAN.R-project.org/package=TouRnament.

[264] H. Tong, Y. Zhu and Y. Xu, "An Enhanced Volleyball Premier League Algorithm with Chaotic Maps," in *2020 12th International Conference on Advanced Computational Intelligence (ICACI)*, Dali, China, 2020.

[265] R. Moghdani, M. A. Elaziz, D. Mohammadi and N. Neggaz , "An improved volleyball premier league algorithm based on sine cosine algorithm for global optimization problem," *Engineering with Computers,* vol. 37, pp. 2633-2662, 2021.

[266] S. Mahdavi, S. Rahnamayan and K. Deb, "Opposition based learning: A literature review," *Swarm and Evolutionary Computation,* vol. 39, pp. 1-23, 2018.

[267] S. Mirjalili, "The Ant Lion Optimizer," *Advances in Engineering Software,* vol. 83, pp. 80-98, 2015.

[268] E. Emary, H. M. Zawbaa and A. E. Hassanien, "Binary ant lion approaches for feature selection," *Neurocomputing,* vol. 213, pp. 54-65, 2016.

[269] K. Ramasubramanian and A. *Singh*, "Sampling and Resampling Techniques," in *Machine Learning Using R: With Time Series and Industry-Based Use Cases in R, 2nd edition*, Berkley, CA, Apress, 2019, pp. 79-150.

[270] J. Too, "Ant Colony Optimization for Feature Selection," [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/80278-ant-colony-optimization-for-feature-selection?s_tid=srchtitle. [Accessed 26 May 2022].

[271] M. K. Heris, "Artificial Bee Colony in MATLAB," 2015. [Online]. Available: https://yarpiz.com/297/ypea114-artificial-bee-colony.

[272] J. Too and A. R. Abdullah, "Binary atom search optimisation approaches for feature selection," *Connection Science,* vol. 32, no. 4, pp. 406-430, 2020.

[273] S. Mirjalili, S. M. Mirjalili and X.-S. Yang, "Binary bat algorithm," *Neural Computing and Applications,* vol. 25, pp. 663-681, 2014.

[274] J. Too, A. R. Abdullah and N. M. Saad, "Hybrid Binary Particle Swarm Optimization Differential Evolution-Based Feature Selection for EMG Signals Classification," *Axioms,* vol. 8, no. 3, 2019.

[275] J. Too and S. Mirjalili, "A Hyper Learning Binary Dragonfly Algorithm for Feature Selection: A COVID-19 Case Study," *Knowledge-Based Systems,* vol. 212, pp. 1-16, 2021.

[276] M. K. Heris, "Firefly Algorithm in Matlab," 2015. [Online]. Available: https://yarpiz.com/259/ypea112-firefly-algorithm.

[277] J. Too, A. . R. Abdullah, N. M. Saad, N. M. Ali and W. Tee, "A New Competitive Binary Grey Wolf Optimizer to Solve the Feature Selection Problem in EMG Signals Classification,," *Computers,* vol. 7, no. 4, 2018.

[278] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic," *Knowledge-Based Systems,* vol. 89, pp. 228-249, 2015.

[279] J. Too, "Particle Swarm Optimization for Feature Selection," [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/71470-binary-particle-swarm-optimization-for-feature-selection. [Accessed 15 March 2022].

[280] J. Too, "Salp Swarm Algorithm for Feature Selection," 10 January 2021. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/78913-salp-swarm-algorithm-for-feature-selection?s_tid=prof_contriblnk.

[281] J. Too, A. R. Abdullah, N. M. Saad and N. M. Ali, "Feature Selection Based on Binary Tree Growth Algorithm for the Classification of Myoelectric Signals," *Machines,* vol. 6, no. 4, pp. 1-19, 2018.

[282] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software,* vol. 95, pp. 51-67, 2016.

[283] J. Too and S. Mirjalili, "General Learning Equilibrium Optimizer: A New Feature Selection Method for Biological Data Classification," *Applied* Artificial *Intelligence,* vol. 35, no. 3, pp. 247-263, 2021.

[284] J. Too and A. R. Abdullah , "A new and fast rival genetic algorithm for feature selection," *The Journal of Supercomputing,* vol. 77, p. 2844–2874, 2021.

[285] J. Too, "Sine Cosine Algorithm for Feature Selection," 9 January 2021. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/80671-sine-cosine-algorithm-for-feature-selection?s_tid=prof_contriblnk.

[286] M. K. Heris, "Teaching-Learning-based Optimization in MATLAB," 2015. [Online]. Available: https://yarpiz.com/83/ypea111-teaching-learning-based-optimization.

[287] S. Saremi, S. Mirjalili and A. Lewis, "Grasshopper Optimisation Algorithm: Theory and application," *Advances in Engineering Software,* vol. 105, pp. 30-47, 2017.

Appendix A - List of tables

Appendix B - List of figures