UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SINGLE, MULTI- AND MANY-OBJECTIVE META-HEURISTIC ALGORITHMS APPLIED TO PATTERN RECOGNITION

Douglas Rodrigues

Orientador: Prof. Dr. João Paulo Papa

São Carlos – SP

Junho/2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SINGLE, MULTI- AND MANY-OBJECTIVE META-HEURISTIC ALGORITHMS APPLIED TO PATTERN RECOGNITION

Douglas Rodrigues

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação, área de concentração: Metodologias e Técnicas de Computação
Orientador: Prof. Dr. João Paulo Papa

São Carlos – SP

Junho/2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Tese de Doutorado do candidato Douglas Rodrigues, realizada em 10/07/2019:

João Paulo Papa

Prof. Dr. João Paulo Papa
UFSCar

Alexandre Luis M. Levada

Prof. Dr. Alexandre Luis Magalhães Levada
UFSCar

Prof. Dr. Cesar Henrique Comin
UFSCar

Prof. Dr. Andre Carlos Ponce de Leon Ferreira de Carvalho
USP

Thierry P Moreira

Prof. Dr. Thierry Pinheiro Moreira
UNESP

Aos meus pais, Ivone e Osmar.

Agradecimentos

Agradeço primeiramente aos meus *pais*, por todo amor, apoio e dedicação.

Agradeço a todos os membros do **Recogna**[1].

Agradeço aos professores da UFSCAR, em especial, o professor *Alexandre Levada*, que acompanhou e colaborou com meu desenvolvimento acadêmico.

E agradeço, especialmente, meu orientador e amigo *João Paulo Papa* pelo apoio, paciência e dedicação.

[1] http://www.recogna.tech/

Escolhe um trabalho de que gostes, e não terás que trabalhar nem um dia na tua vida.

Confúcio

Resumo

Algoritmos meta-heurísticos têm sido empregados, nos últimos anos, para a resolução de diversos problemas na área de engenharia, biologia, física, entre outras, dado que muitos deles podem ser modelados como tarefas de otimização. Tais métodos meta-heurísticos simulam dinâmicas sociais e fenômenos físicos como a interação entre morcegos, algumas espécies de aves, insetos ou até mesmo a própria força gravitacional. Muito embora, essas técnicas meta-heurísticas sejam comumente aplicadas na resolução de problemas mono-objetivo, elas também estão sendo utilizadas para a resolução de problemas multi e de muitos objetivos, onde a ideia de uma única solução ótima global é substituída pelo conceito de fronteira Pareto-ótima. Na área de visão computacional e reconhecimento de padrões, pouco ainda tem sido explorado no que diz respeito à otimização multi-objetivos utilizando meta-heurísticas. Desta forma, a presente tese objetiva o estudo e desenvolvimento de versões mono, multi, e de muitos objetivos de novas técnicas meta-heurísticas no contexto de aprendizado de máquina, que engloba, dentre outras áreas, a seleção e combinação de características, bem como otimização de parâmetros de técnicas de aprendizado de máquina e aprendizado em profundidade.

Palavras-chave: Aprendizado de Máquina, Algoritmos Meta-heurísticos, Otimização

Abstract

In the last few years, metaheuristic algorithms have been used for solving several problems in engineering, biology, physics, among others, since many of them can be modeled as being optimization tasks. Metaheuristic methods simulate social dynamics and physical phenomena such as the interaction among bats, some species of birds, insects or even gravitational force. Although these metaheuristic techniques are commonly applied to solve single-objective problems, they are also being used to solve multi- and many-objective problems, where the idea of a single global optimal solution is replaced by the concept of Pareto-front. In computer vision and pattern recognition areas, little effort has been dedicated to multi-objective optimization using metaheuristics. As such, this thesis aims at studying and developing new mono, multi- and many-objective versions of metaheuristic techniques in the context of machine learning, which include, among other areas, feature combination and selection, parameter optimization of machine learning techniques and deep learning.

Keywords: Machine Learning, Meta-heuristic Algorithms, Optimization

List of Figures

List of tables

Summary

Chapter 1

Introduction

Many problems found in the most diverse areas of research can be modeled as an optimization task, where it is desired to find the maximum/minimum of a given function. However, it is not always possible or feasible to find the optimal solution of the problem, as in many NP-Hard models type. An example of NP-Hard optimization would be the case of the travelling salesman problem, where the objective is to find the smallest route to travel through a group of cities by visiting each one only once. An optimization problem can be modeled by identifying the objective function, the decision variables and the problem constraints (NOCEDAL; WRIGHT, 2006). The objective function is a quantitative measure of system performance and depends on decision variables, which in turn must respect certain constraints.

Meta-heuristic algorithms have become popular in the last few decades as they are applicable to a wide variety of optimization problems that can not be solved accurately within a reasonable amount of time. Using simple rules and principles, meta-heuristic algorithms tend to explore the whole search space, and as soon as they find promising regions, they perform a more refined search in order to find the optimal solution. Such algorithms use concepts based on social dynamics and behavior of several living beings and they were initially developed to deal with single-objective problems, that is, problems that are modeled with only one objective function. Among the best well-known, we can cite the Genetic Algorithm (GA) (HOLLAND, 1992), Particle Swarm Optimization (PSO) (KENNEDY; EBERHART, 2001), and the Gravitational Search Algorithm (GSA) (RASHEDI; Nezamabadi-pour; SARYAZDI, 2009), among others.

In many real optimization problems, it is common to have more than one objective function, called Multi-objective Optimization Algorithms (MOAs). In this case, it is important to point out that if the objectives involved in the optimization process are not conflicting with each other, the problem happens to have only one optimal solution (ZITZLER, 1999). Assuming that the objective functions are conflicting to each other, multi-objective optimization problems present a set of solutions considered optimal replacing the idea of a global optimal solution. One of the most common strategies for solving multi-objective optimization problems is the weighted-sum method, which consists in obtaining a single objective by means of the scalar product between a vector of weights and a vector of objective functions. The main disadvantage of the weighted-sum method is that it can not generate all the solutions in non-convex Pareto Front problems (COELLO; LAMONT; VELDHUIZEN, 2007; ZITZLER, 1999). However, such strategies where the idea is to aggregate the objective functions in order to reduce the multi-objective problem to a single-objective problem are categorized as mathematical programming, and may not work when the Pareto-optimal front is concave or disconnected. In Chapter 2, a brief explanation is given of various mathematical programming techniques.

The first algorithm developed to deal with multi-objective problems was the Vector Estimating Genetic Algorithm (VEGA) (SCHAFFER, 1985) proposed by David Schaffer in the mid-1980s. Many other algorithms have been developed as follows: Multi-Objective Genetic Algorithm (MOGA) (FONSECA; FLEMING, 1993), Niched Pareto Genetic Algorithm (NPGA) (HORN; NAFPLIOTIS; GOLDBERG, 1994), Non-Dominated Sorting Genetic Algorithms (NSGA) (SRINIVAS; DEB, 1994), Strength Pareto Evolutionary Algorithm (SPEA) (ZITZLER; THIELE, 1999), Archived Evolution Strategy (PAES) (KNOWLES; CORNE, 1999), among others. Deb et al. (DEB et al., 2002) proposed the NSGA-II, which underwent some changes in relation to its first version with respect to the non-dominated sorting algorithm to become computationally faster, the crowding distance was also introduced to increase the diversity of the solutions and eliminated the sharing parameter.

MOAs have proven to be highly efficient in solving real-world problems with two or three objective functions (COELLO; LAMONT; VELDHUIZEN, 2007). However, many problems involve a greater number of objective functions (FABRE, 2009), which are called Many-Objective Problems (MaOPs) (ISHIBUCHI; TSUKAMOTO; NOJIMA, 2008). Based on studies involving problems with four or more objective functions, it was noticed that MOAs have an inability to discriminate solutions. This is due to the fact that increasing objective functions makes the solutions unmatched. Thus, techniques with a greater discriminative capacity were developed (HUGHES, 2005; SATO; AGUIRRE; TANAKA, 2007; AGUIRRE; TANAKA, 2009; FABRE, 2009).

Single, multi-, and many-objective optimization algorithms are being implemented to solve optimization problems in the most diverse areas of knowledge, from engineering to economics. However, little has yet been explored in computer vision and pattern recognition fields. In the context of feature selection, for example, the NSGA-II (HAMDANI et al., 2007) and PSO (XUE; ZHANG; BROWNE, 2013) techniques were employed with the aim of optimizing the number of selected features, as well as to minimize the classification error rate. Abbass (ABBASS, 2003) has used the Memetic Pareto Artificial Neural Network (MPANN) to optimize the training error and the architecture of a neural network, and Liu and Deng (LIU; DENG, 2010) applied MPANN to detect carcinoma in mammographic images. Wiegand et al. (WIEGAND; IGEL; HANDMANN, 2004) proposed the optimization of weights and the speed of a neural network for face detection, and Jin (JIN, 2004) optimized both the parameters and structure of a recurrent neural network using NSGA-II. Recently, Onety et al. (ONETY et al., 2013) proposed the Variable Neighborhood Multiobjective Genetic Algorithm (VN-MGA), which is a variation of NSGA-II for routing optimization in the context of computer networks. In addition, Yusiong and Naval Jr. (YUSIONG; JR., 2006) used Multi-Object Particle Swarm Optimization-Crowding Distance (MOPSO-CD) to optimize the architecture and weights of a recurrent neural network. Igel and Suttorp (IGEL, 2005; SUTTORP; IGEL, 2006) have used multi-objective evolutionary algorithms to optimize the selection of models in Support Vector Machines (SVM), where there is a cost/benefit relationship between two or more objectives such as optimization of parameters and kernel function (mapping). Miranda et al. (MIRANDA et al., 2012) used Multi-objective Particle Swarm Optimization (MOPSO) to maximize the accuracy rate and minimize the number of support vectors of the SVM classifier.

The hypothesis and main contributions of the present thesis regard answering the following question: Do meta-heuristic algorithms help improve the performance of machine learning techniques? Two approaches are proposed to accomplish such task:

- **a literary study on single, multi- and many-objective optimization algorithms applied to the machine learning area;**
- **the application of single, multi- and many-objective meta-heuristic optimization algorithms to enhance the performance of machine learning techniques.**

The experimental results, discussed in the following chapters, support the proposed hypothesis. Moreover, this thesis is composed of a collection of works published/submitted by the authors during the period of study.

The remainder of the paper is organized as follows: in Chapter 2, a bibliographic review of the themes that are related to this research is presented. In Chapter 3, we present a methodology used for feature selection in fraud detection in an electric company. In Chapter 4, meta-heuristic algorithms are applied for channel selection using electroencephalogram tests for people recognition. In Chapter 5, meta-heuristic algorithms are employed to find the optimal parameters of a deep belief network. In Chapter 6, a multi-objective optimization concerning Optimum-Path Forest pruning is presented. In Chapter 7, we present two multi- and many-objective feature selection approaches. Finally, conclusions, contributions and future opportunities are presented in Chapter 8.

Chapter 2

Meta-heuristic Multi- and Many-objective Optimization Techniques for Solution of Machine Learning Problems

This chapter presents the theoretical basis for single, multi- and many-objective optimization, as well as a collection of articles in which single, multi- and many-objective optimization is used in the machine learning area. This work was published in the *Expert Systems* (RODRIGUES; PAPA; ADELI, 2017).

2.1 Introduction

Optimization techniques have been widely used in several research areas, since many problems usually refer to the task of finding the maximum/minimum of a given function. Some challenges such as the allocation of resources, product delivery for logistic companies, and cutting and packing problems with direct application in industries are among the most widely pursued tasks.

In regard to optimization techniques, a considerable attention has been given to methodologies based on meta-heuristics, i.e., approaches that aim at solving several problems using concepts based on social dynamics and/or the behavior of living beings. Among the most widely techniques, we shall cite Genetic Algorithm (GA) (HOLLAND, 1992), Particle Swarm Optimization (PSO) (KENNEDY; EBERHART, 2001), Harmony Search (HS) (GEEM, 2009), and Gravitational Search Algorithm (GSA) (RASHEDI; Nezamabadi-pour; SARYAZDI, 2009), just to name a few. In the past years, such optimization techniques have been applied for solving multi-objective optimization problems (MOP), where the idea of an unique global optimal solution is replaced by a non-dominated solution set or Pareto-optimal set (FONSECA; FLEMING, 1993; HORN; NAFPLIOTIS; GOLDBERG, 1994; SRINIVAS; DEB, 1994; ZITZLER; THIELE, 1999; KNOWLES; CORNE, 1999; DEB et al., 2002).

Multi-objective optimization techniques have become popular to solve many optimization problems in the field of engineering (SIVASUBRAMANI; SWARUP, 2011; OMKAR et al., 2011; AKBARI et al., 2012; LAU et al., 2013; KHALILI-DAMGHANI; ABTAHI; TAVANA, 2013a; ZHENG; SONG; CHEN, 2013; MARICHELVAM; PRABAHARAN; YANG, 2014). However, multi-objective optimization has a wide range of applications, mainly in the context of machine learning-oriented problems, which are usually composed of several multi-objective tasks (JIN; SENDHOFF, 2008). It is very common to face problems in which we need to find out the best set of parameters (e.g., a neural network architecture) that lead to both high recognition rates and low computational burden. Therefore, this survey aims at contributing to the aforementioned context, i.e., we present here a review of techniques for the optimization of machine learning techniques, which includes object description, classification and recognition. Among the possible applications, we can highlight:

- feature extraction and selection;
- hyper-parameter optimization and model selection; and
- clustering.

The remainder of this paper is organized as follows. Section 2.2 presents the theoretical background regarding single and multi-objective optimization. Furthermore, Section 2.4 states the literature background in the context of machine learning-oriented works, and Section 2.5 states conclusions and future directions.

2.2 Theoretical Background

In this section, we introduce some basic concepts regarding single and multi-objective optimization.

2.2.1 Single-Objective Optimization

Let $\vec{x} = (x_1, x_2, \ldots, x_N)$ be an N-dimensional vector, and $\mathscr{S}$ an arbitrary search space. Single objective optimization problems try to minimize an objective function $f(\vec{x})$. Therefore, $\vec{x}^*$ is considered the global minimum with respect to some objective function $f : \Re^N \rightarrow \Re$ if, and only if, $f(\vec{x}^*) \geq f(\vec{x}), \forall \vec{x} \in S$. Mathematically speaking, we have:

$$\vec{x}^* = \arg \min_{\forall \vec{x} \in \mathscr{S}} (f(\vec{x})), \tag{2.1}$$

subject to:

$$g_i(\vec{x}) = 0 \quad \forall i = 1, 2, \ldots, p, \tag{2.2}$$

$$h_i(\vec{x}) \geq 0 \quad \forall i = 1, 2, \ldots, q, \tag{2.3}$$

where p and q represent the number of equality $g(\cdot)$ and inequality constraints $h(\cdot)$.

2.2.2 Multi-Objective Optimization

In this section, a theory background on multi-objective optimization using Pareto approach is introduced, in which the idea is to optimize two or more objective functions at the same time. A multi-objective optimization problem aims at finding the global minimum $\vec{x}^* \in \mathscr{S}$ that minimizes a function set M represented by $\vec{f}$, i.e.:

$$\vec{x}^* = \arg \min_{\forall \vec{x} \in \mathscr{S}} \left(\vec{f}(\vec{x}) \right) = \arg \min_{\forall \vec{x} \in \mathscr{S}} (f_1(\vec{x}), f_2(\vec{x}), \ldots, f_M(\vec{x})), \tag{2.4}$$

subject to:

$$g_i(\vec{x}) = 0 \quad \forall i = 1, 2, \ldots, p, \tag{2.5}$$

$$h_i(\vec{x}) \geq 0 \quad \forall i = 1, 2, \ldots, q. \tag{2.6}$$

The set of all values satisfying the above constraints defines the feasible region, and any point in this region is thus considered a feasible solution. In a multi-objective problem, there is no single solution that is optimal with respect to all objectives when considering conflicting objectives. Thus, the solution to a multi-objective optimization problem is no longer a scalar value, but a vector in the form of a "trade-off" known as *Pareto-optimal set*. Figure 2.1 illustrates a multi-objective optimization problem with two functions, i.e. f_1 and f_2. Notice red points stand for the best solutions considering each objective function individually, and the green point is the optimal solution considering now both objective functions. Since there is solution that is optimum concerning both objective functions, we have so-called “utopia point”, which shall not be obtained as solution, in theory.

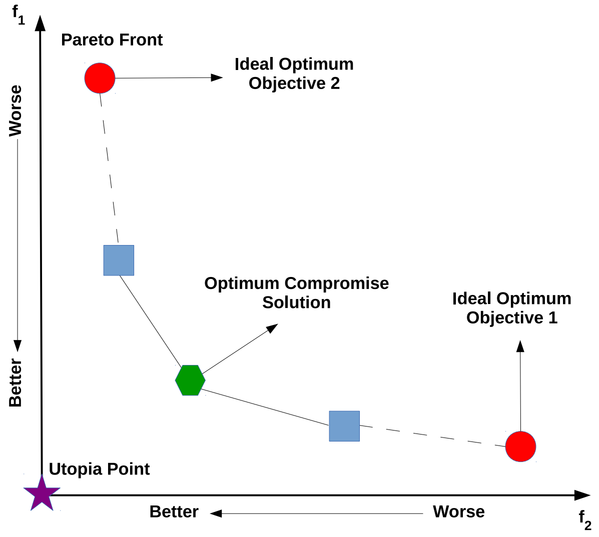


Figure 2.1: Pareto solutions considering the optimization of the objectives f_1 and f_2.

Firstly, we define the *Pareto Dominance*, where a solution vector $\vec{x}^a$ is said to dominate another solution vector $\vec{x}^b$ (i.e., $\vec{x}^a \prec \vec{x}^b$) if $f(x_i^a) \geq f(x_i^b), \forall i = \{1,2,\dots,N\}$, and $\exists i \in \{1,2,\dots,N\}$ such that $f(x_i^a) > f(x_i^b)$. In regard to the Pareto Dominance, a solution vector $\vec{x}^a$ is considered Pareto-optimal if, for every $\vec{x}^b$, $f_j(\vec{x}^a) \geq f_j(\vec{x}^b)$, $j = 1,2,\dots,M$, and if there exists at least one $j \in \{1,2,\dots,M\}$ such that $f_j(\vec{x}^a) > f_j(\vec{x}^b)$. Therefore, the Pareto-optimal set $\mathscr{P}^*$ considering a multi-objective optimization problem $\vec{f}(\vec{x})$ with respect to all Pareto-optimal solutions is thus defined as follows:

$$\mathscr{P}^* = \{\vec{x} \in \mathscr{S} \mid \vec{f}(\vec{x}) \prec \vec{f}(\vec{x}'),\ \forall \vec{x}' \in \mathscr{S}\}. \tag{2.7}$$

The Pareto-optimal front PF^* with respect to a multi-objective optimization problem $\vec{f}(\vec{x})$ and the Pareto-optimal set $\mathscr{P}^*$ is defined as follows:

$$PF^* = \{\vec{f}(\vec{x}) \mid \vec{x} \in \mathscr{P}^*\}. \tag{2.8}$$

Figure 2.2 illustrates the concept of the Pareto Dominance idea, in which f_1 and f_2 stand for two different fitness functions.

An alternative to the idea of Pareto optimality and efficiency, which yields a single solution point, is the idea of a compromise solution. It entails minimizing the difference between the potential optimal point and a utopia point (also called an ideal point). Thus,

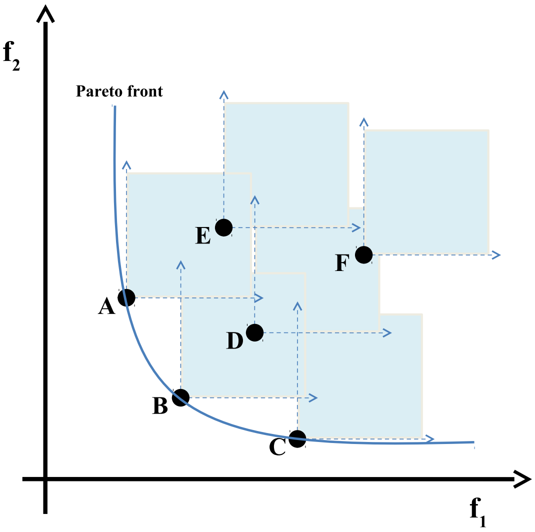


Figure 2.2: Consider a set of points “A”, “B”, “C”, “D”and “E”, as well as their axes and a light blue area highlighting the “area of influence” of such points. A point “X” is said to be dominated by another point “Y”if the former falls in the influence area of “Y”. For instance, point “A” dominates points “E” and “F”, point “B” dominates “D”, “E”and “F” and point “D” dominates point ”F”. Point “C” dominates the point “F” and finally points “E”and “F” do not dominate any other point. The blue line shows the Pareto-optimal front.

the PF^* is bounded by the nadir point z^{nad} and the utopian point $z^{utopian}$. The z^{nad} can be defined as follows:

$$z^{nad} = \arg \max_{\forall \vec{x} \in \mathscr{P}^*} (f_M(\vec{x})), \tag{2.9}$$

On the other hand, $z^{utopian}$ can be defined as follows:

$$z^{utopian} = \arg \min_{\forall \vec{x} \in \mathscr{S}} (f_M(\vec{x})), \tag{2.10}$$

In general, the $z^{utopian}$ is unattainable. The next best thing is a solution that is as close as possible to the utopia point. Such a solution is called a compromise solution and is Pareto optimal. In such a way, one may want to minimize the distance between a solution and the utopian point:

$$N(\vec{x}) = |f(\vec{x}) - z^{utopian}|, \tag{2.11}$$

Roughly speaking, multi-objective optimization techniques are often divided in scalarization and vector optimization methods, where given a vector of objective functions, it is possible to combine the components of this vector to compose a single scalar objective function. Also, a multi-objective optimization problem can be divided in: (i) classical methods, which use direct or gradient-based methods, and (ii) non-traditional methods, which follow some natural or physical principles(SHUKLA; DEB; TIWARI, 2005).

In order to solve an MOP, it is possible to identify two conceptually types of problems(HORN, 1997): search and decision making. The first refers to the optimization process in which the feasible set is sampled for Pareto-optimal solutions. The second addresses the problem of selecting a suitable compromise solution from the Pareto-optimal set. Also, an MOP is referred to supporting a human Decision Maker (DM) in finding the Pareto optimal solution according to his subjective preferences(MIETTINEN, 1998). Hence, one may classify the MOP considering the DM preferences in the following four categories: (i) non-preference method; (ii) a priori method; (iii) posteriori method; (iv) interactive method. More details about classical methods are presented in (MARLER; ARORA, 2004).

2.2.2.1 Non-preference methods

Such methods do not assume any information about the importance of objectives, no DM´s preference is expected to be available, but a heuristic is used to find a single optimum solution. They do not make any attempt to find multiple Pareto-optimal solutions. A well-known example is the global criterion method (GEN; YUN, 2006).

$$\vec{x}^{*} = \arg\min_{\forall \vec{x} \in \mathscr{S}} ((\sum_{k=1}^{M} ||f_k(\vec{x}) - z_k^{*}||^{p}))^{\frac{1}{p}}, \tag{2.12}$$

where $||.||$ stands for the $Lp-norm$ with $1 < p < \infty$. It is recommended that the objective functions are normalized into a uniform, dimensionless scale. We also may cite Nash Arbitration method(STRAFFIN, 1993) and Rao´s method(RAO, 1987).

2.2.2.2 A-priori methods

In this method, the DM´s preference information is available and then a solution best satisfying such preference is found. Figure 2.3 displays the priori method.

The weighted sum method is one of the most used approach, in which several objective functions are combined into a single one through a weight vector. Thus, a problem with multiple objective functions is reduced to a single optimization problem subject to the

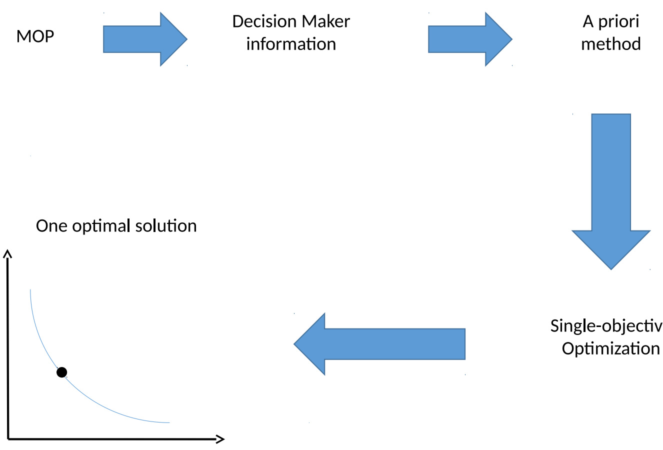


Figure 2.3: A pipeline displaying a priori method.

original constraints, and the choice of the value of each weight is performed according to a preference assigned to each objective function:

$$\vec{x}^{*} = \arg \min_{\forall \vec{x} \in \mathscr{S}} \left(\sum_{k=1}^{M} w_k f_k(\vec{x}) \right), \tag{2.13}$$

with $\sum_{i=1}^{M} w_i = 1$.

Another well-known approach is the ε-constraint method (MESSAC, 1996), in which the idea is to minimize one objective function while expressing other objectives in the form of inequality constraints. Thus, a scalarized problem with respect to the objective function $f_k(\vec{x})$ and a given vector $\vec{\varepsilon} \in \Re^{M-1}$ can be described as follows:

$$\vec{x}^{*} = \arg \min_{\forall \vec{x} \in \mathscr{S}} (f_k(\vec{x})), \tag{2.14}$$

subject to:

$$f_j(\vec{x}) \geq \varepsilon_j \quad \forall j = 1, 2, \ldots, M, j \neq k, \tag{2.15}$$

$$g_i(\vec{x}) \geq 0 \quad \forall i = 1, 2, \ldots, M, \tag{2.16}$$

Although, we have several a priori methods, among them: Weighted Global Criterion method (YU; LEITMANN, 1974; ZELENY, 1982; CHANKONG; HAIMES, 1983), Lexicographic method (MIETTINEN, 1998), Weighted Min-Max method (BOWMAN, 1976), Exponential

Weighted Criterion (ATHAN; PAPALAMBROS, 1996), Weighted Product method (BRIDGMAN, 1922), Physical Programming (HAIMES; LASDON; WISMER, 1971), Goal Programming (JONES; TAMIZ, 2010).

2.2.2.3 A-posteriori methods

The main idea of a posteriori method is to produce all the Pareto-optimal solutions or a representative subset of Pareto-optimal solutions in order to the DM choose the most preferred Pareto-optimal solution. The well-known methods are the Normal Boundary Intersection (NBI) (DAS; DENNIS, 1998), Normal Constraint (NC) (MESSAC; ISMAIL-YAHAYA; MATTSON, 2003), Successive Pareto Optimization (SPO) (MUELLER-GRITSCHNEDER; GRAEB; SCHLICHTMANN, 2009) and Directed Search Domain (DSD) (ERFANI; UTYUZHNIKOV, 2011). This methods constructs several scalarizations in which each one of them yields a Pareto-optimal solution. The scalarizations of the NBI, NC and DSD methods are constructed with the target of obtaining evenly distributed Pareto points that give a good evenly distributed approximation of the real set of Pareto points. Figure 2.4 displays the posteriori method.

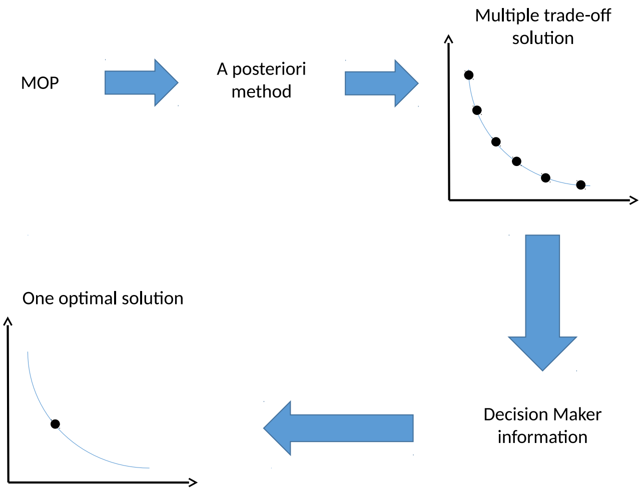


Figure 2.4: A pipeline displaying a posteriori method.

2.2.2.4 Interactive methods

The DM interacts with the method when searching for the most preferred solution at each iteration in order to obtain the Pareto optimal solution. The DM may stop the search whenever he wants to. Three types of preference information can be identified in the interactive method: (i) trade-off information; (ii) reference points and (iii) classification of objective functions. In the trade-off information, at each iteration several objective trade-offs is presented to the DM and he is expected to say which one he likes, dislikes or is indifferent with respect to each trade-off. In reference point method, the DM is expected at each iteration to specify a reference point consisting of desired values for each objective and a corresponding Pareto-optimal solution is then computed for analysis. In classification of objective functions, the DM is assumed to give preferences in the form of classifying objectives at the current Pareto-optimal solution into different classes indicating how the values of the objectives should be changed to get a more preferred solution. Among the well-knowns interactive methods, we can cite: Satisficing trade-off method (STOM) (NAKAYAMA; SAWARAGI, 1984), NIMBUS method (MIETTINEN; MäKELä, 1995).

2.3 Meta-heuristic Multi-objective Algorithms Applied to Engineering

In this section, we present some works related to multi-objective optimization in the context of engineering problems using meta-heuristic algorithms. Hybrid Flowshop Scheduling (HFS), firstly proposed by Arthanari and Ramamurthy (ARTHANARI; RAMAMURTHY, 1971), are generalization of flowshops problems combined with parallel machines in some stages. HFS cannot be solved by exact algorithms due to the complexity (NP-Hard). Hence, some meta-heuristics algorithms have been developed to handle with HFS problems. Tang and Wang (WANG; TANG, 2009) have proposed the tabu search to solve HFS problems. Genetic Algorithm is a widely used meta-heuristics algorithm to solve the HFS problem (şERIFOğLU; ULUSOY, 2004; OğUZ; ERCAN, 2005; RUIZ; MAROTO, 2006; BELKADI; GOURGAND; BENYETTOU, 2006; SHIAU; CHENG; HUANG, 2008; RASHIDI; JAHANDAR; ZANDIEH, 2010). Among the meta-heuristics algorithms employed to handle with HFS problem we can highlight ant colony optimization (KAHRAMAN et al., 2010; KHALOULI; GHEDJATI; HAMZAOUI, 2010), particle swarm optimization (TSENG; LIAO, 2008; TANG; WANG, 2010; SINGH; MAHAPATRA, 2012) and simulated annealing (WANG; CHOU; WU, 2011).

Burdening process of copper strip production is a complex industrial process in which interrelated factors are multiple. In addition to making the proportion of elemental composition within standard range to ensure product quality, many factors, including cost of raw materials, feeding sequence, stocks, original fused mass, burning loss of raw materials for in the smelting process and the maximizing use of waste material, have to be considered. Zhang et al. (ZHANG et al., 2012) have employed a multi-objective version of artificial bee colony to minimize the total cost of raw materials and to maximize the amount of waste material thrown into meting furnace. The results have shown the proposed technique performed better than NSGA-II and MOPSO. Omkar et al. (OMKAR et al., 2011) modified the Vector Evaluated Artificial Bee Colony (VEABC) algorithm to multi-objective design optimization of laminated composite components. Two objectives are formulated being minimizing the weight and the total cost of the composite component to achieve a specified strength. The performance was compared against with PSO, Artificial Immune System (AIS) and GA and the results have shown the proposed VEABC for composite structures performed satisfactorily. Pelletier et al. (PELLETIER; VEL, 2006) also have discussed the multi-objective design of symmetrically laminated plates of different criteria like strength, stiffness and minimal mass employing a multi-objective genetic algorithm.

Zhang and Mahfouf (ZHANG; MAHFOUF, 2007) proposed an optimization algorithm called Reduced Space Searching Algorithm (RSSA), inspired by the simple human experience when searching for an optimal solution. In (ZHANG; MAHFOUF, 2010), a multi-objective version of RSSA is formulated and applied on a real industrial problem relating to the optimal design of mechanical properties of alloy steel.

Vehicle routing problems is widely studied due to real-life applications, Jozefowiez et al. (JOZEFOWIEZ; SEMET; TALBI, 2008) have proposed a survey on existing works about routing problems. Lau et al. (LAU et al., 2013) have applied the multi-objective memetic algorithm in order to handle vehicle resource allocation on sustainable transportation planning.

On the reliability redundancy allocation problem (RAP) (MISRA; LJUBOJEVIC, 1973), Gen and Yun surveyed various reliability problems using GA (GEN; YUN, 2006). Konak et al. (KONAK; COIT; SMITH, 2006) developed an overview for multi-objective RAP problems and Li et al. (LI; LIAO; COIT, 2009) proposed a two-stage multi-objective approach to handle reliability optimization problems. Kalili-Damghani et al. (KHALILI-DAMGHANI; ABTAHI; TAVANA, 2013b) proposed a multi-objective PSO to solve a multi-objective binary-state RAP. A survey on multi-objective structural design problems focusing on the optimization of topology, shape, and sizing of civil engineering structures is reviewed on (ZAVALA et al., 2014).

2.4 Meta-heuristic Multi-objective Algorithms Applied to Machine Learning

In this section, we present some works related to multi-objective meta-heuristic algorithms employed to solve some optimization problems in the context of computer vision and pattern recognition. Such works are divided into three categories: (i) feature extraction and selection, (ii) supervised classification and (iii) unsupervised learning.

2.4.1 Feature Extraction and Selection

The reader can refer to some interesting works regarding feature engineering. Radtke et al. (RADTKE; WONG; SABOURIN, 2005), for instance, proposed a Multi-objective Memetic Algorithm for the feature extraction of isolated handwritten symbols. Zhang and Rockett (ZHANG; ROCKETT, 2007) proposed a multi-dimensional mapping strategy using a Multi-Objective Genetic Programming (MOGP) approach to extract the near-optimal number of features regardless of the domain-specific knowledge (ZHANG; ROCKETT, 2006). This method was compared against with eight other classifiers in eight UCI and Statlog benchmark datasets, being the proposed method more accurate due to its optimized feature extraction process. Albukhanajer et al. (ALBUKHANAJER et al., 2012) developed a feature extraction algorithm by adapting the Non-dominated Sorting Genetic Algorithm (NSGA-II) in order to select the best combination of functionals and the optimal number of projections in the Trace Transform to achieve the optimal triple feature. Consisting by a generalization of the Radon Transform, Trace Transform maps an image to another domain by tracing the image with straight lines for further calculating a functional “Trace”over the pixel value along the straight lines (PETROU; KADYROV, 1998). Further, a second functional called “Diametric”is applied along the columns of the Trace transform to produce a string of numbers, and finally a third functional called “Circus”is applied on the final string of numbers to generate a scalar value, also denoted as triple feature. The between-class variance and within-class variance are used as two objectives to be optimized. The proposed Evolutionary Trace Transform (ETT) algorithm was evaluated on images from fish dataset. Further, Albukhanajer et al. (ALBUKHANAJER; BRIFFA; JIN, 2014) used the same idea to extract features from noise images.

Morita et al. (MORITA et al., 2003) proposed to use NSGA-II to handle feature selection using multi-objective optimization in unsupervised learning in the context of handwritten word recognition. Two criteria were considered: (i) the minimization of the number of features, and (ii) the minimization of a validity index that measures the quality of clusters. Hamdani et al. (HAMDANI et al., 2007) used Non-dominated Genetic Algorithm (NSGA-II) to minimize the number of features and the classification error of the nearest neighbor classifier. Spolaor et al. (SPOLAOR; LORENA; LEE, 2010) applied Multi-Objective Genetic Algorithm (MOGA) with ten different combinations of criteria measuring the importance of the subsets of features. Measures regarding to consistency, dependency, distance and information are used for such purpose. The experimental results showed the combinations obtained by the proposed approach led to models with reduced numbers of features and good accuracy rates. Vatolkin et al. (VATOLKIN; PREUSS; RUDOLPH, 2011) applied S-Metric Selection Evolutionary Multiobjective Algorithm (SMS-EMOA) (BEUME; NAUJOKS; EMMERICH, 2007) for music recognition purposes, which uses the hypervolume quality measure combined with the concept of non-dominated sorting as selection operator. The hypervolume measure is a quality indicator that measures the size of the space covered by the solution points on the Pareto front. SMS-EMOA was evaluated in the context of music genre and style recognition with two different sets of objectives: (i) to optimize recall and specificity; (ii) to optimize the accuracy of the selected features. Xue et al. (XUE; ZHANG; BROWNE, 2013) proposed two multi-objective versions of the well-known Binary Particle Swarm Optimization (BPSO), being one based on mutual information, and the another based on entropy. The results in six benchmark datasets have shown the proposed approaches evolved to Pareto front.

2.4.2 Supervised Learning

Liu and Kadirkamanathan (LIU; KADIRKAMANATHAN, 1995) are one the seminal works that employed the concept of multi-objective optimization in the context of supervised learning. They minimized two error measures ($L_2 - norm$ and $L_\infty - norm$) and one complexity measure (the number of nonzero elements) of a Volterra Polynomial Basis Function (VPBF) network and a Gaussian Radial Basis Function (GRBF) network using the min-max approach. Kottathra and Attikiouzel (KOTTATHRA; ATTIKIOUZEL, 1996) formulated the training process of a multilayer perceptron network (MLP) as a multi-objective problem. The mean square error (MSE) and the number of hidden nodes of the network were optimized using the branch-and-bound algorithm. García-Pedrajas et al. (GARCÍA-PEDRAJAS; HERVÁS-MARTÍNEZ; PÉREZ, 2002) presented a cooperative coevolutive model to handle multi-objective optimization problems, namely MOBNET. This algorithm is an adaptation of NSGA to evolutionary programming, where the main idea is to evolve a population of subnetworks or modules and a population of networks, concurrently. The population of networks is formed by combined subcomponents in order to generate a final solution to a given task. This subcomponents, when used in a cooperative coevolutive model, carry out different criteria usually conflicting with each other. Similar to (KOTTATHRA; ATTIKIOUZEL, 1996), Abbass (ABBASS, 2003) proposed an evolutionary optimization algorithm (MPANN) to simultaneously decrease the training error rate and to optimize the neural network architecture in order to decrease the computational cost. Wiegand et al. (WIEGAND; IGEL; HANDMANN, 2004) proposed to optimize weight and execution time of an artificial neural network in the context of face detection based on NSGA-II. The approach performed well in reducing the number of hidden neurons without loosing the detection accuracy. Jin (JIN, 2004) used a multi-objective optimization approach to handle with the regularization problem, in which a hyperparameter determines how much the regularization influences the learning algorithm in order to prevent the training overfitting. Two multi-objective algorithms were employed to optimize the structure and parameters of the neural network, being them: Dynamic Weighting Aggregation (DWA) and NSGA-II, where the later algorithm performed slightly better when the population size is large.

Fieldsend and Singh (FIELDSEND; SINGH, 2002) used multiobjective neural network regression models in the financial time-series forecasting domain, where the idea was to minimize the risk and to maximize the return. Later on (FIELDSEND; SINGH, 2005), they derived a new methodology called Pareto Evolutionary Neural Network (Pareto-ENN), which evolves a population of ENN models maintaining a set of Pareto solutions. This methodology was evaluated in 37 regression datasets showing the set of ENNs can perform well on unseen test data. Hatanaka et al. (HATANAKA; KONDO; UOSAKI, 2003) proposed a multi-objective structure selection for RBF networks based on MOGA, in which the model accuracy and complexity were the objectives to be optimized. Yusiong and Naval Jr. (YUSIONG; JR., 2006) used the Multiobjective Particle Swarm Optimization-Crowding Distance (MOPSO-CD) to optimize the architecture and the weights of a recurrent neural network. Pettersson et al. (PETTERSSON; CHAKRABORTI; SAXéN, 2007) utilized a Multi-Objective Genetic Algorithm in the training process of a feedforward neural network. A noisy data from an industrial iron blast furnace was used, and a Pareto front was reached optimizing the training error along with the network architecture. Liu e Deng (LIU; DENG, 2010) used the multi-objective technique to optimize initial weights and thresholds of an artificial neural network. The results showed better precision and lower computational cost compared to the literature in the context of detecting carcinoma in mammographic images. Smith and Jin (SMITH; JIN, 2014) presented a hybrid Multi-Objective Evolutionary Algorithm to optimize the training error and the number of connections of the recurrent neural network in the context of time series prediction. Also, methods of selecting prediction models from the Pareto set of solutions are showed and compared. The first method selects all solutions below a threshold, and the second method is based on the training error. Finally, the last method selects solutions based on the diversity of the predictors.

Regarding model selection, Igel and Suttorp (IGEL, 2005; SUTTORP; IGEL, 2006) employed multi-objective evolutionary algorithms to optimizate the Support Vector Machine (SVM) performance and complexity. Miranda et al. (MIRANDA et al., 2012) used Multi-objective Particle Swarm Optimization (MOPSO) to maximize the hit rate and minimize the number of SVM's support vectors. Li et al. (LI; LIU; GONG, 2011) developed a Multi-Objective Uniform Design tool (MOUD), in which the gradient-based search is replaced by an Uniform Design (UD) method to reduce the model selection computational cost, and also to improve the classification accuracy. In this work, the authors applied MOUD in the UCI benchmark datasets and two face databases showing the proposed method outperforms other models. Aydin et al. (AYDIN; KARAKOSE; AKIN, 2011) employed a multi-objective Artificial Immune System (AIS), which is based on the body self-defence method against foreign antigens or pathogens in the context of fault diagnosis of induction motors and anomaly detection. The proposed algorithm uses an evaluation function that maximizes the accuracy rate and minimizes the support vectors. Further, Rosales-Pérez et al. (ROSALES-PÉREZ et al., 2014) used a multi-objective approach to handle model type selection, where the training error and model complexity are considered as objectives. The model complexity was estimated through the Vapnik–Chervonenkis dimension (VAPNIK, 1995), and the results on benchmarks datasets showed the proposed method generated competitive models, thus reducing the overfitting. You et al. (YOU; BENITEZ-QUIROZ; MARTINEZ, 2014) address a kernel optimization problem by minimizing the model fitness and complexity using two new measures in the context of regression. Also, a new ε-constraint method is derived to obtain better Pareto-optimal solutions, and the results have shown the new approach achieved the lowest mean square error.

2.4.3 Unsupervised Learning

A survey on multi-objective clustering is reviewed by Mukhopadhyay (MUKHOPADHYAY; MAULIK; BANDYOPADHYAY, 2015). Handl and Knowles (HANDL; KNOWLES, 2004) developed an algorithm based on Pareto Envelope-based Selection (PESA-

II) (CORNE et al., 2001) known as VIENNA to handle clustering multi-objective optimization, where the main ideia is to optimize the cluster variance and connectivity. The proposed method was compared with k-means and average-link agglomerative clustering technique. Qian et al. (QIAN et al., 2008) proposed a novel Multi-Objective Evolutionary Clustering Ensemble Algorithm (MECEA) to perform texture-based image segmentation. The proposed approach comprises two main phases: (i) to obtain a set of Pareto solutions using MECEA to optimize two objectives: the compactness in the same cluster and the connectedness of different clusters; (ii) to make use of the Meta-Clustering Algorithm (MCLA) to combine all the Pareto solutions to obtain the best texture segmentation. Abdul Latiff et al. (LATIFF et al., 2008) presented a dynamic clustering method using a multi-objective version of PSO applied to wireless sensor networks, where the main idea is to lenghten the network lifetime and to prevent connectivity degradation. Saha and Bandyopadhyay (SAHA; BANDYOPADHYAY, 2013) proposed a new multi-objective clustering method (GenClustMOO), in which each cluster is partitioned into small hyperspherical sub-clusters, and the centers of all these small sub-clusters are enconded in a string to represent the whole clustering. A multi-objective algorithm known as Archived Multi-Objective Simulated Annealing (AMOA) (BANDYOPADHYAY et al., 2008) is used to optimize three objective functions: (i) total compacteness; (ii) total symmetry of the clusters and (iii) cluster connectedness.

Nakib et al. (NAKIB; OULHADJ; SIARRY, 2008) designed a new segmentation technique adapting the simulated annealing to deal with multi-objective optimization using the weighted sum method. Two objective functions are considered: (i) the modified within-class variance and (ii) the overall probability of error. The results showed the method performed efficiently when compared with Otsu's method and Gaussian curve fitting. Shirakawa and Nagao (SHIRAKAWA; NAGAO, 2009) proposed an evolutionary image segmentation method based on multiobjective clustering using Strength Pareto Evolutionary Algorithm 2 (SPEA2), where overall deviation and edge values are optimized simultaneously. Further, Nakib et al. (NAKIB; OULHADJ; SIARRY, 2010) adapted the NSGA-II to optimize several image thresholding criteria. The method was evaluated on different types of images, being the results quite efficient.

2.5 Conclusions

This paper presents a survey on multi-objective meta-heuristic optimization applied to machine learning. Machine learning techniques are naturally modeled as a multi-objective optimization task, since most learning algorithms require some sort of model selection and parameter estimation. Thus, we addressed three different topics in this context: (i) feature extraction and selection; (ii) supervised learning; and (ii) unsupervised learning. We have observed the major effort has been applied to supervised learning, specifically to artificial neural networks, where the common objectives are the training error rate and the network architecture. The idea of using two or more objectives is to avoid under/overfitting in the training phase, thus allowing the model to generalize better on unseen data and to decrease the computational cost.

Chapter 3

On the Study of Commercial Losses in Brazil: A Binary Black Hole Algorithm for Theft Characterization

In this chapter, it was proposed to use the Black Hole Algorithm technique to characterize illegal consumers. This article was published in the *IEEE Transactions on Smart Grid* (RAMOS et al., 2016).

3.1 Introduction

The energy losses are defined as the difference between the energy generated or purchased and the energy billed, being classified in two different types: technical and commercial losses. The former are related to the physical characteristics of the energy system, i.e., the technical losses are defined as the energy lost in the transportation, transformation and in the measuring equipments, being its costs predicted by the electric utilities (OLIVEIRA; BOSON; PADILHA-FELTRIN, 2008). The commercial losses, also called non-technical losses (NTL), are associated with the energy delivered to the consumer that is not billed, being more difficult to be detected and quantified.

The problem of commercial losses is not faced only in emerging countries, but worldwide, even in smaller proportions. Some experts tend to correlate commercial losses with the development of a given country, which may also include aspects of education, income distribution and violence, among others. The percentage for commercial losses rates vary between 0.5% and 25% in Brazil, 20% and 40% in India, 0.2% and 1% in United Kingdom and around 3.5% in Philippines (RODRIGUES et al., 2015; ANEEL, 2011; PARUCHURI;

DUBEY, 2012; MILLARD; EMMERTON, 2009).

Brazil, for instance, is the largest economy in Latin America and the seventh in the world, being also a member of the BRICS group (Brazil, Russia, India, China and South Africa) that stands for the most prominent developing countries, in other words, they refer to the large emerging markets. It is clear that, as the Brazilian economy grows, the consumption of electricity also increases, bringing together very high losses rates (around 17.5%) (EPE, 2014).

Lately, the problem of detecting commercial losses in power systems is a topic that has been extensively researched (RODRIGUES et al., 2015; HUANG; LO; LU, 2013; PORRAS et al., 2015). Theft and tampering of power meters in order to adulterate the measurement of power consumption are the main causes that lead to commercial losses in electric power companies (ALVES et al., 2006). Additionally, performing periodic inspections to minimize such frauds can become very costly in some cases, considering that it is a difficult task to calculate or even measure the amount of losses, and in most cases is almost impossible to know where they occur (RAMOS et al., 2009).

Aiming to reduce fraud and electricity theft, several electric power companies have been interested to characterize a profile of irregular consumers, which are mainly related to the illegal electrical installations. The minimization of losses can assure investments in quality programs of energy, as well as it can allow a reduction of the final energy price to the consumer. Nowadays, some advances in this research field can be noted with the use of several artificial intelligence techniques in order to automatically detect commercial losses, which is a real application in smart grids. It is also important to note that the commercial losses are a global issue and the solution to this problem is not trivial (RODRIGUES et al., 2015; RAMOS et al., 2009).

Despite the extensive use of machine learning techniques for the detection of commercial losses in power systems, the problem of selecting the most representative features has not been widely discussed in the context of commercial losses (NAGI et al., 2010; NIZAR; DONG; WANG, 2008; RAMOS et al., 2011; MONEDERO et al., 2006; NAGI et al., 2011). In regard to the cutting edge research in this field, Nizar et al. (NIZAR; ZHAO; DONG, 2006) proposed a work to select a subset of samples in order to improve the identification of irregular consumers, and Ramos et al. (RAMOS et al., 2011) proposed a new hybrid feature selection algorithm based on Harmony Search and Optimum-Path Forest. Further, Ramos et al. (RAMOS et al., 2012) presented a new methodology based on evolutionary algorithms to the same purpose. Therefore, to point the subset of the most discriminative features to design more effective systems for commercial losses detection is as important as the detection of such losses.

In the last years, there has been an increasing number of researches that concern the problem of feature selection as an optimization task, which can be addressed by means of meta-heuristic and swarm-based techniques. There are a plenty of them, being the most known the Particle Swarm Optimization (PSO) (KENNEDY; EBERHART, 2001), Differential Evolution (DE) (STORN; PRICE, 1997), Genetic Algorithm (GA) (KOZA, 1992) and Harmony Search (HS) (GEEM, 2009), among others. The "No Free Lunch Theorem" (WOLPERT; MACREADY, 1997), which states there is not a single optimization approach that outperforms another one for all optimization problems, may contribute with the development of such new algorithms every time.

Recently, an interesting approach presented by Hatamlou (HATAMLOU, 2013) for data optimization called Black Hole Algorithm (BHA), that is based on the formation of the well-known black holes in the universe and their attraction power. This approach has demonstrated interesting results in the context of continuous-valued optimization problems. However, to the best of our knowledge, only we have been applied the new technique based on BHA to the context of feature selection, although a binary-constrained optimization version of BHA have been presented by Nemati et al. (NEMATI; MOMENI; BAZRKAR, 2013) as well.

This paper brings the problem of commercial losses in Brazil and to highlight the use of intelligent computational tools by electric power companies aiming the revenue recovery. Therefore, the main contributions of this paper are two fold: (i) to shed light over the problem of commercial losses in Brazil focusing on the past years, as well as (ii) to present a novel binary optimization algorithm based on the BHA for irregular consumers characterization. The remainder of this paper is organized as follows. Section 3.2 presents the context of commercial losses in Brazil, and Section 3.3 states the theoretical aspects of BHA. Section 3.4 presents a case study with respect to theft characterization, and finally the conclusions are presented in Section 3.6.

3.2 Commercial Losses in Brazil

In Brazil, according to ANEEL (The Brazilian Electricity Regulatory Agency) (ANEEL, 2011), the losses with energy theft (irregular consumption) have reached the level of R$ 8.1 billion (approximately US$ 4 billion) per year, considering 61 of the 63 utilities who passed the second tariff review cycle in the period from 2007 to 2010. In terms of energy, this amount corresponds to more than 27,000 Gigawatt-hours (GWh), approximately 8% of the consumption in the Brazilian energy captive market, which is formed by consumers that can only buy energy from a distribution utility that operates in the network they are connected.

In order to clarify the problem of energy theft in Brazil, Table 3.1 (ANEEL, 2011) shows the amount of commercial losses for each Brazilian region. The largest amount of losses can be observed in North region, where the implementation of procedures for handling technical and commercial losses are not easy to be performed, mainly due to the difficult access and the large territory in which the electric utility operates (North region is the largest one in Brazil). On the other hand, the smallest losses occur in the South, an opposite scenario of what can be observed in the northern region. In Southeast and Northeast, some utilities suffer with energy theft motivated by the large number of slums presented in states such as Rio de Janeiro, São Paulo, Bahia and Pernambuco. In Midwest and South, where the number of slums is smaller, the occurrence of frauds is more usual in the unit of metering of each consumer.

Table 3.1: Amount of commercial losses in each region of Brazil (Source: ANEEL-2011)

Placing	Region	Commercial Losses (%)
1	North	20%
2	Southeast	10%
3	Northeast	9%
4	Midwest	5%
5	South	3%

The amount of illegal connections in Brazil and the rates regarding them give us an idea of the magnitude of the problem and the degree of difficulty to compute such losses. According to Table 3.2 (ANEEL, 2011), which shows a list of the 15 utilities with the largest commercial losses rates[1], the Centrais Elétricas do Pará (CELPA) leads the ranking with 24.4% of the distributed energy, followed by LIGHT company, located in Rio de Janeiro, where the losses reach 24.2% of the distributed energy. Finally, Centrais Elétricas de Rondônia (CERON) takes the third place with 22%. These three companies are located in North and Southeast regions, corroborating the data available in Table 3.1. The main issue concerning commercial losses is related to the impact in the energy tariff, since this

[1] The commercial losses rates below 10% are tolerated by ANEEL. According to Millard and Emmerton (MILLARD; EMMERTON, 2009), the commercial losses rates in Brazil were between 0.5% and 25.0% in 2007.

kind of loss ends up being divided among legal consumers registered in the electric utility at the time of tariff calculation. In a concession area such as of the LIGHT, for example, the tariff reduction would be of 18% if there was no irregular consumption (ANEEL, 2011).

Table 3.2: Commercial losses rates of electric utilities in Brazil (Source: ANEEL-2011)

Placing	Utilities	Commercial Losses (%)
1	CELPA	24.4%
2	LIGHT	24.2%
3	CERON	22.0%
4	CEMAR	17.8%
5	AMPLA	17.1%
6	CEAL	17.0%
7	AMAZONAS ENERGIA	16.8%
8	ELETROACRE	15.9%
9	CEPISA	15.8%
10	ENERGISA PARAÍBA	11.2%
11	ELETROPAULO	10.8%
12	CEEE	10.5%
13	BANDEIRANTE	10.1%
14	ESCELSA	10.0%
15	BOA VISTA	10.0%

3.2.1 Procedures to Combat Commercial Losses

The problem of commercial losses detection in distribution systems has been decisive. Theft and tampering of energy meters in order to modify the measurement of energy consumption are the main causes of commercial losses in electric utilities. Therefore, to calculate or even measure the amount of these tasks has been a difficult task. In most cases, it is almost impossible to know where they occur.

Aiming to reduce the rates concerning commercial losses, the electric utilities usually operate in the following preventing programs:

- Inspection Programs: they consist in verifying the integrity of the measurement system, to detect equipment failures, frauds and energy thefts, connection errors and other problems that may compromise the measurement of electric energy;
- Replacement of energy meters: it consists in the assessment of energy meters through field sampling, laboratory testing and analysis of the energy meters removed in the

field. In addition, the replacement of energy meters with service life expired or possible technical failures is also performed;

- Regularization of illegal connections: especially in slums, through a regulatory program to reduce commercial losses;
- Implementation of trade policies: it consists in giving attention to the community about explanations, agreements and trainings in healthy energy consumption; and

One of the most traditional ways to combat commercial losses is to perform periodic inspections of consumers, which is not very advantageous, since such task has high costs to the electric utility. Additionally, the selection of consumers that must be inspected is an arduous task, even for experts. In the last decade, the electric utilities have invested much effort in a set of heuristic methods to automatic recognize illegal consumers by means of artificial intelligence-based techniques, which is the main focus if this paper.

3.2.2 Smart Grid and Its Relation to Commercial Losses

The problem of commercial losses may be minimized in the nearby future by means of smart metering resources. In addition to control consumption in real time, it is possible to collect more electrical information through sensors in energy meters, thus enabling a better understanding of the consumer behavior, and to transmit them with certain periodicity to the utilities. This can be seen as an advance with respect to the measurement process, but without employing the concept of machine learning for taking decisions autonomously. In other words, the smart meter does not prevent the fraud, but only provides information more quickly (FANG et al., 2012; GUARRACINO et al., 2012; RODRIGUES et al., 2015).

The computational intelligence aims to point out where is more likely to happen a fraud or irregularity, as well as what is its importance level through a priority criterion, followed by a field inspection. Thus, smart meters are extremely useful to improve the performance of the network, and also to reduce the commercial losses (FANG et al., 2012).

The concept of “Smart Grid”, using smart meters, allows the integration of electrical equipment with data communication networks in a managed and automated system by the electric utility, making the energy to be supplied with safety, reliability and efficiency. Therefore, a network can enable (FANG et al., 2012; GUARRACINO et al., 2012; RODRIGUES et al., 2015):

- Smart services integrated with consumers;

- The use of smart meters and the application of differentiated tariffs by the time or the seasonality;
- Improvement of power quality and reduction of technical and commercial losses;
- Management, monitoring and optimization of the energy system in real time;
- Surveillance and security;
- Integration of public services; and
- Broadband Internet.

The concept of Smart Grid has increased with the demand growth for automatic reading of energy meters. Beyond the aim of reducing frauds, thefts of energy and faulty measurements, several electric utilities have been concerned to better characterize the profile of consumers with irregularities, and also to correctly identify illegal connections (FANG et al., 2012; GUARRACINO et al., 2012). Therefore, minimizing commercial losses may guarantee investments in programs for power quality, and may allow a reduction of the price to the consumer.

3.3 Black Hole Algorithm

The Black Hole Algorithm is a population-based metaheuristic algorithm based on the black hole's gravitational force proposed by Hatamlou (HATAMLOU, 2013). The candidate solutions (stars) are initialized at random positions onto the search space $\vec{x}_i \in \Re^n$ with $i = 1, 2, \ldots, m$, where n and m stand for the number of design variables and the number of stars, respectively. The objective function value of all stars are computed and the best star in the population, i.e., the one which holds the best objective function value, is selected to be the black hole.

All stars move toward the black hole due to its gravitational force absorbing everything that is around. As such, each star position is updated as follows:

$$\vec{x}_i^{(t+1)} = \vec{x}_i^t + \sigma(\vec{x}^* - \vec{x}_i^t), \tag{3.1}$$

where $\vec{x}_i^t$ is the location of the i-th star at iteration t, $\vec{x}^*$ is the location of the black hole in the search space, and $\sigma \sim U(0,1)$. Notice that σ is different for each star and iteration. As the stars move toward the black hole, their positions keep changing and, consequently, it is assumed that their objective function value are getting better. If a star reaches a location with lower objective function value than the black hole, the star become the new black hole and all the other stars start to move toward the new black hole.

A sphere-shaped boundary known as “event horizon”surrounds the black hole swallowing everything that comes close. Each star that crosses such boundary will be sucked by the black hole, and a new star borns randomly in the search space to start a new search. The radius r of the event horizon in BHA is calculated using the following formulation:

$$r = \frac{f^*}{\sum_{i=1}^{m} f_i}, \tag{3.2}$$

where f^* and f_i stands for the objective function values of the black hole and of the i-th star, respectively. When the distance between a star and the black hole is less than r, that star is collapsed and a new star is created. The next iteration takes place after all stars have been moved. Roughly speaking, the idea of BHA is to guarantee diversity in the population when creating new black holes, as well as to avoid stars getting trapped from local optima. This mechanism ends up contributing with both exploitation (local) and exploration (global) searches.

Unlike the standard BHA, in which the solutions are updated in the search space towards continuous-valued positions, in the proposed Binary Black Hole Algorithm (BBHA), the search space is modelled as an n-dimensional boolean lattice and the solutions are updated across the corners of a hypercube. In addition, as the problem is to select or not a given feature, a solution binary vector is employed, where 1 corresponds whether a feature will be selected to compose the new dataset, and 0 otherwise. In order to design this binary vector, we employed Equation (3.4), which can restrict the new solutions to only binary values:

$$S(x_{ij}^t) = \frac{1}{1+e^{-x_{ij}^t}}, \tag{3.3}$$

$$x_{ij}^t = \begin{cases} 1 & \text{if } S(x_{ij}^t) > \gamma, \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

in which $\gamma \sim U(0,1)$ and x_{ij}^t stands for the j-th decision variable of the i-th star at iteration t. This approach is a slight variation of the one proposed by Nemati et al. (NEMATI; MOMENI; BAZRKAR, 2013). Notice each variable to be optimized stands for one feature extracted from a given consumer (Section 3.5.1). Therefore, each star models a binary-valued solution vector that indicates whether a feature will be selected or not to compose the final dataset.

3.4 Case Study

Very often, the literature addresses commercial losses using an exhaustive search in spreadsheets in order to compare the historical consumption of thousands of consumers by hand. However, this procedure is time- and money-consuming, thus being interesting to make use of computational tools for accomplishing this task in a more efficient way.

The computational tools are often implemented using artificial intelligent in the context of machine learning research field. The theme addressed in this research does not only involve pattern recognition, but also optimization tasks, i.e., intelligent techniques are applied for optimization purposes considering feature selection purposes, aiming at characterizing the consumer profile to minimize commercial losses. Such optimization techniques can evidence the process of learning the behavior and characterization of potential consumers with irregularities. In this work, we validated the proposed technique for feature selection based on BHA against with PSO, HS, DE and GA in the context of theft characterization, as well as we provide an economic study based on the results in the Brazilian energy market.

The electric utilities usually look for methods more financially viable, being an affordable solution to employ softwares to support decision-making processes in face of thousands of consumers, pointing out those who may have some kind of error in their measurements. In other words, the software helps reducing the number of inspections, checking consumers suspected with irregularities and avoiding unnecessary inspections. Thus, it is possible to reduce costs with periodic random inspections, since this procedure will determine what may be the cause of the commercial loss, making sure the consumer is committing some kind of fraud or if the energy meter is reading the measurements correctly. Therefore, the utility can decide the best kind of providence that it should be taken to solve the problem quickly and effectively. Moreover, the utility will have a revenue recovery, because the irregular consumers will come back to be regular again, and they will return to properly pay for the consumed energy.

3.5 Methodology and Experimental Results

In this section, we present the experimental evaluation used to assess the effectiveness of BHA in the context of theft characterization (Section 3.5.3), as well as we also discussed the impact with respect to the application of such techniques (Section 3.5.4).

3.5.1 Datasets

A Brazilian electric utility has provided two private datasets, being one with 3,182 profiles of industrial consumers and the other with 4,952 profiles of commercial consumers, represented by eight features:

1. Demand Billed (DB): demand value of the active power considered for billing purposes, in kilowatts (kW);
2. Demand Contracted (DC): the value of the demand for continuous availability requested from the electric utility, which must be paid whether the electric power is used by the consumer or not, in kilowatts (kW);
3. Demand Measured or Maximum Demand (D$_{max}$): the maximum actual demand for active power, verified by measurement at fifteen-minute intervals during the billing period, in kilowatts (kW);
4. Reactive Energy (RE): energy that flows through the electric and magnetic fields of an AC system, in kilovolt-amperes reactive hours (kVArh);
5. Power Transformer (PT): the power transformer installed for the consumers, in kilovolt-amperes (kVA);
6. Power Factor (PF): the ratio between the consumed active and apparent power in a circuit. The PF indicates the efficiency of a power distribution system;
7. Installed Power (P$_{inst}$): the sum of the nominal power of all electrical equipment installed and ready to operate at the consumer unit, in kilowatts (kW);
8. Load Factor (LF): the ratio between the average demand ($D_{average}$) and maximum demand (D_{max}) of the consumer unit. The LF is an index that shows how the electric energy is used in a rational way.

At every 15 minutes, the electric utility recorded consumption data during one year. After that, such technical data was used to compute the aforementioned monthly features for both datasets. However, the company did not inform what kind of irregularity was verified in each consumer.

3.5.2 Experimental Setup

Basically, we employed the very same procedures of our preliminary work (RODRIGUES et al., 2015) concerning the experiments. Roughly speaking, in order to deal with the stochastic behavior of the optimization techniques, we ended up partitioning the dataset into N folds, in which two of them were used as training and validating sets, and the remaining folds were merged together to compose the test set. Therefore, such procedure is repeated over N times, and a statistical evaluation can be performed. The validating set is used to guide the optimization techniques, since the idea is to select the minimal subset of features that allows the best recognition rates over the validating set. Notice the test set does not participate from the learning features step. In regard to the recognition rate, we employed an accuracy measure proposed by Papa et al. (PAPA; FALCÃO; SUZUKI, 2009), which considers unbalanced classes, such as the ones we faced in this work.

Although the reader can employ any supervised classification technique for the above procedure, in this paper we opted to employ the Optimum-Path Forest (OPF) classifier (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012), since it is a parameterless approach and it has obtained similar results to the ones achieved by some state-of-the-art pattern recognition techniques, being sometimes faster for training.

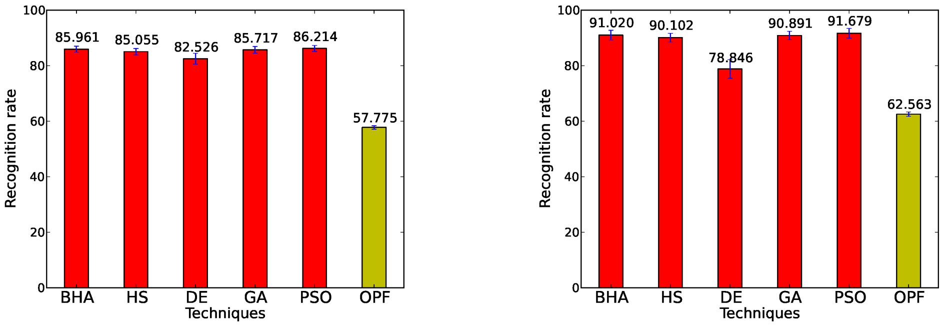
3.5.3 Theft Characterization

In this section, we present the results regarding to BHA, PSO, HS DE and GA techniques for theft characterization, i.e., we are interested to find out the most important set of features in order to identify possible illegal consumers. We employed $N = 5$ folds, a population of 30 candidate solutions (star/particle/harmony/chromosome) and 100 iterations for all techniques. The results presented in this section stand for the mean accuracy and standard deviation over 25 rounds using the methodology presented in Section 3.5.2. Since the meta-heuristic techniques used in this work are non-deterministic, such approaches seem to be robust to avoid biased results. Table 3.3 presents the parameters used for each optimization technique[2].

Table 3.3: Parameters employed for each optimization algorithm

Algorithm	Parameters
PSO	$c_1 = c_2 = 2.0$ and $w \in [0.4, 0.9]$
HS	$HMCR = 0.9$
DE	$f = 0.5$ and $c_r = 0.1$
GA	$p_m = 0.1$
BHA	-

The exploration and exploitation of the metaheuristic algorithms is controlled by user parameters. PSO uses c_1 and c_2 for the pace range control, which guides the particles toward their best local solution as well as to the best global solution of the swarm, respectively. Additionally, the amount of velocity that is going to be used to update the value of each possible solution for the next time step is controlled by the inertia weight w[3]. In regard to HS, $\boldsymbol{HMCR}$ stands for the Harmony Memory Considering Rate, which controls the amount of information extracted from the previously used values to compose new solutions. In regard to DE, f is called differential weight, which scales the influence of the set of pairs of solutions selected to calculate the mutation value, and c_r stands for crossover probability. Finally, GA's parameter p_m denotes the mutation probability. Figs. 3.1a and 3.1b depict the recognition rates over the commercial and industrial datasets, respectively. The "yellow" bar stands for standard OPF recognition rate, i.e., without feature selection.



(a) (b)

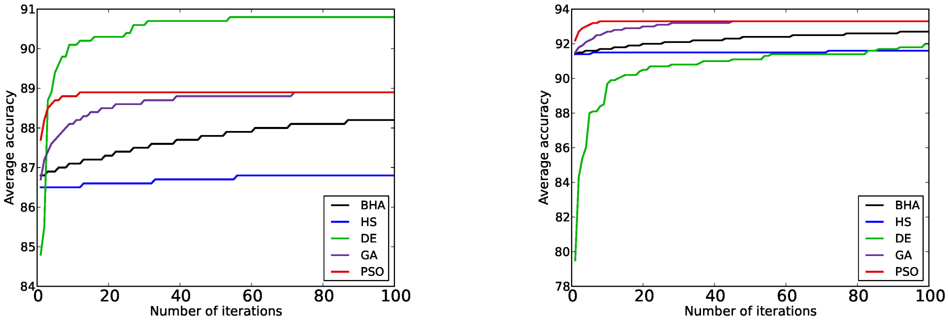
Figure 3.1: Mean recognition rates over (a) commercial and (b) industrial datasets.

Observing Figs. 3.1a and 3.1b, we can note a great improvement with respect to standard results (i.e. naïve OPF), being all techniques similar to each other if we consider

[2] The parameters have been empirically chosen.

[3] Parameter w has been dynamically adjusted within in interval [0.4,0.9].

their standard deviation. Such experiment demonstrated BHA is able to achieve results similar to those obtained by well-accepted meta-heuristic techniques in the literature, such as PSO, HS, DE and GA. Another experiment evaluated the convergence rates of each optimization technique, as displayed in Figs. 3.2a and 3.2b: it is clear PSO and GA have converged faster, but it did not reflect on the final classification results displayed in Fig. 3.1. Although naïve HS may be considered one of the fastest approaches, it often does not benefit from reasonable convergence rates, since it updates only one agent (harmony) at each iteration, while swarm-based techniques usually update all agents.



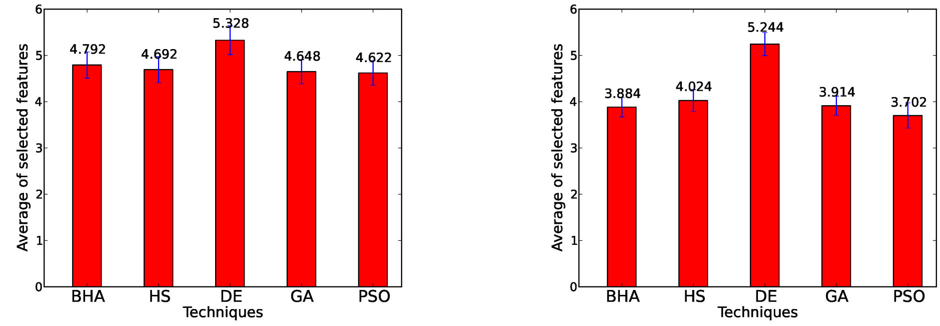
(a) (b)

Figure 3.2: Convergence rates over (a) commercial and (b) industrial datasets.

The average number of selected features for each optimization technique is shown in Figs. 3.3a and 3.3b concerning commercial and industrial profiles, respectively. Such experiment attempts to show that only 58.75% (on average) of the features really matter for illegal consumer recognition in case of commercial profiles, and 50% (on average) considering industrial dataset. We can observe the smallest number of features have been selected by PSO and BHA in this latter dataset.

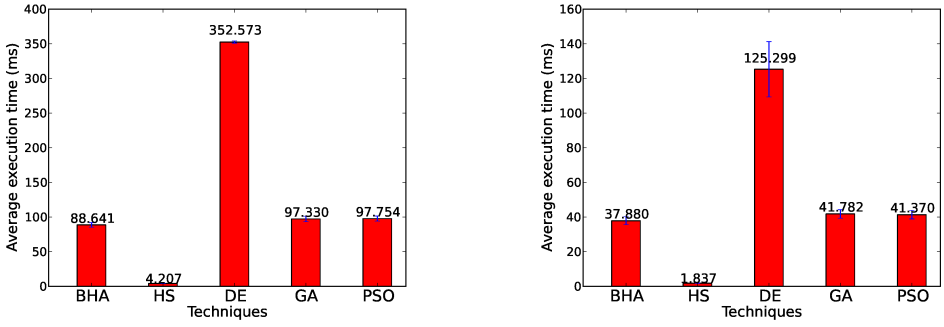
Figs. 3.4a and 3.4b present the computational load (ms) for commercial and industrial datasets, respectively[4]. Two groups of techniques can be observed here: HS and swarm and genetic-based ones, which are composed by BHA, DE, GA and PSO. The latter approaches usually evaluate the entire swarm in order to update all agents' position, while HS only updates one agent at each iteration. Therefore, if we consider the second group, we notice BHA has been the fastest approach. Another interesting skill of BHA is related to its absence of parameters, which is very interesting to avoid meta-optimization, turning the technique user-friendly and less prone to configuration errors.

[4] The experiments were executed on a computer with a Pentium Intel Core *i*7® 1.73Ghz processor, 6 GB of memory RAM and Linux Ubuntu Desktop LTS 13.04 as the operational system.



(a) (b)

Figure 3.3: Average number of selected features over (a) commercial and (b) industrial datasets.



(a) (b)

Figure 3.4: Mean execution time (ms) over (a) commercial and (b) industrial datasets.

Additionally, we have evaluated results using the Wilcoxon statistical test (HARRIS; HARDIN, 2013), as displayed in Table 3.4. The Wilcoxon test evaluates each pair of techniques in order to check whether they are similar to each other or not. The symbol '$\neq$' denotes there exists difference between the methods, and the symbol '$=$' represents the techniques are similar each other. Additionally, if the p-value (parenthesis) is less than the desired significance level, the techniques are considered different to each other. In this paper, we adopted **0.05** (**5%**) of significance. According to Table 3.4, PSO was the most accurate technique with respect to commercial dataset, followed by BHA. The same result can be evidenced for industrial dataset. However, the accuracy rates of PSO and BHA are very close to each other, being BHA faster and parameter-free, which makes it suitable for feature selection purposes.

Table 3.4: Wilcoxon Signed rank Test considering 5% of significance

Dataset	BHA/HS	BHA/DE	BHA/GA	BHA/PSO
Commercial	$\neq$ (0.0000)	$\neq$ (0.0000)	$\neq$ (0.0422)	$\neq$ (0.0054)
Industral	$\neq$ (0.0000)	$\neq$ (0.0000)	$=$ (0.3395)	$\neq$ (0.0000)

3.5.4 Impacts

In this section, two classes are considered for the experiment: (i) the "Regular Consumer", which represents consumers who are under regular conditions, and (ii) the "Irregular Consumer", which stands for potential consumers with irregularities. It is important to highlight the electric utility did not provide any further details about irregularities presented in each consumer, but they were previously confirmed by the technical staff of the electric utility. In this section, the mean accuracy was computed to verify a comparison between different classes, i.e. the also computed the recognition rates for each type of consumer (regular or irregular). Table 3.5 presents the accuracy rates per class without feature selection.

Table 3.5: Mean accuracy rates per class without feature selection

Dataset	Regular Consumer	Irregular Consumer
Commercial	95.01%	58.87%
Industrial	94.86%	64.14%

Observing Table 3.5, we can note that 58.87% of commercial consumers and 64.14% of the industrial consumers who had some kind of irregularity were identified correctly. Therefore, the regular consumers have higher recognition rates, probably because the dataset is biased on such class, i.e., we have much more regular consumers than irregularities in both datasets.

Table 3.6 presents the accuracy rates per class considering the feature selection by means of BHA. We can observe that 64.81% of commercial consumers and 83.76% of industrial consumers who had some kind of irregularity were identified correctly. Thus, the accuracy rates increased approximately 20% for industrial dataset and 6% for commercial dataset when compared with the vanilla results (without feature selection). Therefore, the results were quite optimistic.

Table 3.6: Mean accuracy rates per class with feature selection

Dataset	Regular Consumer	Irregular Consumer
Commercial	97.54%	64.81%
Industrial	98.94%	83.76%

Table 3.7 presents the selected features considering both datasets. Notice these features are extracted from a single execution of the algorithms, and may not reflect the final subset of features, since the experiments average the number of them. The same set of features have been chosen for both datasets, which highlights their importance.

Table 3.7: Selected features considering BHA

Features for Commercial	Features for Industrial
DC, D_{max}, PF, P_{inst}, LF	DC, D_{max}, PF, P_{inst}, LF

3.6 Conclusion

We presented here the context of commercial losses (non-technical losses) in Brazil, and discussed how this issue is recent. Note that in less developed regions, where the socio-economic aspects, such as education, income distribution and violence, among others, are very precarious, the rates of commercial losses are extremely high. The fraud and theft of energy are attitudes of many consumers unable to pay for the consumed energy, or even malicious behaviour in order to save money. Hence, there is a need for several ways to combat these commercial losses in order to not compromise the power system, thus generating electric energy with quality, as well as to possibly reduce the final energy price to consumers.

The development of intelligent computational tools has been widely pursued to contribute to the reduction of commercial losses, since these methods can be easily employed in smart grids, which will be deployed worldwide. However, these tools only assist in decision-making processes to indicate a potential consumer with irregularities, which is checked after an inspection conducted by the technical staff of the electric utility, i.e., they do not confirm whether the consumer is fraudster or not. The most works address only the identification or detection of commercial losses. In this paper, we are concerned about characterizing the profile of possible irregular consumers, i.e., we want to determine the most relevant features considering the context of the problem.

We also presented a case study to demonstrate the usefulness of the methodology described in this work which was conducted by means of meta-heuristic techniques and OPF classifier, as well as we have introduced BHA for feature selection purposes in the context of commercial losses in power systems.

Chapter 4

EEG-based Person Identification through Binary Flower Pollination Algorithm

In this chapter, the binary version of the Flower Pollination Algorithm technique was used for the task of channel selection in electroencephalogram tests. This work was published in the *Expert Systems with Applications* (RODRIGUES et al., 2016).

4.1 Introduction

In modern life, we constantly make use of passwords to access our bank accounts, e-mail boxes, and social networks, just to name a few. As passwords can be easily circumvented, the use of biometrics has been proposed for safe person identification (JAIN; ROSS; NANDAKUMAR, 2011). Over the years, the use of biometric systems has increased, and systems based on several biometric modalities such as fingerprint, face and iris, have been successfully deployed. This successful and widespread deployment of biometric systems brings on a new challenge: spoofing. Spoofing methods are developed to breach the security of biometric systems so that unauthorized users can gain access to places and/or information (e.g., an artificial finger made from silicone is placed on the fingerprint scanner).

In this scenario, the EEG (electroencephalogram) signal presents a great potential for highly secure biometric-based person identification, due to its characteristics of universality, uniqueness, and robustness to spoofing attacks (BEIJSTERVELDT; BOOMSMA, 1994). It is well-known the importance of EEG signals in several areas, since one can find a number of works that deal with such a source of data (SUBASI, 2007; GUO et al., 2011; OCAK, 2009; NUNES et al., 2014). In high security environments, EEG sensors can be integrated in order to contribute to the robustness of the system, and the person can be continuously authenticated. Although the idea of using EEG as a biometric trait is not new, there are a few works that address such kind of signal only. One possible explanation for that is the difficulty in obtaining such signals, and also because the biometric characteristics of the EEG signal may be held only for short periods of time (POLLOCK; SCHNEIDER; LYNESS, 1991).

With the emergence of new mobile devices that capture brain signals driven by the most keenly studies in the brain computer interface, the EEG as a biometric trait can now be used in some other scenarios, such as: (i) distance-based education environments, in which the continuous authentication of a student becomes increasingly necessary; (ii) with the increase in life expectancy worldwide, health monitoring systems may become popular along with home automation and smart homes, thus making the EEG-based identification very useful in this scenario; (iii) with the popularization of biometric systems for the validation of financial transactions, mobile EEG sensors become a viable alternative in the future.

Basically, an EEG-based biometric approach aims at placing a set of sensors in the person's head in order to capture the output signals for further feature extraction and analysis using signal processing techniques. The signal acquisition session is then repeated over time to make the system more discriminative and robust to errors. In a recent paper, (CAMPISI; ROCCA, 2014) presented a review on the state-of-the-art of EEG-based automatic recognition systems, as well as an overview of the neurophysiological basis that constitutes the foundations on which EEG biometric systems can be built. The authors also discussed about the major obstacles towards the deployment of EEG based biometric systems in everyday life.

One of the main problems of EEG-based person identification is the acquisition, which may be too invasive to the user. The process of putting a considerable amount of sensors up on a person's head might be a bit uncomfortable, and it also requires a previous knowledge by the person in charge of the sensors placement in order to put them in their correct positions. In light of this context, some questions may rise: "Is it really necessary to put all these sensors on a persons' head? If not, can we identify the most relevant channels for person identification and then use a smaller number of sensors in order to measure them?".

These questions motivated our work in modelling the task of channel selection as an evolutionary-based optimization problem. The idea is to propose a wrapper approach composed by an optimization technique and a pattern classifier, in which the accuracy of the latter is used to guide the evolutionary agents in the search space looking for the best solutions, i.e., the subset of channels that maximize the accuracy of the classifier in the validation set. Any optimization technique and classifier could be used.

In our work, we propose an optimum channel selection by means of a binary constrained version of the recently proposed optimization technique Flower Pollination Algorithm (BFPA) (YANG, 2012), and the Optimum-Path Forest (OPF) (PAPA et al., 2012; PAPA; FALCÃO; SUZUKI, 2009) classifier, which is a supervised pattern recognition technique that has the advantage of providing a faster training phase compared to other state-of-the-art classifiers. This characteristic of fast training is very important in the context of this paper, since a training procedure followed by a classification of a validation set need to be performed for each evolutionary agent (sometimes we may have several of them). Additionally, this version of OPF is parameterless, which is another advantage over other classifiers.

The main contributions of this paper are three-fold: (i) to evaluate a recent binary version of the Flower Pollination Algorithm (BFPA) proposed by (RODRIGUES et al., 2015) under different transfer functions[1]; (ii) to model the problem of EEG channel selection as an evolutionary-based optimization task; and (iii) to introduce the OPF classifier for EEG-based biometric person identification. The use of evolutionary optimization algorithms for the EEG channel selection is due to their elegant and simple solutions to solve optimization problems, similar to the way nature does.

This paper is organized as follows: Section 4.2 presents a brief theoretical background about EEG, and Section 4.3 discusses previous works related to this paper. Section 4.4 presents the proposed approach for person identification using a reduced number of EEG channels, and Section 4.5 presents a description of the dataset and the experimental setup. Sections 4.6 and 4.7 discuss the experiments and conclusions, respectively.

4.2 The EEG Signal

The human central nervous system consists of the encephalous (brain), which is inside the cranium, and the spinal cord contained in the spine. The nerve tissue is a complex network formed mostly by millions of nerve cells (glial cells and neurons), whose primary function is the transmission of electrical impulses that run through this intrinsic and

[1] A transfer function, in this context, aims at mapping a real-valued solution to a binary-valued one.

huge network, thus propagating information among cells (SANEI; CHAMBERS, 2007; TAU; PETERSON, 2009). These small electrical impulses emitted by the huge amount of neurons create an electric field that can be measured on the surface of the human skull, with the help of sensors or electrodes. The measurement of this complex electrical signal from our nervous system is what is known as electroencephalogram (EEG). In the literature, it is common among authors to directly refer to those brain waves as EEG.

The neural activity of the human being begins between the 17th and 23rd week of gestation. It is believed that, since this stage, and throughout the life, the signals from the brain activity represent not only the functioning of the brain, but also of the whole body. Published studies also show that even if a variation in amplitude of EEG signals during the development of a normal person exists, over the years, their functional connections remain largely unchanged (GASSER et al., 1988; TAU; PETERSON, 2009).

Figure 4.1 shows an example of a map of sensors located at a person's head. This map describes the head surface locations via relational distances, also called as International 10-10 System (JURCAK; TSUZUKI; DAN, 2007; NUWER et al., 1998). The nomenclature of the electrodes is associated to the human brain areas as follows: Frontal (F), Central (C), Temporal (T), Parietal (P) and Occipital (O) lobes. Electrodes named with two letters refer to a location between areas, for example: CP electrode is in a position between central and parietal lobes. The sub-index indicates the side of the brain hemisphere (odd numbers are located on the left side and even numbers on the right side), and the sub-index "z"indicates that the electrode is located in the main vertical axis.

4.3 Related Work

One of the first studies regarding EEG as a biometric trait was conducted by (POULOS et al., 1999), which described the EEG signal by means of an autoregressive (AR) model as the basis for a person identification method. In their work, the correct classification rates reached 91% in experiments using data obtained from 45 EEG recordings of 75 subjects, who were at rest and with the eyes closed during the test. Another study by (POULOS; RANGOUSSI; ALEXANDRIS, 1999) employed spectral features extracted from the EEG signals followed by the use of neural networks as classifiers to identify a person. The authors have achieved correct classification rates ranging from 80% to 100%, reaffirming the great potential of using EEG as a biometric feature. (ABDULLAH et al., 2010) implemented a practical system that uses four (sometimes fewer) channels and two types of EEG signals

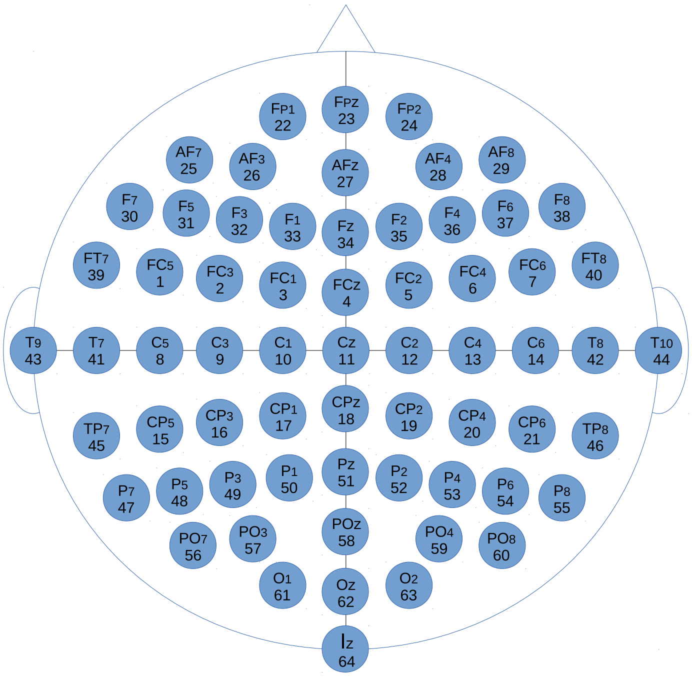


Figure 4.1: International 10-10 System standards for sensor positioning. Just for the sake of clarification, sensor T9 is placed close to the left ear, as well as sensor #23 is placed close to the nose.

(one with the eyes open and another one with the eyes closed), which were used in ten male subjects at rest in five different sessions conducted over the course of two weeks. The feature extraction was performed using AR models, and the classification was performed using a multilayer neural network. The authors observed classification rates from 70% to 97%, depending on the amount of channels and EEG type.

(PALANIAPPAN, 2004) used the gamma-band spectral power ratio as features and a Multilayer Perceptron Neural Network to recognize a person based on the EEG signal. Later on, (PALANIAPPAN; MANDIC, 2007) proposed to use 61 channels for feature extraction followed by classification using Elman Neural Network. (KOSTÍLEK; ŜTÁSTNÝ, 2012) focused on the importance of the repeatability and the influence of movements during the EEG signal acquisition session. In their work, an autoregressive model and a Mahalanobis distance-based classifier for person identification were applied to evaluate the robustness of the proposed approach. (SAFONT et al., 2012) used a set of classifiers and multiple features to perform EEG-based person identification. In their work, all possible combinations of features and classifiers have been addressed in order to improve the person recognition results.

More recently, (ROCCA et al., 2014) proposed a novel approach that fuses spectral coherence-based connectivity between different brain regions as a possibly viable biometric feature. The proposed approach was tested on a dataset of 108 subjects with eyes-closed (EC) and eyes-open (EO) resting state conditions. Their results show that using brain connectivity leads to higher distinctiveness when compared with the traditional power-spectrum measurements, reaching 100% of recognition accuracy in EC and EO conditions when integrating functional connectivity between regions in the frontal lobe.

4.4 Proposed Method

In this section, we present our proposed method for person identification based on features from EEG signals, as well as we briefly review some of the main concepts regarding the techniques employed in this paper.

4.4.1 Autoregressive Model

An Autoregressive Model can be described by a linear difference equation in the time domain as follows:

$$x(k) = P + \sum_{i=1}^{p} a(i)x(t-i) + e(t), \tag{4.1}$$

where P is a constant, p stands for the number of parameters of the model and $e(t)$ denotes a white noise input (JAIN; DESHPANDE, 2004). Notice In this work, we used the Yule-Walker method to estimate the coefficients of the AR model by employing the least square method criterion.

4.4.2 EEG Channel Selection

In order to select the best subset of channels, we evaluate a recent proposed binary version of the Flower Pollination Algorithm (RODRIGUES et al., 2015) under different transfer functions, and we also show we can obtain distinct results for each one. Firstly, we present the theoretical basis about FPA, and then its binary version.

4.4.2.1 Flower Pollination Algorithm

The Flower Pollination Algorithm proposed by (YANG, 2012) is inspired by the flow pollination process of flowering plants. The FPA is governed by four basic rules:

1. Biotic cross-pollination can be considered as a process of global pollination, and pollen-carrying pollinators move in a way that obeys Lévy flights;

2. For local pollination, abiotic pollination and self-pollination are used;
3. Pollinators such as insects can develop flower constancy, which is equivalent to a reproduction probability that is proportional to the similarity of two flowers involved; and
4. The interaction or switching of local pollination and global pollination can be controlled by a switch probability $p \in [0,1]$, slightly biased towards local pollination.

In order to model the updating formulas, the above rules have to be converted into proper updating equations. For example, in the global pollination step, flower pollen gametes are carried by pollinators such as insects, and pollen can travel over a long distance because insects can often fly and move over a much longer range. Therefore, Rules 1 and 3 can be represented mathematically as follows:

$$x_i^{(t+1)} = x_i^t + \alpha L(\lambda)(g_* - x_i^t), \tag{4.2}$$

where

$$L(\lambda) = \frac{\lambda \cdot \Gamma(\lambda) \cdot \sin(\lambda)}{\pi} \cdot \frac{1}{s^{1+\lambda}}, \quad s \gg s_0 > 0 \tag{4.3}$$

where x_i^t is the pollen i (solution vector) at iteration t, g_* is the current best solution among all solutions at the current generation, and α is a scaling factor to control the step size. $L(\lambda)$ is the Lévy-flights step size, that corresponds to the strength of the pollination, $\Gamma(\lambda)$ stands for the gamma function and s is the step size. Since insects may move over a long distance with various distance steps, a Lévy flight can be used to mimic this characteristic efficiently.

For local pollination, both Rules 2 and 3 can be represented as:

$$x_i^{(t+1)} = x_i^t + \varepsilon(x_j^t - x_k^t), \tag{4.4}$$

where x_j^t and x_k^t are pollen from different flowers j and k of the same plant species at time step t. This mimics flower constancy in a limited neighbourhood. Mathematically, if x_j^t and x_k^t come from the same species or are selected from the same population, it equivalently becomes a local random walk if ε is drawn from a uniform distribution in [0,1]. In order to mimic the local and global flower pollination, a switch probability (Rule 4) or proximity probability p is used.

4.4.2.2 Binary Flower Pollination Algorithm

In the standard FPA, the solutions are updated in the search space towards continuous-valued positions. However, in the proposed Binary Flower Pollination Algorithm the search space is modelled as an n-dimensional boolean lattice, in which the solutions are updated across the corners of a hypercube. In addition, as the problem is to select or not a given feature, a solution binary vector is employed, where 1 corresponds to a feature being selected to compose the new set, and 0 otherwise. In order to build this binary vector, (RODRIGUES et al., 2015) employed Equations 4.5 and 4.6, which can restrict the new solutions to only binary values:

$$S(x_i^j(t)) = \frac{1}{1+e^{-x_i^j(t)}}, \tag{4.5}$$

$$x_i^j(t) \quad = \quad \begin{cases} 1 & \text{if } S(x_i^j(t)) > \sigma, \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

in which $\sigma \sim U(0,1)$. Algorithm 1 presents the proposed approach that employs BFPA for EEG-channel selection using the OPF classifier as the objective function and Equation 4.5 and 4.6 as the transfer function. Note that the proposed approach can be used with any other supervised classification technique.

Lines $1-4$ initialize each pollen's position as being a binary string with random values, as well as the fitness value f_i of each individual i. The main loop in Lines $6-27$ is the core of the proposed algorithm, in which the inner loop in Lines $7-13$ is responsible for creating the new training Z_1' and evaluating sets Z_2', and then OPF is trained over Z_1' and it is used to classify Z_2'. The recognition accuracy over Z_2' is stored in acc and then compared with the fitness value f_i (accuracy) of individual i: if the later is worse than acc, the old fitness value is kept; in the opposite case, the fitness value is then updated. Lines $12-13$ update the best local position of the current pollen. Lines $14-18$ update the global optimum, and the last loop (Lines $19-27$) moves each pollen to a new binary position restricted by Equations 4.5 and 4.6 (Lines $25-27$).

4.4.3 Optimum-Path Forest Classifier

We used the Optimum-Path Forest Classifier (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012) applied to the features learned from the AR model to classify a person based

Algorithm 1: BFPA - Binary Flower Pollination Algorithm

```
input       : Training set $Z_1$ and evaluating set $Z_2$, $\alpha$, number of flowers $m$,
              dimension $d$ and iterations $T$.
output      : Global best position $\widehat{g}$.
auxiliaries: Fitness vector $f$ with size $m$ and variables $acc$, $maxfit$,
              $globalfit \leftarrow -\infty$ and $maxindex$.
for each flower $i$ ($\forall i = 1, \ldots, m$) do
    for each dimension $j$ ($\forall j = 1, \ldots, d$) do
        $x_i^j(0) \leftarrow$ Random$\{0, 1\}$;
    $f_i \leftarrow -\infty$;
for each iteration $t$ ($t = 1, \ldots, T$) do
    for each flower $i$ ($\forall i = 1, \ldots, m$) do
        Create $Z_1'$ and $Z_2'$ from $Z_1$ and $Z_2$, respectively, such that both contains only
          features such that $x_i^j(t) \neq 0$, $\forall j = 1, \ldots, d$;
        Train OPF over $Z_1'$, evaluate its over $Z_2'$ and stores the accuracy in $acc$;
        if ($acc > f_i$) then
            $f_i \leftarrow acc$;
            for each dimension $j$ ($\forall j = 1, \ldots, d$) do
                $\widehat{x}_i^j \leftarrow x_i^j(t)$;
    $[maxfit, maxindex] \leftarrow max(f)$;
    if ($maxfit > globalfit$) then
        $globalfit \leftarrow maxfit$;
        for each dimension $j$ ($\forall j = 1, \ldots, d$) do
            $\widehat{g}^j \leftarrow x_{maxindex}^j(t)$;
    for each flower $i$ ($\forall i = 1, \ldots, m$) do
        for each dimension $j$ ($\forall j = 1, \ldots, d$) do
            $rand \leftarrow$ Random$\{0, 1\}$;
            if $rand < p$ then
                $x_i^j(t) \leftarrow x_i^j(t-1) + \alpha \oplus$ Lévy$(\lambda)$; else
                    $x_i^j(t) \leftarrow x_i^j(t-1) + \varepsilon(x_i^j(t-1) - x_i^k(t-1))$;
            if ($\sigma < \frac{1}{1+e^{x_i^j(t)}}$) then
                $x_i^j(t) \leftarrow 1$; else
                    $x_i^j(t) \leftarrow 0$;
```

on the EEG signal. The OPF works by modelling the samples as graph nodes, whose arcs are defined by an adjacency relation and weighted by a distance function. Further, a role competition process between some key nodes (prototypes) is carried out in order to partition the graph into optimum-path trees (OPTs) according to a path-cost function. In fact, each OPT is rooted at one prototype, which means a sample that belongs to a given tree is more strongly connected to its root than to any other in the forest.

4.5 Methodology

In this section, we present the proposed approach for channel selection in EEG-based signal acquisition, as well as we briefly describe the employed dataset, the nature-inspired meta-heuristic algorithms, and the experimental setup.

4.5.1 Dataset

The EEG signals used in this work were obtained from the EEG Motor Movement/Imagery dataset[2] (GOLDBERGER et al., 2000). The data was collected from 109 healthy volunteers using the BCI2000 System (SCHALK et al., 2004), which makes use of 64 channels (sensors) and provides a separated EDF (European Data Format) file for each of them. The subjects performed different motor/imagery tasks: such tasks are mainly used in BCI (Brain-Computer Interface) applications and neurological rehabilitation, and consists of imagining or simulating a given action, like open and close the eyes, for example.

Each subject performed four tasks according to the position of a target that appears on the screen placed in front of the volunteers (if the target appears on the right or left side, the subject opens and closes the corresponding fist; if the target appears on the top or bottom side, the subject opens and closes both fists or both feets, respectively). In short, the four experimental tasks were:

1. To open and close left or right fist;
2. To imagine opening and closing left or right fist;
3. To open and close both fists or both feet; and
4. To imagine opening and closing both fists or both feet.

[2] http://physionet.org/pn4/eegmmidb

Each of these tasks were performed three times, thus generating 12 recordings for each subject of a two-minutes run, and the 64 channels were sampled at 160 samples per second.

The features of the twelve recordings are extracted by means of an AR model with three output configurations for each EEG-channel: 5, 10 and 20 features. Further, the average of each configuration is then been computed in order to obtain just one feature per EEG-channel (sensor). In short, for each sensor, we have extracted three different numbers of AR-based features, being the output of each sensor the average of their values. Henceforth, we have adopted the following notation for each of the dataset configurations: AR_5 for 5 autoregression coefficients extracted, and AR_{10} and AR_{20} for 10 and 20 autoregression coefficients, respectively.

4.5.2 Nature-Inspired Meta-heuristic Algorithms

In this work, we have compared our proposed method with other meta-heuristic-based optimization methods described below:

Genetic Algorithm (GA): The Genetic Algorithm was proposed by (HOLLAND, 1975), and its main concept is to emulate the biological evolution to solve optimization problems. It is composed of an initial population (or a set of unique elements) and a set of operators inspired by the nature. These operators can change the elements, and according to the evolutionary theory, only the most capable individuals are able to survive and transmit their biological heredity to the next generations.

Particle Swarm Optimization (PSO): This method is inspired on the social behaviour of a bird flocking or a fish schooling (KENNEDY; EBERHART, 2001). The fundamental idea is that each particle represents a potential solution which is updated according to its own experience and from its neighbours' knowledge. The motion of an individual particle for the optimal solution is governed through its position and velocity interactions, and also by its own previous best performance and the best performance of their neighbours.

Firefly Algorithm (FA): This method was proposed by (YANG, 2010a), being derived from the flash attractiveness of fireflies for mating partners (communication) and attracting potential preys. The brightness of a firefly at a given position is determined by the value of the objective function in that position. Each firefly is attracted by a brighter firefly through the attraction factor.

Harmony Search (HS): This method is a meta-heuristic algorithm inspired in the improvisation process of music players (GEEM, 2009). Musicians often improvise the pitches of their instruments searching for a perfect state of harmony. The main idea is to use the same process adopted by musicians to create new songs to obtain a near-optimal solution according to some fitness function. Each possible solution is modelled as a harmony, and each musical note corresponds to one decision variable.

Charged System Search (CSS): This method, based on the governing Coulomb's law (a physics law used to describe the interactions between electrically charged particles), was proposed by (KAVEH; TALATAHARI, 2010). In this method, named CSS, each Charged Particle (CP) in the system is affected by the electrical fields of the others, generating a resultant force over each CP, which is determined by using the electrostatic laws. The CP interaction movement is determined by Newtonian mechanics laws.

We have used the binary optimization version of each aforementioned method, as proposed in: Binary GA (BGA) (HOLLAND, 1975), Binary PSO (BPSO) (FIRPI; GOODMAN, 2004), Binary HS (BHS) (RAMOS et al., 2011), Binary Firefly (BFA) (FALCÓN; M.; NAYAK, 2011; PALIT et al., 2011), and Binary CSS (RODRIGUES et al., 2013b). The optimization algorithms were implemented in C language following the guidelines provided by their references. Notice the transfer function defined by Equations 4.5 and 4.6 were the very same for all techniques compared in this work.

4.5.3 Experimental Setup

We partitioned our fully labeled dataset into $\mathscr{Z} = \mathscr{Z}_1 \cup \mathscr{Z}_2 \cup \mathscr{Z}_3$ subsets, in which $\mathscr{Z}_1$, $\mathscr{Z}_2$ and $\mathscr{Z}_3$ stand for training, validation, and test sets, respectively. The training dataset contains 50% of the original dataset, followed by 30% and 20% concerning the validation and test sets, respectively. The idea is to employ $\mathscr{Z}_1$ and $\mathscr{Z}_2$ to find the subset of features that maximize the accuracy over $\mathscr{Z}_2$, with the accuracy being the fitness function.

Each agent is initialized with random binary positions and the original dataset is mapped to a new one that contains the features that were selected in this first sampling. In addition, the fitness function of each agent is set to the OPF recognition rate over $\mathscr{Z}_2$ after training in $\mathscr{Z}_1$. The final subset will be the one that maximizes the curve over the range of values, i.e., the features that maximize the accuracy over $\mathscr{Z}_2$. The accuracy over the test set $\mathscr{Z}_3$ is then assessed by using the final subset of the selected features. Notice

Technique	Parameters
BGA	$mutation = 0.1$
BPSO	$c_1 = c_2 = 2$
BFA	$\gamma = 0.8,\ \beta_0 = 1.0,\ \alpha = 0.01$
BCSS	–
BHS	HMCR$= 0.9$
BFPA	$\alpha = 1.0,\ p = 0.8$

Table 4.1: Parameters used for each meta-heuristic optimization technique. Notice the inertia weight w for PSO was linearly decreased from 0.9 to 0.4.

the fitness function employed in this paper is the accuracy measure proposed by (PAPA; FALCÃO; SUZUKI, 2009), which is capable of handling unbalanced classes. Figure 4.2 presents the methodology used to evaluate the proposed approach.

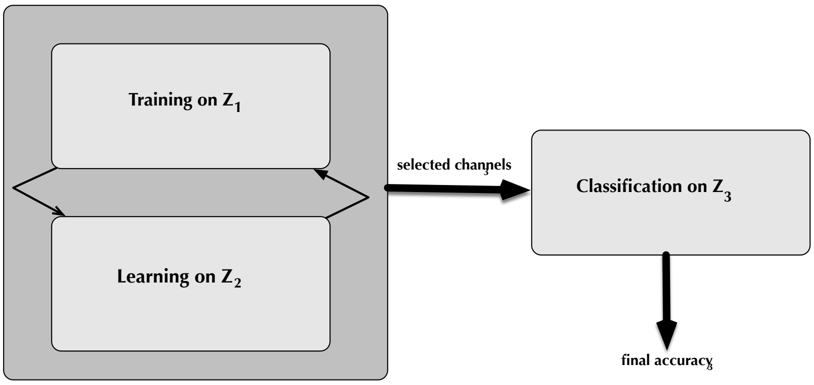


Figure 4.2: Block diagram of the proposed approach.

Table 4.1 shows the parameters used for each optimization technique employed in this work[3]. The c_1 and c_2 parameters of PSO control the pace during the particles movement, and the "Harmony Memory Considering Rate" (HMCR) of BHS stands for the amount of information that will be used from the artist's memory (songs that have been already composed) in order to compose a new harmony. In regard to BFA, α and β_0 are related to the step size of a firefly, and γ stands for the light absorption coefficient.

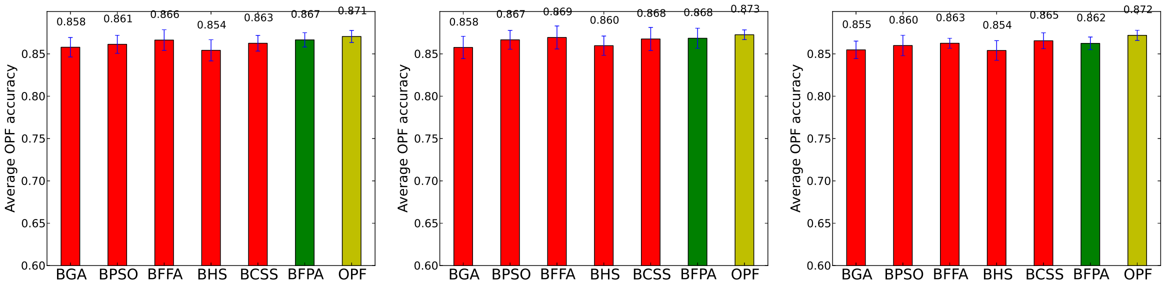
[3] We have used the same variable notation for different methods because we believe it makes it easier to understand since it is the same notation used in the respective original papers.

4.6 Experimental Results

The experimental results stand for the mean accuracy and standard deviation over 25 rounds using the methodology presented in Section 4.5.3. Since the meta-heuristic algorithms are non-deterministic, we adopt this protocol to avoid biased results. The experiments were executed in a computer with a Pentium Intel Core *i*7® 1.73Ghz processor, 6 GB of RAM and Linux Ubuntu Desktop LTS 13.04 as the operational system.

Figures 4.3 and 4.4 present the mean OPF accuracy over the three different feature sets (AR_5, AR_{10} and AR_{20}), as well as the average number of selected channels, respectively. Notice the "yellow" bar stands for the standard OPF, i.e., without channel selection. From Figure 4.3, one can observe there is not a relevant difference in terms of accuracy considering the different number of autoregression coefficients. As the coefficients are averaged at the output of each channel, such non-linear operation may have alleviated the influence of each approach. However, this operation seems to work well, since a recognition rate of around 86% is very competitive when compared to other works in the literature (Section 4.3).

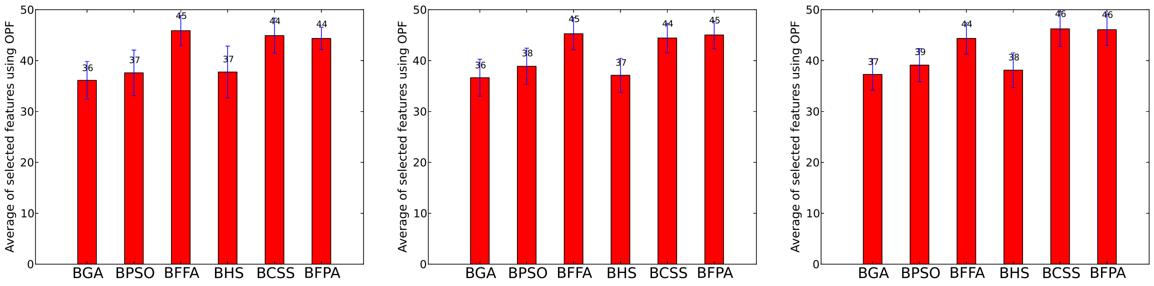
Table 4.2 presents the percentage of selected EEG-channels. From the data, it is possible to observe three important points: (i) BGA and BHS have selected the lowest number of channels for all dataset configurations; (ii) considering the accuracy results shown in Figure 4.3, we can conclude that we can achieve similar performance of that obtained using all the 64 channels by using less than a half of them; and (iii) the proposed BFPA has been very competitive in terms of binary-constrained optimization tasks when compared to the techniques addressed in this work.



(a) (b) (c)

Figure 4.3: Average OPF accuracy over (a) AR_5, (b) AR_{10} and (c) AR_{20} configurations.

Figure 4.5 depicts the mean computational load (in seconds) for all optimization techniques regarding the learning step (dark gray module in Figure 4.2.). As we did not



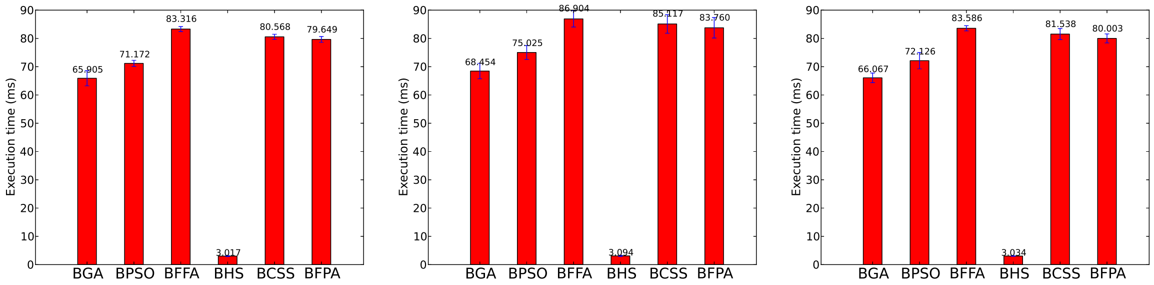
(a) (b) (c)

Figure 4.4: Average number of selected channels of all techniques over (a) AR$_5$, (b) AR$_{10}$ and (c) AR$_{20}$ configurations. These values have been truncated for sake of simplicity.

Dataset	BGA	BPSO	BFA	BHS	BCSS	BFPA
AR$_5$	36%	38%	45%	38%	44%	46%
AR$_{10}$	36%	39%	44%	36%	45%	45%
AR$_{20}$	37%	40%	44%	36%	44%	45%

Table 4.2: Percentual of selected EEG-channels.

consider the feature extraction procedure, i.e., the autoregression coefficients computation, the execution time over all dataset configurations are quite similar for each specific optimization technique. It is possible to observe BHS has been the fastest technique in all situations, since it only updates one agent per iteration. Although it may be a drawback in terms of convergence, it is still the fastest approach.



(a) (b) (c)

Figure 4.5: Mean execution times of all techniques over (a) AR$_5$, (b) AR$_{10}$ and (c) AR$_{20}$ configurations.

Finally, we performed the Wilcoxon signed-rank statistical test (WILCOXON, 1945) to verify whether there is a significant difference between BFPA and the other techniques used in this work (considering the OPF recognition rate). Table 4.3 displays a pairwise comparison against all techniques and BFPA, showing whether two techniques are

Dataset	BGA	BPSO	BFA	BHS	BCSS
AR_5	$\neq$	$=$	$=$	$\neq$	$=$
AR_{10}	$\neq$	$=$	$=$	$\neq$	$=$
AR_{20}	$\neq$	$=$	$=$	$\neq$	$=$

Table 4.3: Wilcoxon signed-rank test evaluation.

considered similar ('=') or not ('$\neq$') to each other. The only technique that has been considered similar to BFPA in all situations is BFA, followed by BPSO. An interesting point is related to the number of parameters, since BFPA requires only two, meanwhile BFA needs three parameters.

Since the nature of the proposed task in the EEG recording session has a close relation with different brain areas, like the movements of the hands and feet that mainly activates the central region of the brain (WANG; GAO; GAO, 2005; YANG et al., 2013), it is important to figure out whether the expected channels are actually included in the subset selected by the optimization techniques. Therefore, since we executed a cross-validation procedure with 25 runnings, and due to the stochastic behaviour of the meta-heuristic techniques, this means a certain feature may not be selected at a given execution, and may be at another. In order to cope with this challenge, we opted to display the frequency of occurrence concerning each sensor, as displayed in Figure 4.6. In this case, we considered BFPA with feature extraction by model AR_5.

Some interesting conclusions can be drawn if we consider the different range of frequencies modelled by distinct colours. It seems the frontal sensors are slightly more important than the back ones, since we can find more "yellow" and "blue" sensors right below the horizontal line (i.e., the one that goes from the left ear to the right one) than above that line. Another observation is that the "yellow" sensors are place everywhere, i.e., they correspond to the sensors that have been selected in between the range $[85\%, 89\%]$, which is a considerable frequency. This means BFPA tried to select sensors placed at different positions of the brain in order to capture different information.

4.6.1 Transfer Function Analisys

In order to map the possible solutions (i.e., a position in the search space) from a continuous-valued space to a binary one, a transfer function needs to be employed (RASHEDI; NEZAMABADI-POUR; SARYAZDI, 2010; MIRJALILI; HASHIM, 2011). A transfer function defines the probability of changing the position of a possible solution from 0 to 1 and

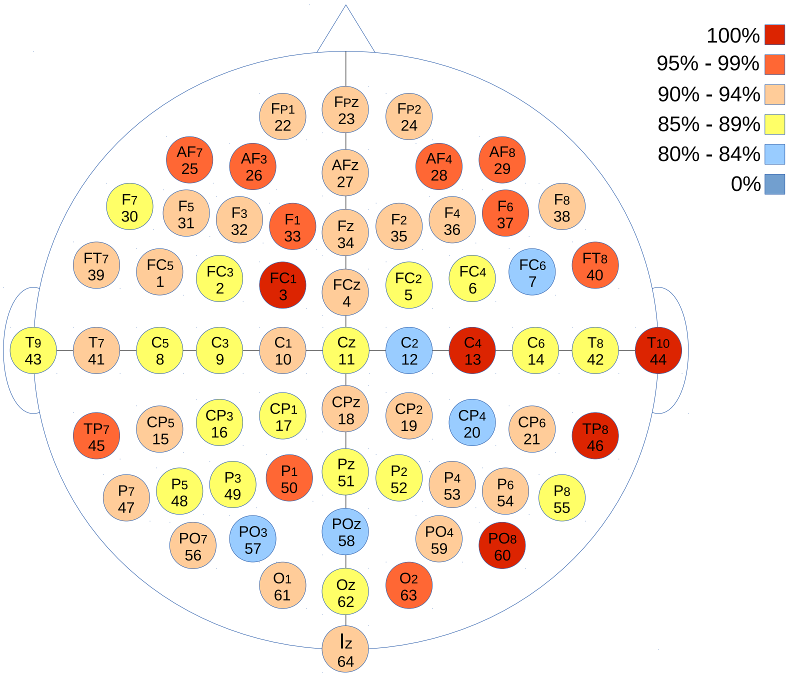


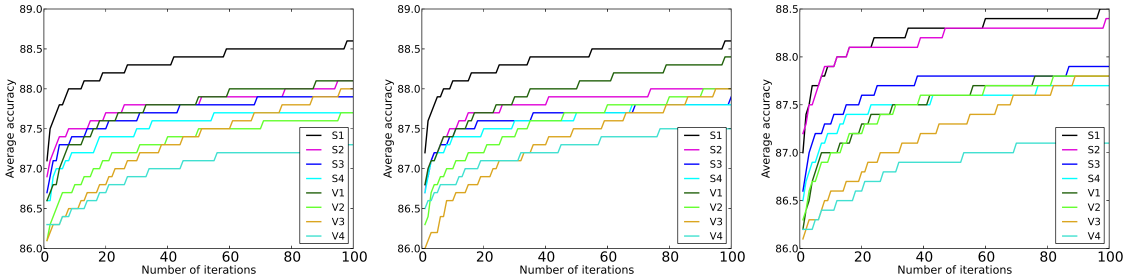
Figure 4.6: Frequency of selected sensors during the experimental evaluation using AR$_5$ and BFPA.

vice-versa forcing the agents to move onto a binary space. (MIRJALILI; LEWIS, 2013) introduced a study of two families of transfer functions on binary-based PSO. Since the binary version of FPA makes use of a transfer function either, we also investigated these two different families of transfer functions (S-shaped and V-shaped) on Binary FPA. In short, we evaluated 8 transfer functions, as follows:

- S-shaped: S1, S2, S3 and S4; and
- V-shaped: V1, V2, V3 and V4.

Notice the transfer function S2 is the same one used in the experiments conducted in the previous section (Equations 4.5 and 4.6). In this section, we just reproduced the results obtained with S2. For a more detailed explanation about the functions employed in this section, the reader can refer to the work by (RASHEDI; NEZAMABADI-POUR; SARYAZDI, 2010; MIRJALILI; HASHIM, 2011).

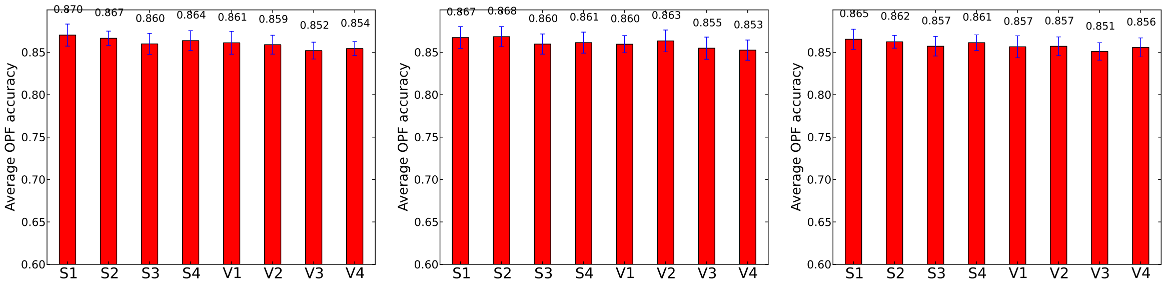
First of all, we evaluated the convergence of all tranfer functions considering the AR models used in this work. Figure 4.7 displays this experiment, in which transfer function S1 obtained the best results in all AR models, followed by S2 and V1. According to (MIRJALILI; LEWIS, 2013), the larger the velocity of a given particle, the highest it should be the probability to change its position from 1 to 0 and vice-versa, since this particle probably is far away from the best global solution. In this context, the "most abrupt" transfer functions are S1 and V1, i.e., they are more prone to switch the binary values.



(a) (b) (c)

Figure 4.7: Convergence evaluation of the transfer functions considering all AR models.

Following a similar behaviour to the ones obtained in the convergence-driven experiment, functions S1 and V1 provided very good recognition rates over the test set, as displayed in Figure 4.8. Such behaviour can be observed for all AR models. Additionally, the number of selected features can influence the recognition rates, as one can observe in Figure 4.9. Although transfer function V3 has selected less features, it obtained the lowest recognition rates (Figure 4.8), which is somehow expected. In regard to the computational load, Figure 4.10 presents the mean execution time to learn the most representative subset of features. Since transfer function S1 has selected more features, it is expected a higher computational burden when compared to the others.



(a) (b) (c)

Figure 4.8: Average OPF accuracy over (a) AR_5, (b) AR_{10} and (c) AR_{20} configurations considering different transfer functions

.

Table 4.4 displays the Wilcoxon signed-rank test considering the experiment with different transfer functions. Considering model AR_5, the most accurate techniques were S1, S2 and S4, and with respect to AR_{10} we can highlight S1, S4 and V2 as the top-3

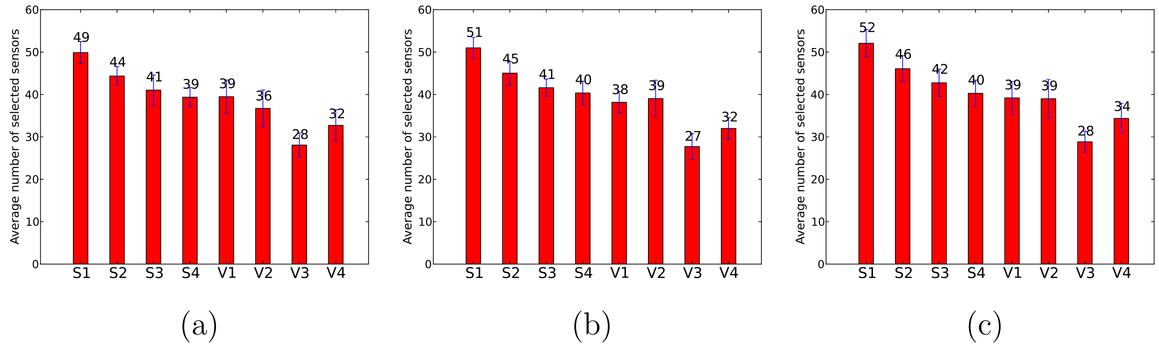


Figure 4.9: Average number of selected channels of all techniques over (a) AR_5, (b) AR_{10} and (c) AR_{20} configurations considering different transfer functions. These values have been truncated for sake of simplicity.

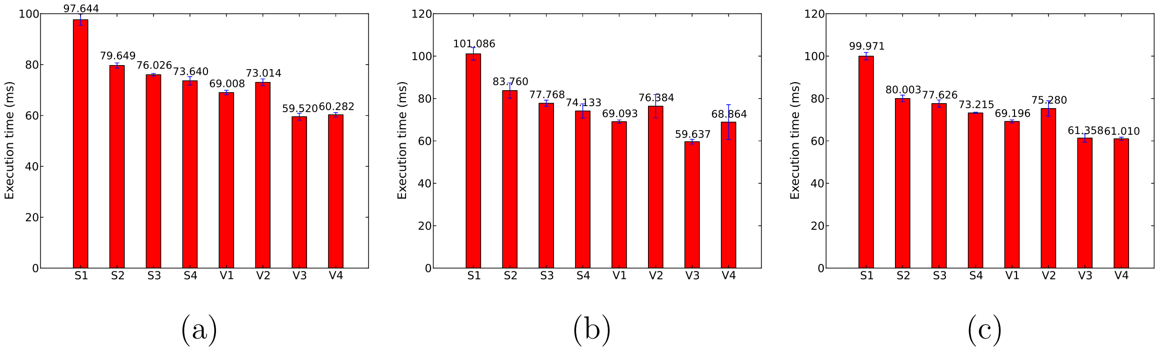


Figure 4.10: Mean execution times of all techniques considering different transfer functions over (a) AR_5, (b) AR_{10} and (c) AR_{20} configurations.

techniques. Finally, S1 and S4 obtained the best results considering the model AR_{20}.

4.6.2 Discussion

Roughly speaking, all techniques achieved similar recognition rates considering all AR models, with an advantage to BFPA and BFA, which are swarm-oriented. It is important to highlight one might obtain better recognition rates using a different feature extraction, but the main goal of this work is to evaluate BFPA in the context of sensor selection, as well as to show the importance of selecting sensors in order to make such approach less prone to errors and probably cheaper.

Using AR models with different number of coefficients seemed to does not provide different recognition rates, since the output of each AR model is given by the average of the coefficients. This could be a plausible explanation for that case. Such assumption can be applied to all meta-heuristic techniques used in this paper.

Another important point concerns with the sensors selected by BFPA. A more detailed

Table 4.4: Wilcoxon signed-rank test computed between the transfer functions.

AR$_5$	S1	S2	S3	S4	V1	V2	V3	V4
S1	—	=	≠	=	≠	≠	≠	≠
S2	=	—	≠	=	=	≠	≠	≠
S3	≠	≠	—	=	=	=	≠	=
S4	=	=	=	—	=	=	≠	≠
V1	≠	=	=	=	—	=	≠	≠
V2	≠	≠	=	=	=	—	≠	=
V3	≠	≠	≠	≠	≠	≠	—	=
V4	≠	≠	=	≠	≠	=	=	—
AR$_{10}$	S1	S2	S3	S4	V1	V2	V3	V4
S1	—	=	=	=	≠	=	≠	≠
S2	=	—	≠	=	≠	=	≠	≠
S3	=	≠	—	=	=	=	=	≠
S4	=	=	=	—	=	=	=	≠
V1	≠	≠	=	=	—	=	=	=
V2	=	=	=	=	=	—	≠	≠
V3	≠	≠	=	=	=	≠	—	=
V4	≠	≠	≠	≠	=	≠	=	—
AR$_{20}$	S1	S2	S3	S4	V1	V2	V3	V4
S1	—	=	≠	=	≠	≠	≠	≠
S2	=	—	=	=	=	≠	≠	=
S3	≠	=	—	=	=	=	=	=
S4	=	=	=	—	=	=	≠	=
V1	≠	=	=	=	—	=	=	=
V2	≠	≠	=	=	=	—	≠	=
V3	≠	≠	=	≠	=	≠	—	=
V4	≠	=	=	=	=	=	=	—

study showed the most frequent sensors are located in the front of the head, tough they are also spread along the head. That is an interesting observation, which means BFPA tried to select sensors that are not so close to each other in order to capture relevant information from all places of the head.

Finally, an additional study with different transfer functions showed we can obtain different results, being the number of selected features strongly related to the final recognition rates. It seems the more features one has, the most accurate the transfer function. However, we still need to deal with a trade-off between the number of features and the computational efficiency. Using all sensors does not give us too much different results, which supports the idea of this work, that is to emphasize one can find out the subset of sensors that can obtain reasonable results.

4.7 Conclusions and Future Work

We have addressed the problem of channel selection in EEG-based biometric person identification. The goal of this work to highlight we may not need to employ all EEG channels available in order to obtain high identification rates. Therefore, we proposed to model the problem of channel selection as a meta-heuristic-based optimization task, in which the subset of channels that maximize the recognition rate over a validation set is used as the fitness function.

For the identification (classification) task, we have used the Optimum-Path Forest classifier, which has demonstrated to be similar to the state-of-the-art supervised pattern recognition techniques, but faster for training. In regard to the meta-heuristics, we have introduced a binary-constrained optimization version of the recently proposed Flower Pollination Algorithm, which seemed to be very competitive to other state-of-the-art optimization techniques employed in this paper: Binary Genetic Algorithm, Binary Particle Swarm Optimization, Binary Firefly Algorithm, Binary Harmony Search, and Binary Charged System Search.

The experimental results showed the BFPA outperformed many of the other methods, obtaining very good person identification rates using much less channels. It is important to emphasize that reducing EEG channels while keeping high identification rates is crucial towards the effective use of EEG in biometric applications. In addition, the selected sensors seemed to cover all the person's head, mainly in the front. Moreover, the number of coefficients in the AR model does not seem to impact in the final results, although we are taking the average of the coefficients as the final feature. Finally, different transfer functions were also analyzed, which allowed slightly better results.

Although using EEG data for biometric purposes seems to be a little bit far from reality in non-controlled environments, we would like to shed light over the importance in keep going with such studies, since good recognition rates can be obtained, being such sort of biometric approaches much less prone to spoofing attacks. Probably, in the future when mobile devices can be used to easily capture EEG signals, such techniques can be widely employed for biometric purposes as well.

Our future work will involve using modified versions of FPA to perform channel selection aiming at improving the overall identification performance while selecting fewer channels.

Chapter 5

Fine-Tuning Deep Belief Networks using Cuckoo Search

In this chapter, it was proposed to use the Cuckoo Search technique to fine-tune the parameters of a Deep Belief Network, for image reconstruction. This paper has been published as a chapter in the Bio-Inspired Computation and Applications in Image Processing book (RODRIGUES; YANG; PAPA, 2016).

5.1 Introduction

Image analysis comprises with a workflow in charge of extracting relevant features from a collection of images for further classification. A number of works coped with such problem, which is usually addressed by a first overview of it, followed by learning the proper features that better describe the data. Soon after, a pattern recognition technique is employed to separate samples (feature vectors extracted from images) from different classes.

However, learning features is not so straightforward, since there is a gap in “what a person (expert) uses to describe the problem” and “what is really important to describe it”. Therefore, the use of handcrafted features can lead us to a painful and time-consuming step to design good features. In this context, deep learning techniques seem to be very useful, since they aim at learning features by means of unsupervised approaches. Convolutional Neural Networks (CNNs) (LECUN et al., 1998) and Restricted Boltzmann Machines (RBMs) (HINTON, 2012; ACKLEY; HINTON; SEJNOWSKI, 1988) are among the most used techniques to perform unsupervised learning tasks. Although their rationale is the very same one, CNNs and RBMs differ from each other in the internal working mechanism.

However, they share the same shortcomings, that are related to the fine-tuning parameters, which can easily reach thousands of them.

Recently, some works have attempted at modeling the task of choosing suitable parameters for such deep learning techniques as a meta-heuristic optimization problem. Papa et al. (PAPA et al., 2015a) introduced the Harmony Search in the context of RBM optimization, and Papa et al. (PAPA et al., 2015b) dealt with the problem of fine-tuning Discriminative Restricted Boltzmann Machines, which are a variant of naïve RBMs that can address both feature learning and pattern classification. Rosa et al. (ROSA et al., 2015) also employed Harmony Search to fine-tune CNNs, and Papa et al. (PAPA; SCHEIRER; COX, 2015) addressed Harmony Search and a number of its variants to optimize Deep Belief Networks (DBNs), which is essentially composed of stacked RBMs. Last but not least, Fedorovici et al. (FEDOROVICI et al., 2012) optimized CNNs in the context of Optical Character Recognition using Gravitational Search Algorithm.

However, as the reader can observe, the area of meta-heuristic-based deep learning optimization is still in its embryonic stage. In this work, we evaluated a swarm-based meta-heuristic optimization technique called Cuckoo Search (CS) (YANG; S., 2010) to this task, which is based on the predator mechanism of cuckoos, which make use of nests from other species to raise their own brood. The CS is employed to optimize DBNs and RBMs in the context of binary image reconstruction. We present a discussion about the viability in using such approach against with Harmony Search and Particle Swarm Optimization. The experimental section comprised two public datasets, as well as a statistical evaluation by means of Wilcoxon signed-rank test. We hope this work can guide readers and enthusiasts towards a better comprehension about using meta-heuristics for deep learning techniques fine-tuning.

The remainder of this chapter is organized as follows. Section 5.2 introduces the theory background about RBMs, DBNs and CS. Sections 5.3 and 5.4 present the methodology and the experimental results, respectively. Finally, Section 5.5 states conclusions and future works.

5.2 Theoretical Background

In this section, we briefly review some of the main important concepts regarding RBMs and DBNs, as well as the Cuckoo Search technique.

5.2.1 Deep Belief Networks

5.2.1.1 Restricted Boltzmann Machines

Restricted Boltzmann Machines are energy-based stochastic neural networks composed of two layers of neurons (visible and hidden), in which the learning phase is conducted by means of an unsupervised fashion. Figure 5.1 depicts the architecture of a Restricted Boltzmann Machine, which comprises a visible layer $\mathbf{v}$ with m units and a hidden layer $\mathbf{h}$ with n units. The real-valued $m \times n$ matrix $\mathbf{W}$ models the weights between visible and hidden neurons, where w_{ij} stands for the weight between the visible unit v_i and the hidden unit h_j.

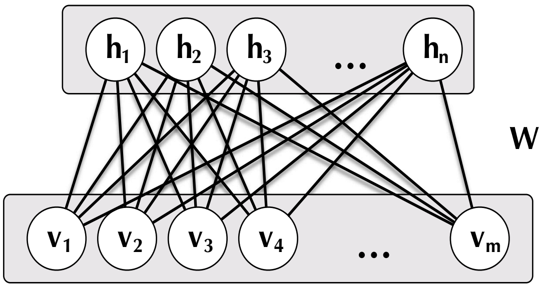


Figure 5.1: The RBM architecture.

Let us assume $\mathbf{v}$ and $\mathbf{h}$ as the binary visible and hidden units, respectively. In other words, $\mathbf{v} \in \{0,1\}^m$ and $\mathbf{h} \in \{0,1\}^n$. The energy function of a Bernoulli Restricted Boltzmann Machine is given by:

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i=1}^{m} a_i v_i - \sum_{j=1}^{n} b_j h_j - \sum_{i=1}^{m}\sum_{j=1}^{n} v_i h_j w_{ij}, \tag{5.1}$$

where $\mathbf{a}$ and $\mathbf{b}$ stand for the biases of visible and hidden units, respectively. The probability of a configuration $(\mathbf{v},\mathbf{h})$ is computed as follows:

$$P(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum\limits_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}, \tag{5.2}$$

where the denominator of above equation is a normalization factor that stands for all possible configurations involving the visible and hidden units. In short, the BRBM learning algorithm aims at estimating $\mathbf{W}$, $\mathbf{a}$ and $\mathbf{b}$. The next section describes in more details this procedure.

5.2.2 Learning Algorithm

The parameters of an BRBM can be optimized by performing stochastic gradient ascent on the log-likelihood of training patterns. Given a training sample (visible unit), its probability is computed over all possible hidden vectors, as follows:

$$P(\mathbf{v}) = \frac{\sum\limits_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}{\sum\limits_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}. \tag{5.3}$$

In order to update the weights and biases, it is necessary to compute the following derivatives:

$$\frac{\partial \log P(\mathbf{v})}{\partial w_{ij}} = E[h_j v_i]^{data} - E[h_j v_i]^{model}, \tag{5.4}$$

$$\frac{\partial \log P(\mathbf{v})}{\partial a_i} = v_i - E[v_i]^{model}, \tag{5.5}$$

$$\frac{\partial \log P(\mathbf{v})}{\partial b_j} = E[h_j]^{data} - E[h_j]^{model}, \tag{5.6}$$

where $E[\cdot]$ stands for the expectation operation, and $E[\cdot]^{data}$ and $E[\cdot]^{model}$ correspond to the data-driven and the reconstructed-data-driven probabilities, respectively.

In practical terms, we can compute $E[h_j v_i]^{data}$ considering $\mathbf{h}$ and $\mathbf{v}$ as follows:

$$E[\mathbf{hv}]^{data} = P(\mathbf{h}|\mathbf{v})\mathbf{v}^T, \tag{5.7}$$

where $P(\mathbf{h}|\mathbf{v})$ stands for the probability of obtaining $\mathbf{h}$ given the visible vector (training data) $\mathbf{v}$:

$$P(h_j = 1|\mathbf{v}) = \sigma\left(\sum_{i=1}^{m} w_{ij} v_i + b_j\right), \tag{5.8}$$

where $\sigma(\cdot)$ stands for the logistic sigmoid function. Therefore, it is straightforward to compute $E[\mathbf{hv}]^{data}$: given a training data $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ stands for a training set, we just need to set $\mathbf{v} \leftarrow \mathbf{x}$ and then employ Equation 5.8 to obtain $P(\mathbf{h}|\mathbf{v})$. Further, we use Equation 5.7 to finally obtain $E[\mathbf{hv}]^{data}$.

However, we need to deal with the problem of estimating $E[\mathbf{hv}]^{model}$, which is the model learned by the system. One possible strategy is to perform alternating Gibbs sampling starting at any random state of the visible units until a certain convergence criterion, such as k steps, for instance. The Gibbs sampling consists of updating hidden units using Equation 5.8 followed by updating the visible units using $P(\mathbf{v}|\mathbf{h})$, given by:

$$P(v_i = 1|\mathbf{h}) = \sigma\left(\sum_{j=1}^{n} w_{ij}h_j + a_i\right), \tag{5.9}$$

and then updating the hidden units once again using Equation 5.8. In short, it is possible to obtain an estimative of $E[\mathbf{hv}]^{model}$ by initializing the visible unit with random values and then performing Gibbs sampling. Notice a single iteration is defined by computing $P(\mathbf{h}|\mathbf{v})$, followed by computing $P(\mathbf{v}|\mathbf{h})$ and then computing $P(\mathbf{h}|\mathbf{v})$ once again.

For the sake of explanation, let us assume $P(\mathbf{v}|\tilde{\mathbf{h}})$ is used to denote the visible unit $\mathbf{v}$ is going to be reconstructed using $\tilde{\mathbf{h}}$, which was obtained through $P(\mathbf{h}|\mathbf{v})$. The same takes place with $P(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})$, that reconstructs $\tilde{\mathbf{h}}$ using $\tilde{\mathbf{v}}$, which was obtained through $P(\mathbf{v}|\tilde{\mathbf{h}})$. However, to perform Gibbs sampling until convergence is time-consuming, being also quite hard to establish suitable values for k[1]. Fortunately, Hinton (HINTON, 2002) introduced a faster methodology to compute $E[\mathbf{hv}]^{model}$ based on contrastive divergence. Basically, the idea is to initialize the visible units with a training sample, to compute the states of the hidden units using Equation 5.8, and then to compute the states of the visible unit (reconstruction step) using Equation 5.9. Roughly speaking, this is equivalent to perform Gibbs sampling using $k = 1$.

Based on the above assumption, we can now compute $E[\mathbf{hv}]^{model}$ as follows:

$$E[\mathbf{hv}]^{model} = P(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})\tilde{\mathbf{v}}^T. \tag{5.10}$$

Therefore, the equation below leads to a simple learning rule for updating the weight matrix $\mathbf{W}$, as follows:

$$\begin{aligned}\mathbf{W}^{t+1} &= \mathbf{W}^t + \eta(E[\mathbf{hv}]^{data} - E[\mathbf{hv}]^{model}) \\ &= \mathbf{W}^t + \eta(P(\mathbf{h}|\mathbf{v})\mathbf{v}^T - P(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})\tilde{\mathbf{v}}^T),\end{aligned} \tag{5.11}$$

where $\mathbf{W}^t$ stands for the weight matrix at time step t, and η corresponds to the learning

[1] Actually, it is expected a good reconstruction of the input sample when $k \to +\infty$.

rate. Additionally, we have the following formulae to update the biases of the visible and hidden units:

$$\begin{aligned}\mathbf{a}^{t+1} &= \mathbf{a}^t + \eta(\mathbf{v} - E[\mathbf{v}]^{model}) \\ &= \mathbf{a}^t + \eta(\mathbf{v} - \tilde{\mathbf{v}}),\end{aligned} \tag{5.12}$$

and

$$\begin{aligned}\mathbf{b}^{t+1} &= \mathbf{b}^t + \eta(E[\mathbf{h}]^{data} - E[\mathbf{h}]^{model}) \\ &= \mathbf{b}^t + \eta(P(\mathbf{h}|\mathbf{v}) - P(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})),\end{aligned} \tag{5.13}$$

where $\mathbf{a}^t$ and $\mathbf{b}^t$ stand for the visible and hidden units biases at time step t, respectively. In short, Equations 5.11, 5.12 and 5.13 are the vanilla formulation for updating the RBM parameters.

Later on, Hinton (HINTON, 2012) introduced a weight decay parameter λ, which penalizes weights with large magnitude[2], as well as a momentum parameter α to control possible oscillations during the learning process. Therefore, we can rewrite Equations 5.11, 5.12 and 5.13 as follows[3]:

$$\mathbf{W}^{t+1} = \mathbf{W}^t + \underbrace{\eta(P(\mathbf{h}|\mathbf{v})\mathbf{v}^T - P(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})\tilde{\mathbf{v}}^T) - \lambda\mathbf{W}^t + \alpha\Delta\mathbf{W}^{t-1}}_{=\Delta\mathbf{W}^t}, \tag{5.14}$$

$$\mathbf{a}^{t+1} = \mathbf{a}^t + \underbrace{\eta(\mathbf{v} - \tilde{\mathbf{v}}) + \alpha\Delta\mathbf{a}^{t-1}}_{=\Delta\mathbf{a}^t} \tag{5.15}$$

and

$$\mathbf{b}^{t+1} = \mathbf{b}^t + \underbrace{\eta(P(\mathbf{h}|\mathbf{v}) - P(\tilde{\mathbf{h}}|\tilde{\mathbf{v}})) + \alpha\Delta\mathbf{b}^{t-1}}_{=\Delta\mathbf{b}^t}. \tag{5.16}$$

[2] The weights may increase during the convergence process.

[3] Notice when $\lambda = 0$ and $\alpha = 0$, we have the naïve gradient ascent.

5.2.2.1 Deep Belief Nets

Truly speaking, DBNs are composed of a set of stacked RBMs, being each of them trained using the learning algorithm presented in Section 5.2.2 in a greedy fashion, which means an RBM at a certain layer does not consider others during its learning procedure. Figure 5.2 depicts such architecture, being each RBM at a certain layer represented as illustrated in Figure 5.1. In this case, we have a DBN composed of L layers, being $\mathbf{W}^i$ the weight matrix of RBM at layer i. Additionally, we can observe the hidden units at layer i become the input units to the layer $i+1$. Although we did not illustrate the bias units for the visible (input) and hidden layers in Figure 5.2, we also have such units for each layer.

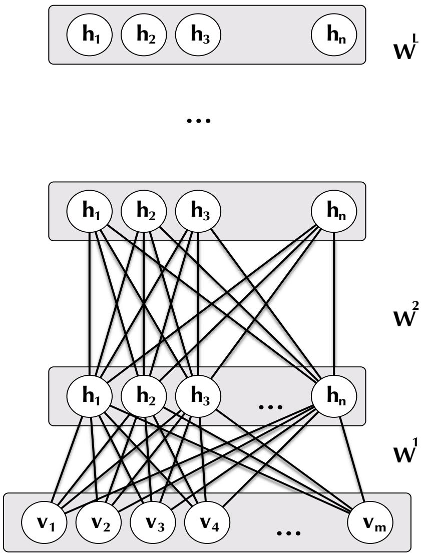


Figure 5.2: The DBN architecture.

The approach proposed by Hinton et al. (HINTON; OSINDERO; TEH, 2006) for the training step of DBNs also considers a fine-tuning as a final step after the training of each RBM. Such procedure can be performed by means of a Backpropagation or Gradient descent algorithm, for instance, in order to adjust the matrices $\mathbf{W}^i$, $i = 1,2,\ldots,L$. The optimization algorithm aims at minimizing some error measure considering the output of an additional layer placed at the top of the DBN after its former greedy training. Such layer is often composed of softmax or logistic units, or even some supervised pattern recognition technique.

5.2.3 Cuckoo Search

The parasite behavior of some cuckoo species is extremely intriguing. These birds can lay down their eggs in host nests, and mimic external characteristics of host eggs such as color and spots. In case of this strategy is unsuccessful, the host can throw the cuckoo's egg away, or simply abandon its nest, making a new one in another place. Based on this context, Yang and Deb (YANG; S., 2010) presented a novel evolutionary optimization algorithm named as Cuckoo Search, and they have summarized CS using three rules, as follows:

1. Each cuckoo choose a nest randomly to lays eggs.
2. The number of available host nests is fixed, and nests with high quality of eggs will carry over to the next generations.
3. In case of a host bird discovered the cuckoo egg, it can throw the egg away or abandon the nest, and build a completely new nest. There is a fixed number of host nests, and the probability that an egg laid by a cuckoo is discovered by the host bird is $p_a \in [0,1]$.

CS performs a balanced combination of a local random walk and the global explorative random walk, controlled by a switching parameter $p_a \in [0,1]$. The local random walk can be written as

$$x_i^j(t) = x_i^j(t-1) + \alpha \cdot s \oplus H(p_a - \varepsilon) \oplus (x_{k'}^j(t-1) - x_{k''}^j(t-1)), \tag{5.17}$$

where $x_{k'}^j$ and $x_{k''}^j$ are two different solutions selected by random permutation, and and x_i^j stands for the j^{th} egg at nest i , $i = 1,2,\ldots,M$, and $j = 1,2,\ldots,d$. $H(\cdot)$ is a Heaviside function, ε is a random number drawn from a uniform distribution, and s is the step size.

The global random walk is carried out using Lévy flights as follows:

$$x_i^j(t) = x_i^j(t-1) + \alpha \cdot L(s,\lambda), \tag{5.18}$$

where

$$L(s,\lambda) = \frac{\lambda \cdot \Gamma(\lambda) \cdot \sin(\lambda)}{\pi} \cdot \frac{1}{s^{1+\lambda}}, \quad s \gg s_0 > 0. \tag{5.19}$$

The Lévy flights employ a random step length which is drawn from a Lévy distribution. Therefore, the CS algorithm is more efficient in exploring the search space as its step length is much longer in the long run. The parameter $\alpha > 0$ is the step size scaling factor, which should be related to the scales of the problem of interest. Yang and Deb (YANG; S., 2010) claim that $\alpha = O(S/10)$ can be used in most cases, where S denotes the scale of the problem of interest, while $\alpha = O(S/100)$ can be more effective and avoid flying too far.

5.3 Methodology

In this section, we present the methodology used to evaluate the performance of CS regarding the task of DBN model selection and its application for binary image reconstruction. Details about the dataset, experimental setup and the compared techniques are provided next.

5.3.1 Datasets

- MNIST dataset: it is composed of images of handwritten digits. The original version contains a training set with 60,000 images from digits '0' to '9', as well as a test set with 10,000 images. Due to the high computational burden for RBM model selection, we decided to employ the original test set together with a reduced version of the training set. In addition, we resized all images to a resolution of 14×14.

- Semeion Handwritten Digit dataset: this dataset contains 1,593 binary images of manuscript digits with resolution of 16×16 from around 80 persons. We employed the whole dataset in the experimental section.

5.3.2 Nature-Inspired Metaheuristic Algorithms

In this work, we have also considered other evolutionary optimization techniques for comparison purposes. A brief detail about each of them is given below.

Harmony Search (HS): is a meta-heuristic algorithm inspired in the improvisation process of music players (GEEM, 2009). Musicians often improvise the pitches of their instruments searching for a perfect state of harmony. The main idea is to use the same process adopted by musicians to create new songs to obtain a near-optimal solution according to some fitness function. Each possible solution is modeled as a harmony, and each musical note corresponds to one decision variable.

Improved Harmony Search (IHS): The Improved Harmony Search (MAHDAVI; FE-

SANGHARY; DAMANGIR, 2007) differs from traditional HS by updating the PAR and ρ values dynamically, thus enhancing accuracy and convergence rate.

Particle Swarm Optimization (PSO): is inspired on the social behavior of bird flocking or fish schooling (KENNEDY; EBERHART, 2001). The fundamental idea is that each particle represents a potential solution that is updated according to its own experience and from its neighbors' knowledge. The motion of an individual particle for the optimal solution is governed through its position and velocity interactions, and also by its own previous best performance and the best performance of their neighbors.

5.3.3 Experimental Setup

In this work, we compared the proposed CS-based DBN model selection against with HS, IHS and PSO. The robustness of parameter fine-tuning was evaluated in three DBN models: one layer (1L) [4], two layers (2L) and three layers (3L). Additionally, 5 agents over 50 iterations for convergence considering all techniques with 20 runnings with a cross-validation procedure in order to provide a statistical analysis by means of Wilcoxon signed-rank test (WILCOXON, 1945). Table 5.1 presents the parameter configuration for each meta-heuristic optimization technique. Finally, we have set each DBN parameter according to the following ranges: $n \in [5,100]$, $\eta \in [0.1,0.9]$, $\lambda \in [0.1,0.9]$ and $\alpha \in [0.0,0.001]$. We employed $T = 10$ as the number of epochs for the DBN learning weights procedure with mini-batches of size 20 and Contrastive Divergence (HINTON, 2002) as the training method. Notice the fitness function used in this work is the reconstruction error (i.e., Mean Squared Error - MSE) over the training set.

Technique	**Parameters**
HS	$HMCR = 0.7, PAR = 0.7, \eta = 1.0$
IHS	$HMCR = 0.7, PAR_{min} = 0.1, PAR_{max} = 0.7, \eta_{min} = 1.0, \eta_{max} = 0.10$
PSO	$c_1 = 1.7, c_2 = 1.7, w = 0.7$
CS	$\alpha = 0.1, p_a = 0.25$

Table 5.1: Parameters used for each technique.

5.4 Experimental results

In this section, we present the experimental evaluation considering CS, HS, IHS and PSO over the MNIST and SEMEION datasets. Table 5.2 presents the MSE for each

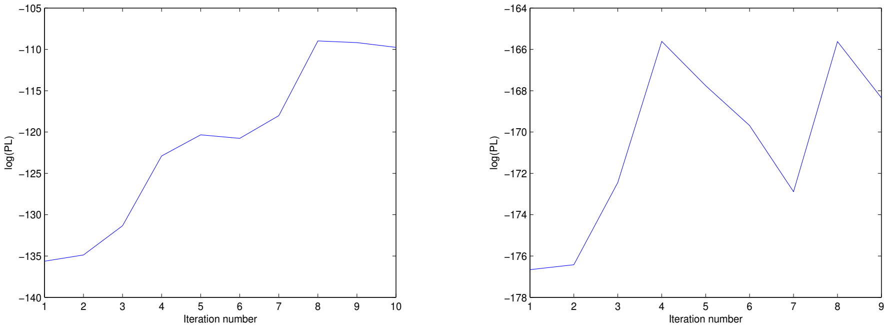
[4] Notice the 1L approach stands for the standard RBM

optimization technique over the test set considering DBNs with one, two and three layers for the MNIST dataset. Notice we used only 2% of the original training set for training purposes. The most accurate techniques are in bold.

Table 5.2: Average MSE over the test set considering MNIST dataset.

	1L	2L	3L
HS	0.1059±0.0002	0.1059±0.0002	0.1059±0.0002
IHS	0.0903±0.0048	**0.0885±0.0039**	**0.0877±0.0003**
PSO	0.1057±0.0002	0.1060±0.0005	0.1058±0.0003
CS	0.1066±0.0028	0.1076±0.0007	0.1064±0.0037

Although the lowest mean squared error was obtained by IHS using three layers (IHS-3L), a statistical evaluation by means of the Wilcoxon signed-rank test (WILCOXON, 1945) with $\alpha = 0.05$ pointed no difference between IHS-2L and IHS-3L. However, all remaining techniques, including CS, obtained close results as well. Figure 5.3a displays the Logarithm of the Pseudo-likelihood considering the 10 iterations of CS-1L learning step over MNIST dataset. Although we employed 10 iterations for learning only, we can observe the Pseudo-likelihood values are increasing, which means the reconstruction error is decreasing at each iteration. Usually, the literature employs thousands of iterations, but for the sake of computational purposes, we did not go so far. The main idea of this work is to show we can obtain reasonable reconstructed images using Cuckoo Search, and therefore we are not interested into outperforming the best results out there, since they use specific configurations that concern the mini-batch size and number of epochs.



(a) (b)

Figure 5.3: Logarithm of the Pseudo-likelihood values considering (a) MNIST and (b) SEMEION datasets using CS.

In regard to Semeion dataset, 30% was used for training, and the remaining 70% employed for testing purposes. Table 5.3 presents the same procedure applied to the MNIST dataset, where the most accurate technique is in bold. Once again, IHS obtained the lowest MSE using three layers. Figure 5.3b displays the Logarithm of the Pseudo-likelihood considering the 10 iterations of CS-1L learning step over Semeion dataset. In this case, if one take a look a the convergence curve, a more oscillating behavior can be observed, since this dataset poses a bigger challenge than MNIST, which can be reflected in the MSE as well. Actually, although IHS obtained the best result, all techniques achieved very close results, thus showing all of them are suitable to the task addressed in this work.

Table 5.3: Average MSE over the test set considering Semeion dataset.

	1L	2L	3L
HS	0.2128±0.0002	0.2128±0.0002	0.2129±0.0002
IHS	0.2127±0.0003	0.2116±0.0010	**0.2103±0.0009**
PSO	0.2128±0.0002	0.2128±0.0003	0.2128±0.0002
CS	0.2135±0.0005	0.2134±0.0002	0.2132±0.0008

5.5 Conclusions

In this work, we evaluated the Cuckoo Search for the optimization of Deep Belief Networks concerning the task of binary image reconstruction. We considered two public datasets and a DBN with one, two and three layers. In order to evaluate the robustness of CS, we compare it against HS, IHS and PSO. The experimental results using the Wilcoxon signed-rank test highlighted IHS with three layers as the most accurate technique, although all techniques obtained very close results.

Actually, it is expected better results using three layers, since one can obtain more discriminative information to be used in the reconstruction process. Based on our experience, IHS has been the most accurate technique when compared to a number of meta-heuristic techniques out there. In regard to future works, we aim at using modified versions of the Cuckoo Search as well as to perform a deeper study about the influence of its parameters for the optimization of Deep Belief Networks.

Chapter 6

Pruning Optimum-Path Forest Classifiers Using Multi-Objective Optimization

In this chapter, we present meta-heuristic multi-objective algorithms applied to the Optimum-Path Forest pruning. This work was published in the 30th Conference on Graphics, Patterns and Images *SIBGRAPI* (RODRIGUES; SOUZA; PAPA, 2017).

6.1 Introduction

Optimization techniques have been widely used in several research areas, since many problems usually refer to the task of finding the minimum or maximum of a given function. Some challenges such as the allocation of resources, product delivery for logistic companies, and cutting-and-packing problems with direct application in industries are among the most widely pursued tasks.

In regard to optimization techniques, a considerable attention has been given to nature-inspired meta-heuristics, i.e., approaches that aim at solving several problems using concepts based on physical process, social dynamics and/or the behavior of living beings (HOLLAND, 1992; STORN; PRICE, 1997; KENNEDY; EBERHART, 2001; RASHEDI; Nezamabadi-pour; SARYAZDI, 2009; YANG, 2010b; YANG; S., 2010; KAVEH; TALATAHARI, 2010). Since such techniques are quiet elegant to solve optimization problems, they have been applied for solving multi-objective optimization problems, where the idea of a unique global optimal solution is replaced by a non-dominated solution set, the so-called Pareto-optimal set (FONSECA; FLEMING, 1993; HORN; NAFPLIOTIS; GOLDBERG, 1994; SRINIVAS; DEB, 1994; ZITZLER; THIELE, 1999; KNOWLES; CORNE, 1999; DEB et al., 2002).

Meta-heuristic multi-objective optimization techniques have become popular to solve many optimization problems in the field of engineering (SIVASUBRAMANI; SWARUP, 2011; OMKAR et al., 2011; AKBARI et al., 2012; LAU et al., 2013; KHALILI-DAMGHANI; ABTAHI; TAVANA, 2013a; ZHENG; SONG; CHEN, 2013; MARICHELVAM; PRABAHARAN; YANG, 2014). However, such techniques have a wider range of applications, mainly in the context of machine learning-oriented problems, which are usually composed of several multi-objective tasks (JIN; SENDHOFF, 2008). It is very common to face problems in which we need to find out the best set of parameters (e.g., a neural network architecture) that lead to both high recognition rates and low computational burden.

Parameter-dependent machine learning techniques are often preferable to cope with real-world problems, since they can be adjusted to fit better to a given application. Support Vector Machines (SVMs) (CORTES; VAPNIK, 1995), Neural Networks (NNs) (HAYKIN, 2007) and Optimum-Path Forest (OPF) (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012; PAPA; FERNANDES; FALCÃO, 2017) are some examples of parameter-dependent techniques, just to name a few. Although OPF comprises a collection of classifiers, being some of them parameterless, new problems may require some of them to be parameterized. Papa et al. (PAPA et al., 2010) proposed to design compact though representative training sets by means of learning the most important samples during training, thus discarding the remaining ones (WILSON; MARTINEZ, 2000; JANKOWSKI; GROCHOWSKI, 2004; PęKALSKA; DUIN; PACLíK, 2006). Such process is ruled by a parameter that controls the desired loss in accuracy with respect to the final pruned training set when compared to the original one. Later on, Nakamura et al. (NAKAMURA et al., 2011) modeled the problem of finding the OPF pruning parameter automatically as a mono-objective optimization problem. In fact, they combined both information of training set size and accuracy in a single equation (fitness function) for further optimization.

Actually, since a good recognition accuracy does require a considerable training set size (for most applications where training is complex), these two criteria fit perfectly into a multi-objective optimization problem, since they are conflicting with each other. Therefore, the main contribution of this paper is apply meta-heuristic multi-objective algorithms to the Optimum-Path Forest pruning algorithm in order to obtain compact and representative training sets without the need for the desired loss and the maximum number of iteration parameters. The experiments showed the robustness of the proposed approach in a number of datasets.

The remainder of this paper is organized as follows. Section 6.2 presents the theoretical background about multi-objective optimization, while Section 6.3 presents the OPF classifier and its pruning strategy. Section 6.4 discusses the experiments, and Section 6.5 states conclusions and future works.

6.2 Multi-Objective Optimization

The multi-objective optimization problem aims at finding the global minimum $\mathbf{x}^* \in \mathscr{S}$ that minimizes a set of M functions represented by $\mathbf{f}$, i.e.:

$$\mathbf{x}^* = \arg \min_{\forall \mathbf{x} \in \mathscr{S}} (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})), \tag{6.1}$$

subject to:

$$g_i(\mathbf{x}) = 0 \quad \forall i = 1, 2, \ldots, p, \tag{6.2}$$

$$h_i(\mathbf{x}) \geq 0 \quad \forall i = 1, 2, \ldots, q, \tag{6.3}$$

where p and q represent the number of equality $g(\cdot)$ and inequality constraints $h(\cdot)$, and $\mathscr{S} \in \mathbb{R}^N$ stands for the search space.

In a multi-objective problem, there is no single solution that is optimal with respect to all objectives when considering conflicting objectives. Thus, the solution to a multi-objective optimization problem is no longer a scalar value, but a vector in the form of a "trade-off" known as *Pareto-optimal set*.

Firstly, we define the *Pareto Dominance*, where a solution vector $\mathbf{x}^a$ is said to dominate another solution vector $\mathbf{x}^b$ (i.e., $\mathbf{x}^a \prec \mathbf{x}^b$) if $f(x_i^a) \neq f(x_i^b), \forall i = \{1, 2, \ldots, N\}$, and $\exists i \in \{1, 2, \ldots, N\}$ such that $f(x_i^a) < f(x_i^b)$. In regard to the Pareto Dominance, a solution vector $\mathbf{x}^a$ is considered Pareto-optimal if, for every $\mathbf{x}^b$, $f_j(\mathbf{x}^a) \neq f_j(\mathbf{x}^b)$, $j = 1, 2, \ldots, M$, and if there exists at least one $j \in \{1, 2, \ldots, M\}$ such that $f_j(\mathbf{x}^a) < f_j(\mathbf{x}^b)$. Therefore, the Pareto-optimal set $\mathscr{P}^*$ considering a multi-objective optimization problem $\mathbf{f}(\mathbf{x})$ with respect to all Pareto-optimal solutions is thus defined as follows:

$$\mathscr{P}^* = \{\mathbf{x} \in \mathscr{S} \mid \mathbf{f}(\mathbf{x}) \prec \mathbf{f}(\mathbf{x}'), \ \forall \mathbf{x}' \in \mathscr{S}\}. \tag{6.4}$$

The Pareto-optimal front PF^* with respect to a multi-objective optimization problem $\mathbf{f}(\vec{x})$ and the Pareto-optimal set $\mathscr{P}^*$ is defined as follows:

$$PF^* = \{\mathbf{f}(\mathbf{x}) \mid \mathbf{x} \in \mathscr{P}^*\}. \tag{6.5}$$

One way to solve multi-objective optimization problems is to combine all objectives into a single-objective problem. Hence, the idea of scalarized multi-objective optimization is to convert a problem of minimizing the vector $\mathbf{x}$ into a scalar optimization problem (MIETTINEN, 1998). The weighted-sum method is one of the most used approach, in which several objective functions are combined into a single one through a weight vector. Thus, a problem with multiple objective functions is reduced to a single optimization problem subject to the original constraints, and the choice of the value of each weight is performed according to a preference assigned to each objective function:

$$\mathbf{x}^* = \arg \min_{\forall \mathbf{x} \in \mathscr{S}} \left(\sum_{k=1}^{M} w_k f_k(\mathbf{x}) \right), \tag{6.6}$$

with $\sum_{i=1}^{M} w_i = 1$. A single point of the Pareto front will be produced by a given a weight vector. A sufficiently large number of weight vectors generate a good approximation to the true Pareto front. If the weights are positive for all objectives, the solutions to the problem are Pareto optimal (MIETTINEN, 1998; DEB, 2001). The weights are calculated as follows:

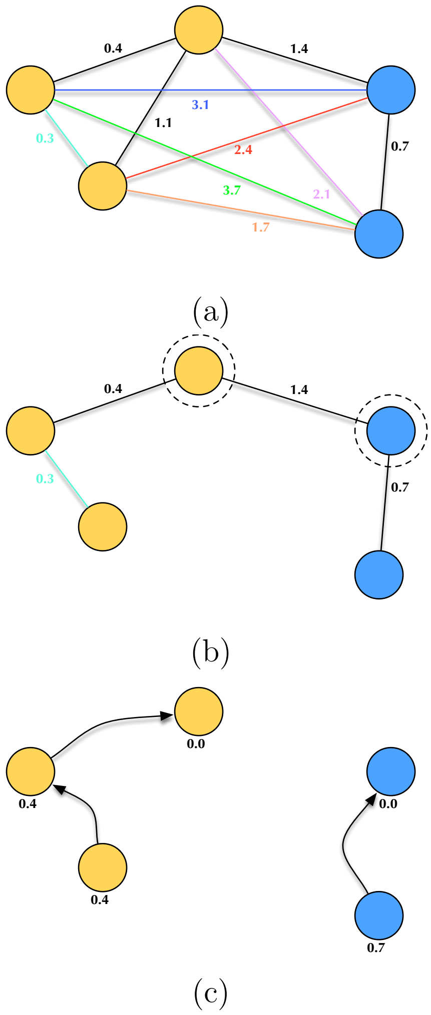
$$w_i = \frac{u_i}{\sum_{i=1}^{M} u_i}, \tag{6.7}$$

where $u_i \sim \mathscr{U}(0,1)$.

6.3 Optimum-Path Forest

Let $\mathscr{D} = \mathscr{D}^{tr} \cup \mathscr{D}^{ts}$ be a λ-labeled dataset such that $\mathscr{D}^{tr}$ and $\mathscr{D}^{ts}$ stand for the training and testing sets, respectively. Additionally, let $\mathbf{s} \in \mathscr{D}$ be an n-dimensional sample that encodes features extracted from a certain data, and $d(\mathbf{s}, \mathbf{v})$ be a function that computes the distance between two samples $\mathbf{s}$ e $\mathbf{v}$, $\mathbf{v} \in \mathscr{D}$.

Let $\mathscr{G}^{tr} = (\mathscr{D}^{tr}, \mathscr{A})$ be a graph derived from the training set, such that each node $\mathbf{v} \in \mathscr{D}^{tr}$ is connected to every other node in $\mathscr{D}^{tr} \backslash \{\mathbf{v}\}$, i.e. $\mathscr{A}$ defines an adjacency relation known as *complete graph* (Figure 6.1a illustrates such training graph), in which the arcs are weighted by function $d(\cdot,\cdot)$. We can also define a path π_s as a sequence of adjacent and distinct nodes in $\mathscr{G}^{tr}$ with terminus at node $\mathbf{s} \in \mathscr{D}^{tr}$. Notice a *trivial path* is denoted by $\langle s \rangle$, i.e. a single-node path.



(a)

(b)

(c)

Figure 6.1: Illustration of the OPF working mechanism: (a) a two-class (orange and blue labels) training graph with weighted arcs, (b) a MST with prototypes highlighted, and (c) optimum-path forest generated during the training phase with costs over the nodes (notice the prototypes have zero cost).

Let $f(\pi_s)$ be a path-cost function that essentially assigns a real and positive value to a given path π_s, and $\mathscr{S}$ be a set of prototype nodes. Roughly speaking, OPF aims at solving the following optimization problem:

$$\min f(\pi_s),\ \forall\ \mathrm{s} \in \mathscr{D}^{tr}. \tag{6.8}$$

The good point is that one does not need to deal with mathematical constraints, and the only rule to solve Equation 6.8 concerns that all paths must be rooted at $\mathscr{S}$. Therefore, we must choose two principles now: how to compute $\mathscr{S}$ (prototype estimation heuristic) and $f(\pi)$ (path-cost function).

Since prototypes play a major role, Papa et al. (PAPA; FALCÃO; SUZUKI, 2009) proposed to position them at the regions with the highest probabilities of misclassification, i.e. at the boundaries among samples from different classes. In fact, we are looking for the nearest samples from different classes, which can be computed by means of a Minimum Spanning Tree (MST) over $\mathscr{G}^{tr}$. The MST has interesting properties, which ensure OPF can be errorless during training when all arc-weights are different to each other (ALLèNE et al., 2010). Figure 6.1b depicts a MST with prototypes highlighted.

Finally, with respect to the path-cost function, OPF requires f to be a smooth one (aO; STOLFI; LOTUFO, 2004). Previous experience in image segmentation led the authors to use a chain code-invariant path-cost function, that basically computes the maximum arc-weight along a path, being denoted as f_{max} and given by:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0 & \text{if } \mathbf{s} \in \mathscr{S} \\ +\infty & \text{otherwise,} \end{cases} \\ f_{max}(\pi_s \cdot (\mathbf{s}, \mathbf{t})) &= \max\{f_{max}(\pi_s), d(\mathbf{s}, \mathbf{t})\}, \end{aligned} \tag{6.9}$$

where $\pi_s \cdot (\mathbf{s}, \mathbf{t})$ stands for the concatenation between path π_s and arc $(\mathbf{s}, \mathbf{t}) \in \mathscr{A}$. In short, by computing Equation 6.9 for every sample $\mathbf{s} \in \mathscr{D}^{tr}$, we obtain a collection of optimum-path trees (OPTs) rooted at $\mathscr{S}$, which then originate an optimum-path forest. A sample that belongs to a given OPT means it is more strongly connected to it than to any other in $\mathscr{G}^{tr}$. Roughly speaking, the OPF training step aims at solving Equation 6.9 in order to build the optimum-path forest, as displayed in Figure 6.1c. A gentle implementation of the aforementioned procedure is given by *Algorithm* 2.

Line 1 calls ***SelectPrototypes*** function, which computes the minimum spanning tree over the input graph, selects prototypes as the connected elements with different classes (Figure 6.1b), and finally it outputs the prototype set $\mathscr{S}$. Lines $2-4$ and $5-7$ initialize the prototypes and remaining samples, respectively, where $C_{\mathbf{s}}$ stands for the cost of sample s, and $P_{\mathbf{s}}$ denotes its predecessor in the optimum-path forest. Line 8 creates a priority queue based on the input graph and the cost of each sample (for such purpose, LibOPF implements a binary heap).

The main loop in Lines $9-17$ is in charge of the OPF competition process, in which Line 10 removes a sample s from the priority queue whose cost is minimum, and Line 11 inserts s in the ordered list K (such list will be used to speed up the classification phase). The inner loop in Lines $12-17$ evaluates all neighbors of $\mathbf{s}$ in order to conquer them, and line 13 computes f_{max} as described by Equation 6.9. When sample $\mathbf{v}$ is conquered by $\mathbf{s}$

Algorithm 2: OPF with Complete Graph - Training Algorithm

```
   Input: A $\lambda$-labeled training graph $\mathscr{G}^{tr} = (\mathscr{D}^{tr}, \mathscr{A})$ and the distance function $d$.
   Output: An optimum-path forest $P$, label map $L$, cost map $C$, and a list of nodes
           ordered by their costs (ascending order) $K$.
1  $\mathscr{S} \leftarrow$ SelectPrototypes($\mathscr{G}^{tr}$);
2  for $s \in \mathscr{S}$ do
3      $C_{\mathbf{s}} \leftarrow 0$;
4      $P_{\mathbf{s}} \leftarrow$NIL;
5  for $s \in \mathscr{D}^{tr} \backslash \mathscr{S}$ do
6      $C_{\mathbf{s}} \leftarrow \infty$;
7      $P_{\mathbf{s}} \leftarrow$NIL;
8  $Q \leftarrow$ BuildPriorityQueue($\mathscr{D}^{tr}$,$C$); $K \leftarrow \emptyset$;
9  while $Q \neq \emptyset$ do
10     Remove from $Q$ a sample $\mathbf{s}$ whose $C_{\mathbf{s}}$ is minimum;
11     $K \leftarrow K \cup \{s\}$;
12     for $v \in \mathscr{D}^{tr} \backslash s$ do
13         $tmp \leftarrow \max\{C_{\mathbf{s}}, d(\mathbf{s}, \mathbf{v})\}$;
14         if $(tmp < C_v)$ then
15             $C_{\mathbf{v}} \leftarrow tmp$;
16             $P_{\mathbf{v}} \leftarrow \mathbf{s}$;
17             $L_{\mathbf{v}} \leftarrow \lambda(\mathbf{s})$;
18 return $[P, L, C, K]$;
```

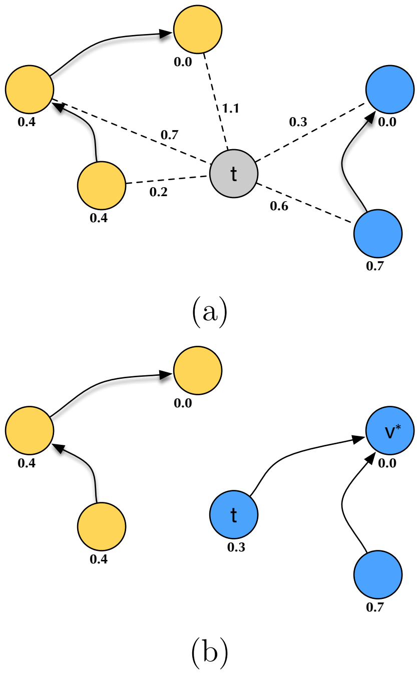
(Lines $15-17$), the cost (Line 15), predecessor (Line 16) and label map (Line 17) of $\mathbf{v}$ are updated.

The next step concerns the testing phase, where each sample $\mathbf{t} \in \mathscr{D}^{ts}$ is classified individually as follows: $\mathbf{t}$ is connected to all training nodes from the optimum-path forest learned in the training phase (Figure 6.2a), and it is evaluated the node $\mathbf{v}^* \in \mathscr{D}^{tr}$ that conquers $\mathbf{t}$, i.e. the one that satisfies the following equation:

$$C_{\mathbf{t}} = \arg\min_{\mathbf{v} \in \mathscr{D}^{tr}} \max\{C_{\mathbf{v}}, d(\mathbf{v}, \mathbf{t})\}. \tag{6.10}$$

The classification step simply assigns $L(\mathbf{t}) = \lambda(\mathbf{v}^*)$, as depicted in Figure 6.2b. Roughly speaking, the testing step aims at finding the training node $\mathbf{v}$ that minimizes $C_{\mathbf{t}}$.

The example displayed in Figure 6.2 shows an interesting situation: although $\mathbf{t}$ is closest to a sample from "yellow" class, it has been labeled to the another class, which emphasizes OPF is not a distance-based classifier, but instead it uses the "power of connectivity" among samples. The OPF with complete graph degenerates to a nearest neighbor classifier only when all training samples are prototypes. Actually, such situation is con-



(a)

(b)

Figure 6.2: Illustration of the OPF classification mechanism: (a) sample t is connected to all training nodes, and (b) t is conquered by v* and it receives the "blue" label.

siderably difficult to face, thus indicating a high degree of overlapping among samples, which means the features used for that specific problem may not be adequate enough to describe it. Algorithm 3 implements the OPF classification procedure. Notice this algorithm uses the ordered list of notes K, where $k_i \in K$ stands for a node in $\mathscr{D}^{tr}$, to speed up the classification step, as proposed by Papa et al. (PAPA; FERNANDES; FALCÃO, 2017).

Algorithm 3: OPF Classification Algorithm

```
Input: Classifier $[P_1, C_1, L_1, K]$, test set $\mathscr{D}^{ts}$, and the distance function $d$ .
Output: Label $L_2$ and predecessor $P_2$ maps defined for $\mathscr{D}^{ts}$, and accuracy value
        $Acc$.
Auxiliary: Cost variables $tmp$ and $mincost$.
for each $t \in \mathscr{D}^{ts}$ do
    $i \leftarrow 1$, $mincost \leftarrow max\{C_1(k_i), d(k_i, t)\}$;
    $L_2(t) \leftarrow L_1(k_i)$, $P_2(t) \leftarrow k_i$;
    while $i < |K|$ and $mincost > C_1(k_{i+1})$ do
        Compute $tmp \leftarrow max\{C_1(k_{i+1}, d(k_{i+1}, t\}$;
        if $tmp < mincost$ then
            $mincost \leftarrow tmp$;
            $L_2(t) \leftarrow L_1(k_{i+1})$, $P_2(t) \leftarrow k_{i+1}$;
        $i \leftarrow i + 1$;
Compute accuracy $Acc$ according to (PAPA; FALCÃO; SUZUKI, 2009);
return $[L_2, P_2, Acc]$;
```

6.3.1 Learning with Pruning of Irrelevant Patterns

Large datasets usually present redundancy, so at least in theory it should be possible to estimate a reduced training set with the most relevant patterns for classification. The use of a training set $\mathscr{D}^{tr}$ and an evaluating set $\mathscr{D}^{ev}$ has allowed OPF to learn relevant samples for $\mathscr{D}^{tr}$ from the classification erros in $\mathscr{D}^{ev}$, by swapping misclassified samples of $\mathscr{D}^{ev}$ and non-prototype samples of $\mathscr{D}^{tr}$ during a few iterations (PAPA; FALCÃO; SUZUKI, 2009). In this learning strategy, $\mathscr{D}^{tr}$ remains the same size and the classifier instance with the highest accuracy is selected to be tested on the unseen set $\mathscr{D}^{ts}$. Algorithm 4 implements this learning procedure.

```
Algorithm 4: OPF Learning Algorithm
Input: A $\lambda$-labeled training and evaluating sets $\mathscr{D}^{tr}$ and $\mathscr{D}^{ev}$, respectively,
       number $T$ of iterations, and distance function $d$.
Output: Optimum-path forest $P_1$, cost map $C_1$, label map $L_1$, ordered set $K$ and
        $MaxAcc$.
Auxiliary: Arrays $FP$ and $FN$ of sizes $c$ for false positives and false negatives, set
           $S$ of prototypes, and list $LM$ of misclassified samples.
Set MaxAcc $\leftarrow -1$;
for each iteration $I = 1, 2, \ldots, T$ do
    $LM \leftarrow 0$ and compute the set $\mathscr{S} \subset \mathscr{D}^{tr}$ of prototypes;
    $[P_1, C_1, L_1, K] \leftarrow$ Algorithm 2($\mathscr{D}^{tr}, \mathscr{S}, d$);
    for each class $i$ do
        $FP(i) \leftarrow 0$ and $FN(i) \leftarrow 0$.
    $[L_2, P_2, Acc] \leftarrow$Algorithm 3($[P_1, C_1, L_1, K], \mathscr{D}^{ev}, d$);
    if $Acc > MaxAcc$ then
        $[P_1^*, C_1^*, L_1^*, K^*] \leftarrow [P_1, C_1, L_1, K]$;
        $MaxAcc \leftarrow Acc$;
    while $LM \neq 0$ do
        $LM \leftarrow LM \backslash \{t\}$;
        Replace $t$ by a non-prototype sample randomly selected from $\mathscr{D}^{tr}$;
return $[P_1^*, C_1^*, L_1^*, Z^*]$ and $MaxAcc$;
```

The efficacy of Algorithm 4 increases with the size of $\mathscr{D}^{tr}$, because more non-prototype samples can be swapped by misclassified samples of $\mathscr{D}^{ev}$. However, for sake of efficiency, we need to choose a reasonable maximum size for $\mathscr{D}^{tr}$. After learning the best training samples for $\mathscr{D}^{tr}$, we may also mark paths in P_1 used to classify samples in $\mathscr{D}^{ev}$ and define their nodes as *relevant samples* in a set $\mathscr{R}$. The "irrelevant" training samples in $\mathscr{D}^{tr} \backslash \mathscr{R}$ can then be moved to $\mathscr{D}^{ev}$. Algorithm 5 applies this idea repetitively, while the loss in accuracy on $\mathscr{D}^{ev}$ with respect to the highest accuracy obtained by Algorithm 4 (using the initial training set size) is less or equal to a maximum value *MLoss* specified by the user or there are no more irrelevant samples in $\mathscr{D}^{tr}$.

Algorithm 5: OPF Pruning Algorithm

```
   **Input:** Training and evaluating sets, $\mathscr{D}^{tr}$ and $\mathscr{D}^{ev}$, labeled by $\lambda$, distance function
          $d$, maximum loss $MLoss$ in accuracy on $\mathscr{D}^{ev}$, and number $T$ of iterations.
   **Output:** OPF classifier $[P_1, C_1, L_1, Z]$ with reduced training set.
   **Auxiliary:** Set $\mathscr{R}$ of relevant samples, and variables $Acc$ and $tmp$.
1  $[P_1, C_1, L_1, K] \leftarrow$ Algorithm 4$(\mathscr{D}^{tr}, \mathscr{D}^{ev}, T, d)$;
2  $[L_2, P_2, Acc] \leftarrow$ Algorithm 3$([P_1, C_1, L_1, K], \mathscr{D}^{ev}, d)$;
3  $tmp \leftarrow Acc$ and $\mathscr{R} \leftarrow \emptyset$;
4  while $(Acc - tmp) \leq MLoss$, $\mathscr{R} \neq \mathscr{D}^{tr}$ do
5      $\mathscr{R} \leftarrow \emptyset$;
6      for each sample $t \in \mathscr{D}^{ev}$ do
7          $s \leftarrow P_2(t) \in \mathscr{D}^{tr}$;
8          while $s \neq NIL$ do
9              $\mathscr{R} \leftarrow \mathscr{R} \cup s$;
10             $s \leftarrow P_1(s)$;
11     Move samples from $\mathscr{D}^{tr} \backslash \{\mathscr{R}\}$ to $\mathscr{D}^{ev}$;
12     $[P_1, C_1, L_1, K] \leftarrow$ Algorithm 4$(\mathscr{D}^{tr}, \mathscr{D}^{ev}, T, d)$;
13     $[L_2, P_2, Acc] \leftarrow$Algorithm 3$([P_1, C_1, L_1, K]), \mathscr{D}^{ev}, d)$;
14 return $[P_1, C_1, L_1, K]$;
```

In Algorithm 5, Lines $1-3$ compute learning and classification using the highest accuracy classifier obtained for an initial training set size. Its accuracy is returned in *Acc* and used as reference value in order to stop the pruning process when the loss in accuracy is greater than a user-specified *MLoss* value, or when all training samples are considered relevant. The main loop in Lines $4-13$ essentially marks the relevant samples in $\mathscr{D}^{tr}$ by following the optimum paths used for classification (Lines $6-10$) backwards, moves irrelevant samples to $\mathscr{D}^{ev}$, and repeats learning and classification from a reduced training set until it reaches the above stopping criterion.

6.4 Experimental Results

In this section, we present the experimental results with respect to the proposed approach to optimize OPF pruning. In order to minimize both the OPF classification error and the training set size, we considered the following optimization techniques: Multi-objective Black Hole Algorithm (MOBHA) (JEET; DHIR, 2016), Multi-objective Cuckoo Search (MOCS) (RANI et al., 2012), Multi-objective Firefly Algorithm (MOFFA) (YANG, 2013) and Multi-objective Particle Swarm Optimization (MOPSO) (BHUVANESWARI, 2015).

Additionally, we employed ten benchmark datasets, being eight from UCI repository[1] and two private: NTL-Comercial and NTL-Industrial. Notice we set the number of possible solutions, $m = 10$, and the number of iterations $T = 10$. We used 50%, 30% and 20% for the training, evaluating and testing set percentages, respectively. In regard to the source-code, we used the LibOPT (Papa et al., 2017).

The proposed approach aims at modeling the problem of automatically tuning the $MLoss$ parameter and the number of iterations T by means of a multi-objective problem, which means we are going to use the OPF classification error over the evaluating set and the training set size as the fitness functions to be minimized. Mathematically speaking, the aforementioned problem can be formulated as follows:

$$\mathbf{x}^* = \arg \min_{\forall \mathbf{x} \in \mathscr{S}} \left(w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) \right), \tag{6.11}$$

where $f_1(\mathbf{x})$ stands for the the OPF classification error over the evaluating set, and $f_2(\mathbf{x})$ denotes the training set size. As such, we opted to approach the problem as a scalarized multi-objective problem, as previously discussed in Section 6.2.

Table 6.1 displays the number of samples, number of features, and number of classes for each dataset. Notice we decided to evaluate the robustness of the proposed approach under different scenarios.

Table 6.1: Description of the benchmarking datasets.

Dataset	No. samples	No. features	No. classes
German Numer	1,000	24	2
Ionosphere	351	34	2
MPEG-7	1,400	180	70
Pendigits	7,494	16	10
Satimage	4,435	36	6
Sonar	208	60	2
Splice	1,000	60	2
SVM-Guide 2	391	20	3
NTL-Comercial	4952	8	2
NTL-Industrial	3182	8	2

Table 6.2 presents the parameter configuration for each meta-heuristic optimization technique. The parameters were set based on empirical studies conducted over a number of experiments.

In order to evaluate the techniques compared in this work, we used a measure F that

[1] http://archive.ics.uci.edu/ml/

Table 6.2: Parameters used for each technique.

Technique	Parameters
PSO	$c_1 = 1.7, c_2 = 1.7, w = 0.7$
CS	$\alpha = 0.1, p_a = 0.25$
FFA	$\alpha = 0.2, \beta = 1, \gamma = 1$
BHA	–

considers the results of both fitness functions as follows:

$$F = \frac{\text{accuracy over the test set}}{\text{training set size}}. \tag{6.12}$$

Therefore, since one aims at using smaller training sets and obtaining higher accuracy rates, the greater the value of F, the better the technique is.

Table 6.3 presents the results concerning the trade-off between the OPF classification accuracy and the training set size. However, Table 6.4 displays the F values for each optimization technique considered in this work, where the values in bold stand for the best results. Notice OPF denotes the standard classifier, i.e., without pruning samples. It is worth noting that MOBHA obtained the best results in German Numer, MPEG-7 and Pendigits datasets. MOCS achieved the best F values in SVM-Guide 2, NTL-Comercial and NTL-Industrial datasets. Concerning Splice and Sonar datasets, the highest values belong to MOFFA, while the better results in Ionosphere and Satimage datasets were achieved by MOPSO.

Clearly, one can observe the multi-objective techniques obtained better results than standard OPF in all datasets. Also, different optimization techniques obtained the best results in distinct datasets, although they have performed similarly in all situations. Another interesting point to observe concerns the stability of the methodology employed in this work. The results presented in Table 6.4 stand for a single run over the datasets. Although Equation 6.11 initializes w_1 and w_2 with random values, we have observed that different runs, i.e., experiments with other values for these variables, did not influence the final results. Probably, the results would be different if we have used more fitness functions, as well as the decision variables ($MLoss$ and T) seem to stabilize both the accuracy and training set size after some time.

Table 6.5 displays the classification time over the testing set (miliseconds) regarding the proposed approach to find $MLoss$ and T. One can observe that MOBHA obtained the lowest classification time in German Numer, MPEG-7, Pendigits, SVM-Guide 2 and

Table 6.3: Trade-off between the OPF classification accuracy and training set size.

	German Numer	Ionosphere	MPEG-7	Pendigits	Satimage	Sonar	Splice	SVM-Guide 2	NTL-Comercial	NTL-Industrial
MOBHA	61.03%/167	85.53%/51	88.65%/251	99.25%/928	91.95%/666	86.63%/36	70.70%/137	76.12%/64	63.19%/601	70.37%/414
MOCS	60.00%/175	85.53%/52	89.61%/263	99.44%/960	92.15%/665	84.69%/31	70.91%/134	78.61%/64	66.75%/615	75.11%/436
MOFFA	63.97%/174	88.58%/50	89.25%/254	99.44%/962	92.28%/672	88.35%/31	70.25%/132	79.95%/68	64.32%/637	70.43%/414
MOPSO	62.86%/178	89.47%/48	89.49%/256	99.36%/946	92.64%/630	87.04%/31	72.11%/147	74.55%/65	62.76%/614	70.08%/418
OPF	59.64%/500	81.58%/174	91.85%/700	99.67%/3744	93.15%/2214	85.98%/103	65.86%/499	74.16%/194	62.97%/2475	57.91%/1590

Table 6.4: F values considered the dataset and techniques employed in this work.

	German Numer	Ionosphere	MPEG-7	Pendigits	Satimage	Sonar	Splice	SVM-Guide 2	NTL-Comercial	NTL-Industrial
MOBHA	**0.0037**	0.016771	**0.003532**	**0.00107**	0.001381	0.024064	0.005161	0.011894	0.001051	0.0017
MOCS	0.003429	0.016448	0.003407	0.001036	0.001386	0.027319	0.005292	**0.012283**	**0.001085**	**0.001723**
MOFFA	0.003676	0.017716	0.003514	0.001034	0.001373	**0.0285**	**0.005322**	0.011757	0.001010	0.001701
MOPSO	0.003532	**0.01864**	0.003496	0.001050	**0.001471**	0.028077	0.004905	0.011469	0.001022	0.001677
OPF	0.001193	0.004689	0.001312	0.000266	0.000421	0.008348	0.001320	0.003823	0.000254	0.000364

NTL-Comercial datasets, thus obtaining a gain of 322%, 272%, 419%, 322% and 454%, respectively, in terms of computational load. With respect to Ionosphere, Satimage, Sonar and NTL-Industrial datasets, MOPSO obtained the lowest classification time with gains of 390%, 363%, 313% and 413%, respectively. MOCS obtained the best classification time in Splice dataset achieving a gain of 368%. Therefore, we can highlight the gain in efficiency concerning the proposed multi-objective OPF pruning algorithm.

Table 6.5: OPF classification time [ms] over the test set.

Datasets	OPF	MOBHA	MOCS	MOFFA	MOPSO
German Numer	0.012205	**0.003800**	0.004236	0.004243	0.004359
Ionosphere	0.001462	0.000454	0.000447	0.000488	**0.000375**
MPEG-7	0.080667	**0.029715**	0.033082	0.030930	0.032747
Pendigits	0.481865	**0.115130**	0.120072	0.122654	0.122084
Satimage	0.251095	0.073188	0.075933	0.077176	**0.069209**
Sonar	0.000653	0.000448	0.000228	0.000246	**0.000209**
Splice	0.018213	0.005239	**0.004956**	0.005084	0.005645
SVM-Guide 2	0.001660	**0.000516**	0.000828	0.001063	0.000808
NTL-Comercial	0.185389	**0.040857**	0.041952	0.044368	0.043448
NTL-Industrial	0.071641	0.017673	0.018219	0.018176	**0.017353**

6.5 Conclusions

The very basic idea of OPF pruning is to learn the most representatives samples in order to create a compact training set. However, OPF pruning is parameter-dependent (***MLoss*** and ***T***), and finding the proper values for such parameters can be a hard task.

In this paper, we proposed to use well-known meta-heuristic algorithms in order to find near-optimal values concerning the aforementioned parameters. The experiments were conducted over 10 datasets in order to show the robustness of the proposed approach. We have observed that all meta-heuristic algorithms obtained $\boldsymbol{F}$ values considerably better than standard OPF, indicating a good compactness of the training sets, being MOBHA and MOCS the ones that obtained the best results in the majority datasets. Also, we can highlight the improvements in the computational load.

In regard to future works, we intend to work on a many-objective optimization model by adding more fitness functions, such as pruning less representative OPF prototypes. We believe that keeping more prototypes, we can also obtain more accurate results.

Chapter 7

A Multi-Objective Artificial Butterfly Optimization Approach for Class-Oriented Feature Selection

In this chapter, the binary version of Artificial Butterfly Optimization technique was used for the task of feature selection concerning multi- and many-objective optimization. The chapter in question was submitted to *Applied Soft Computing*.

7.1 Introduction

Nowadays, artificial intelligence is present in several fields of knowledge (SODHRO et al., 2017, 2019b, 2019; SODHRO; PIRBHULAL; ALBUQUERQUE, 2019; SODHRO et al., 2019a). Consequently, high-dimensional datasets have become pretty much useful. However, most of the features are usually irrelevant and/or redundant, thus contributing to degrading the classification performance, as well as to poor data visualization and understanding, overfitting and high computational burden. Feature selection attempts to remove such irrelevant features without much loss of information looking for a better understanding of the data, thus obtaining a compact subset of relevant features. The main idea is to obtain similar or even better classification accuracies than using the complete set of features (HARVEY; TODD, 2015).

Since the size of the search space grows exponentially according to the number of features, exhaustive search techniques may not be viable for online applications (XUE; ZHANG; BROWNE, 2013). In the last few years, considerable attention has been given to nature-inspired meta-heuristic algorithms, i.e., approaches that aim at solving several problems using concepts based on physical process, social dynamics, and/or the behavior of living beings (HOLLAND, 1992; STORN; PRICE, 1997; KENNEDY; EBERHART, 2001; RASHEDI; Nezamabadi-pour; SARYAZDI, 2009; YANG, 2010b; YANG; S., 2010; KAVEH; TALATAHARI, 2010; ABEDINPOURSHOTORBAN et al., 2016). Such techniques are quietly elegant and applicable to a wide range of optimization problems that can not be solved exactly within a reasonable amount of time. Using simple rules and principles, these algorithms tend to explore the entire search space for a suitable solution, and as soon as they find promising regions, they perform a more refined search to find the optimal solution.

Feature selection can be naturally modeled as an optimization task, with particular attention to the ones called "wrapper approaches". Such methods use the classification performance as the fitness function mostly, thus guiding the process into selecting the subset of samples that maximize some measure that considers the classifier's output. In the past years, several nature-inspired meta-heuristic algorithms have been often applied to solve one objective function only (RODRIGUES et al., 2013a; WANG et al., 2014; DIAO; SHEN, 2015; AHMAD; BAKAR; YAAKUB, 2015; RODRIGUES et al., 2016; WANG; TAN; NIU, 2019). Concerning single-objective feature selection, Papa et al. (PAPA et al., 2011) proposed a feature selection approach combining the Gravitational Search Algorithm (GSA) with the Optimum-Path Forest (OPF) classifier. Experiments on vowel recognition, image classification, and fraud detection in power distribution datasets demonstrated the proposed approach outperformed techniques such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and PSO. Hegazy et al. (HEGAZY; MAKHLOUF; EL-TAWEL, 2018) tested an improved version of the Salp Swarm Algorithm (SSA) (MIRJALILI et al., 2017) to the task of feature selection. By incorporating an inertia weight parameter to enhance the performance, the proposed technique (ISSA) is validated on twenty-three benchmark datasets. The experimental results demonstrated superior classification accuracy and feature reduction compared to SSA, PSO, Genetic Algorithm (GA), Ant Lion Optimizer (ALO) and Grey Wolf Optimizer (GWO). Papa et al. (PAPA et al., 2018) proposed a binary version of the BrainStorm Optimization (BSO) in the context of feature selection. The proposed approach is then compared with other fourteen meta-heuristic optimization approaches achieving suitable results on twenty-five benchmark datasets. Posteriorly, Pourpanah et al. (POURPANAH et al., 2019) proposed a hybrid version of the BSO combined with a fuzzy neural network architecture (CARPENTER et al., 1992) to handle feature selection problem. The authors found the results were promising compared to other feature selection methods such as PSO, GA, Genetic Programming (GP), and Ant Colony Optimization (ACO). Taradeh et al. (TARADEH et al., 2019) combined the Gravitational Search Algorithm (GSA) with evolutionary crossover and mutation operators and produced a wrapper-based feature selection approach. The proposed approach was validated on eighteen benchmark datasets demonstrating higher performance than GA, PSO, and GWO.

Usually, we have problems with two or three objective functions to be optimized at the same time, and they are addressed as Multi-objective Optimization Problems (MOPs). In this case, it is important to note that if the objective functions involved in the optimization process are not conflicting, the problem has only one optimal solution. Assuming the objective functions are conflicting with each other, MOPs present a set of solutions considered optimal, and the idea of a global optimal solution is then replaced by the Pareto-optimal Front (FONSECA; FLEMING, 1993; HORN; NAFPLIOTIS; GOLDBERG, 1994; SRINIVAS; DEB, 1994; ZITZLER; THIELE, 1999; KNOWLES; CORNE, 1999; DEB et al., 2002). One of the most common and simple strategy for solving MOPs is the weighted-sum method, which consists of obtaining a single objective function using the scalar product between a vector of weights and the objective functions. Moreover, the main disadvantage is that it can not generate all solutions in problems with non-convex Pareto Fronts (COELLO; LAMONT; VELDHUIZEN, 2007).

Feature selection is naturally formulated as a multi-objective optimization problem in which the two objective functions are: (i) to minimize the size of the subset of selected features and (ii) to maximize the classification accuracy (OLIVEIRA et al., 2002). Xue et al. (XUE; ZHANG; BROWNE, 2013) proposed two multi-objective versions of the well-known Binary PSO, being one based on mutual information and the other based on entropy. The results in six benchmark datasets have shown the proposed approaches evolved to Pareto front. Peimankar et al. (PEIMANKAR et al., 2017) introduced the binary Multi-objective Particle Swarm Optimization (MOPSO) for fault diagnosis of power transformers. Multi-objective feature selection and ensemble classifier selection are employed for dissolved gas analysis (DGA) of power transformers. The proposed method achieved good results for fault classification when compared to a multi-objective ensemble classifier without feature selection, random forests, and a decision tree called Oblique Random Forests.

Deniz et al. (DENIZ et al., 2017) combined multi-objective GA with machine learning techniques for feature selection in binary classification problems. The idea is to select the minimum number of features while preserving or increasing the classification accuracy. The performance was evaluated on eleven benchmark datasets and the proposed approach achieved better results than Greedy Search (GS), PSO, Tabu Search (TS), and Scatter

Search (SS) on most datasets. Later on, in the same context, Kiziloz et al. (KIZILOZ et al., 2018) combined the Teaching-Learning Based Optimization (TLBO) with machine learning classifiers to handle multi-objective feature selection problem. The experiments were carried out on thirteen UCI benchmark datasets demonstrating similar results compared with Non-dominated Genetic Algorithm (NSGA-II) while performing better than PSO, TS, GS, and SS. Kozodoi et al. (KOZODOI et al., 2019) employed the NSGA-II to the task of feature selection on credit scoring. The main idea was to use a profit measure and the number of features as fitness functions. Experiments on ten credit scoring datasets demonstrated the proposed feature selection approach achieved a fewer number of features than Sequential Forward Selection (SFS), Sequential Backward Selection (SBS), Least Absolute Shrinkage and Selection Operator (LASSO), GA, and PSO.

Although much effort has been devoted to developing evolutionary multi-objective optimization techniques as long as many real-world applications often involve four or more objective functions to be optimized. Nowadays, handling a large number of objectives, also known as many-objective optimization, has been one of the major research topics in the context of optimization problems (ISHIBUCHI; TSUKAMOTO; NOJIMA, 2008; CHAND; WAGNER, 2015). Since nature-inspired meta-heuristic algorithms have demonstrated suitable results in single-objective optimization problems concerning feature selection, our work aims to extend the idea to solve multi- and many-objective feature selection optimization problems.

Recently, a new butterfly-inspired meta-heuristic algorithm was developed by Qi et al. (QI; ZHU; ZHANG, 2017), which is based on the preference of speckled woods by finding warm sunspots in the woodlands. The Artificial Butterfly Optimization (ABO) algorithm concerns the mate-finding strategy since the sunspots are the best place to find females. ABO provides flight strategies that allow the construction of different algorithms. The overall performance of a meta-heuristic algorithm depends basically on its balance between exploration and exploitation. Thus, ABO divides the entire population into two groups of butterflies. The sunspot butterfly group is responsible for the exploitation process, i.e., a local search in a limited region of the search space with the hope of improving a promising solution; while the canopy butterfly group is responsible for the exploration process, i.e., consists of probing a much larger portion of the search space with the hope of finding other promising solutions.

In this paper, we propose two different binary versions of ABO for feature selection purposes: (i) a single-objective and (ii) a multi-objective one. Two approaches are introduced: (i) to minimize the classifier error for each class over the evaluating set, and (ii) to minimize the classifier error for each class over the evaluating set and also minimizing the number of selected features. The proposed approach is compared against swarm-family algorithms, such as PSO, FA, FPA, BSO, and BHA in several datasets. Such techniques are considered due to their widespread usage in the literature.

In short, this paper has the following main contributions:

- A binary version of the ABO for feature selection purposes;
- Multi- and many-objective versions of the Binary ABO (MOABO); and
- Two different approaches for MOABO in the context of feature selection.

The remainder of this paper is organized as follows. Sections 7.2 and 7.3 present the theoretical background regarding Artificial Butterfly Optimization and multi- and many-objective optimization, respectively. Furthermore, Section 7.4 presents the proposed approach concerning multi- and many-objective feature selection optimization. Section 7.5 discusses the experiments, and Section 7.6 states conclusions and future works.

7.2 Artificial Butterfly Algorithm

Inpired on the mate-finding strategy of speckled woods, Qi et al. (QI; ZHU; ZHANG, 2017) proposed a new meta-heuristic algorithm called Artificial Butterfly Optimization. The speckled woods prefer to live on the borders of woodlands where the sun shines on trees and create lots of sunspot. Some rules are made to idealize the mate-finding strategies of butterflies in ABO algorithm:

- In order to increase the likeliness of encoutering female butterflies, all male butterflies attempt to fly towards a better location called a sunsport;
- To ocupy a better sunspot, each sunspot butterfly always attempts to fly to its neighbor's sunspot; and
- Each canopy butterfly continually flies towards any sunspot butterfly to contend for the sunspot.

The butterfly population is sorted and divided into two groups according to their fitness. Butterflies with better fitness form the sunspot butterflies and the rest form canopy butterflies, and a different flight strategy is applied to each group. Three flight modes compose the ABO algorithm including sunspot flight mode, canopy flight mode, and free flight mode.

The following strategy is used for the sunspot flight mode or the canopy flight mode: each butterfly flies towards a randomly selected butterfly as follows:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + (x_{i,j}^{t} - x_{k,j}^{t})\beta, \tag{7.1}$$

where i is the i^{th} butterfly, j is a randomly selected dimension index between $[1,N]$, t is the current iteration, β is a random generated number between $[1,-1]$, and k is a randomly selected butterfly $(k \neq i)$.

Additionally, the following strategy can be used for the sunspot flight mode or canopy flight mode as well: each butterfly flies towards a randomly selected sunspot butterfly as follows:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + \frac{x_{k,j}^{t} - x_{i,j}^{t}}{x_{k,j}^{t} - x_{i,j}^{t}}(U-L)s\beta, \tag{7.2}$$

where U and L stands for lower and upper bound of the flying range for the i^{th} butterfly. The s parameter decreases linearly from 1 to s_e, as follows:

$$s = 1 - (1 - s_e)\frac{t}{T}, \tag{7.3}$$

where T denotes the max iteration count.

On the free flight mode, each butterfly flies towards a randomly new position to enhance the exploration phase in ABO algorithm using the following strategy:

$$x_{i,j}^{t+1} = x_{k,j}^{t} - 2\alpha\beta - \alpha D, \tag{7.4}$$

where α linearly decreases from 2 to 0 over the course of iteration, D is a randomly generated value as follows:

$$D = |2\beta(x_{k,j}^{t} - x_{i,j}^{t})|. \tag{7.5}$$

7.3 Multi- and Many-Objective Optimization

In this section, a theoretical background concerning multi-objective optimization using Pareto-dominance approach (DEB et al., 2002) is introduced, in which the idea is to optimize two or more objective functions at the same time. A multi-objective optimization problem aims at finding the global minimum $\boldsymbol{x}^* \in \mathscr{S}$ that minimizes a function set $\boldsymbol{M}$ represented by $\boldsymbol{f}$, i.e.:

$$\boldsymbol{x}^* = \arg \min_{\forall \boldsymbol{x} \in \mathscr{S}} (\boldsymbol{f}(\boldsymbol{x})) = \arg \min_{\forall \boldsymbol{x} \in \mathscr{S}} (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})), \tag{7.6}$$

subject to:

$$g_i(\boldsymbol{x}) = 0 \quad \forall i = 1, 2, \ldots, p, \tag{7.7}$$

$$h_i(\boldsymbol{x}) \geq 0 \quad \forall i = 1, 2, \ldots, q. \tag{7.8}$$

The set of all values satisfying the above constraints defines the feasible region, and any point in this region is thus considered a feasible solution. In a multi-objective problem, there is no single solution that is optimal with respect to all objectives when considering conflicting objectives. Thus, the solution to a multi-objective optimization problem is no longer a scalar value, but a vector in the form of a "trade-off" known as *Pareto-optimal set*.

Firstly, we define the *Pareto Dominance*, where a solution vector $\boldsymbol{x}^a$ is said to dominate another solution vector $\boldsymbol{x}^b$ (i.e., $\boldsymbol{x}^a \prec \boldsymbol{x}^b$) if $f(x_i^a) \geq f(x_i^b), \forall i = \{1, 2, \ldots, N\}$, and $\exists i \in \{1, 2, \ldots, N\}$ such that $f(x_i^a) > f(x_i^b)$. In regard to the Pareto Dominance, a solution vector $\boldsymbol{x}^a$ is considered Pareto-optimal if, for every $\boldsymbol{x}^b$, $f_j(\boldsymbol{x}^a) \geq f_j(\vec{x}^b)$, $j = 1, 2, \ldots, \boldsymbol{M}$, and if there exists at least one $j \in \{1, 2, \ldots, \boldsymbol{M}\}$ such that $f_j(\boldsymbol{x}^a) > f_j(\boldsymbol{x}^b)$. Therefore, the Pareto-optimal set $\mathscr{P}^*$ considering a multi-objective optimization problem $\boldsymbol{f}(\boldsymbol{x})$ with respect to all Pareto-optimal solutions is thus defined as follows:

$$\mathscr{P}^* = \{\boldsymbol{x} \in \mathscr{S} \mid \boldsymbol{f}(\boldsymbol{x}) \prec \boldsymbol{f}(\boldsymbol{x}'), \ \forall \boldsymbol{x}' \in \mathscr{S}\}. \tag{7.9}$$

The Pareto-optimal front $\boldsymbol{PF}^*$ with respect to a multi-objective optimization problem $\boldsymbol{f}(\boldsymbol{x})$ and the Pareto-optimal set $\mathscr{P}^*$ is defined as follows:

$$PF^* = \{\boldsymbol{f}(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathscr{P}^*\}. \tag{7.10}$$

In addition to Pareto-dominance algorithms, two more well-known categories can be found in the literature currently: (i) indicator-based algorithms (ZITZLER; KÜNZLI, 2004) and (ii) decomposition-based algorithms (ZHANG; ROCKETT, 2007). Although most of these algorithms perform well on MOPs considering two or three objectives, but they suffer some degrees of deterioration on their performance when dealing with many-objective optimization problems (MaOPs).

Many-objective[1] optimization (LI et al., 2015) is considered a special case of multi-objective optimization when we have four or more conflicting objective functions ($M \geq 4$) to be optimized at the same time. When dealing with MaOPs some difficulties may be highlighted on algorithms such as the number of non-dominated solutions that approximates the entire Pareto front. The Pareto front consists of a ($M-1$)-dimensional trade-off hyperspace in the objective space making the number of solutions required to fill the entire surface exponential in M.

Dominance-based algorithms suffer deterioration in the search ability where most of the solutions become non-dominated due to a weak dominance selection pressure toward the Pareto front affecting the convergence property. Also, increasing the number of objective functions makes visualization of non-dominated solutions become very difficult, that is, the choice of a final solution ends up being impaired. Concerning indicator-based algorithms, computing the indicator such as hypervolume could be costly in most cases. Already when the subject is algorithms based on decomposition, they suffer with the increasing number of objectives and face difficulties on configuring the weighting vector and choosing appropriate scalarizing methods.

7.4 Proposed Approach

In this section, we first present the proposed binary version of the Artificial Butterfly Optimization for single-objective optimization. Further, we present the proposed multi- and many-objective optimization approaches for feature selection purposes.

[1] The name “many-objective”was suggested for the OR community (FARINA; AMATO, 2002).

7.4.1 Binary Artificial Butterfly Optimization

In the standard ABO, the solutions are updated in the search space towards continuous-valued positions. However, in the proposed BABO, the search space is modelled as an N-dimensional boolean lattice, in which the solutions are updated across the corners of a hypercube. In addition, as the problem is to select or not a given feature, a solution binary vector is employed, where 1 corresponds whether a feature will be selected to compose the new dataset, and 0 otherwise. To obtain such binary vector, we employed Equation 7.12 right after Equation 7.4, which can restrict the new solutions to only binary values:

$$S(x_{i,j}^{t}) = \frac{1}{1+e^{-x_{i,j}^{t}}}, \tag{7.11}$$

$$x_{i,j}^{t} = \begin{cases} 1 & \text{if } S(x_{i,j}^{t}) > \sigma, \\ 0 & \text{otherwise} \end{cases} \tag{7.12}$$

where $\sigma \sim U(0,1)$. Algorithm 6 presents the proposed Binary ABO for feature selection using a classifier's recognition rate as the objective function.

Lines $1-4$ initialize the algorithm assigning random values to each butterfly's position, as well as the fitness value f_i of each individual i. The main loop in Lines $6-34$ concerns the stop criterion, in other words, the core of the proposed algorithm. Line 7 sorts all butterflies in ascending order according to their fitness value. Lines $9-16$ are responsible for updating the location of each sunspot butterfly according to Equation 7.1, and also for creating the new training Z_1' and evaluating sets Z_2'. Then, a classifier is trained over Z_1' and it is used to classify Z_2'. The recognition accuracy over Z_2' is stored in acc and then compared with the fitness value f_i (accuracy) of individual i: if the later is better than acc, the old fitness value is kept; in the opposite case, the fitness value is then updated.

Lines $13-16$ update the best local position of the current butterfly and Lines $17-21$ update the global optimum. Lines $22-29$ concern the loop in charge of updating the location of each canopy butterfly according to Equation 7.2. Lines $24-29$ compare acc with the fitness value f_i (accuracy) of individual i: if the later is worse than acc, the fitness value is updated; in the opposite case, it updates the location according to Equation 7.4. The last loop (Lines $30-34$) moves each butterfly to a new binary position restricted by Equation 7.12.

Algorithm 6: ABO - Artificial Butterfly Optimization

```
input       : Training set $Z_1$ and evaluating set $Z_2$, $\alpha$, number of butterflies $m$,
              dimension $d$ and iterations $T$.
output      : Global best position $\widehat{g}$.
auxiliaries: Fitness vector $f$ with size $m$ and variables $acc$, $maxfit$, $globalfit$ and
              $maxindex$.

for each butterfly $i$ $(\forall i = 1, \ldots, m)$ do
    for each dimension $j$ $(\forall j = 1, \ldots, d)$ do
        $x_i^j(0) \leftarrow \text{Random}\{0,1\}$;
    $f_i \leftarrow -\infty$;
$globalfit \leftarrow -\infty$;
for each iteration $t$ $(t = 1, \ldots, T)$ do
    Sort all butterflies by their fitness;
    Divide the entire population into sunspot and canopy butterflies according to
      their fitness;
    for each sunspot butterfly $i$ $(\forall i = 1, \ldots, sunspot)$ do
        Update the location according to sunspot flight mode;
        Create $Z_1'$ and $Z_2'$ from $Z_1$ and $Z_2$, respectively, such that both contains only
          features such that $x_i^j(t) \neq 0$, $\forall j = 1, \ldots, d$;
        Train the classifier over $Z_1'$, evaluate its over $Z_2'$ and stores the accuracy in
          $acc$;
        if $(acc > f_i)$ then
            $f_i \leftarrow acc$;
            for each dimension $j$ $(\forall j = 1, \ldots, d)$ do
                $\widehat{x}_i^j \leftarrow x_i^j(t)$;
    $[maxfit, maxindex] \leftarrow max(f)$;
    if $(maxfit > globalfit)$ then
        $globalfit \leftarrow maxfit$;
        for each dimension $j$ $(\forall j = 1, \ldots, d)$ do
            $\widehat{g}^j \leftarrow x_{maxindex}^j(t)$;
    for each canopy butterfly $i$ $(\forall i = sunspot, \ldots, m)$ do
        Update the location according to canopy flight mode;
        if $(acc > f_i)$ then
            $f_i \leftarrow acc$;
            for each dimension $j$ $(\forall j = 1, \ldots, d)$ do
                $\widehat{x}_i^j \leftarrow x_i^j(t)$;
        else
            Update the location according to the free flight mode;
    for each butterfly $i$ $(\forall i = 1, \ldots, m)$ do
        for each dimension $j$ $(\forall j = 1, \ldots, d)$ do
            if $(\sigma < \frac{1}{1+e^{x_i^j(t)}})$ then
                $x_i^j(t) \leftarrow 1$; else
                    $x_i^j(t) \leftarrow 0$;
```

7.4.2 Multi- and Many-objective Optimization

The weighted-sum method is one of the most common used technique to handle multi-objective optimization, in which the objective functions are combined into a single one through a weight vector. Thus, a problem with multiple objective functions is reduced to a single optimization problem subject to the original constraints, and the choice of the value of each weight is in accordance with the preference assigned to each objective function.

In this work, we propose two different many-objective feature selection approaches using the weighted-sum method, where the main idea is to select the subset of features that minimizes the classification error. The first approach consists of minimizing the classification error of each class individually. Mathematically speaking, the fitness function that will guide each agent into the best solution can be formulated as follows:

$$\mathbf{f} = \arg \min_{\forall \mathbf{x} \in \mathscr{S}} \left(\sum_{i=1}^{M} w_i f_i(\mathbf{x}) \right), \tag{7.13}$$

where $\sum_{i=1}^{k} w_i = 1$ with $w \geq 0$, $f_i(\mathbf{x})$ stands for the classification error over the evaluating set for class i, and M denotes the total number of classes (objective functions). Figure 7.1 displays a detailed pipeline of the proposed approach MO-I.

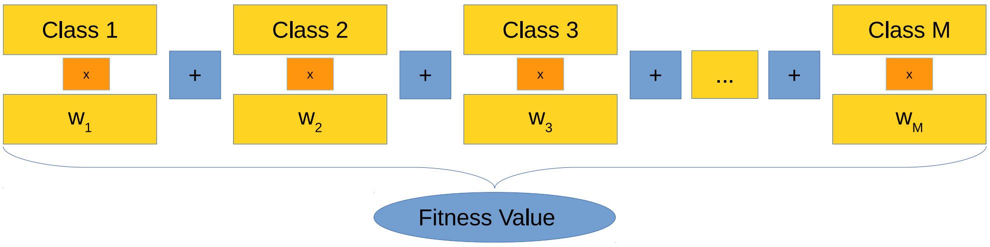


Figure 7.1: Proposed approach MO-I.

Furthermore, the second approach is an extension to the first one with a minor difference in which we want to minimize the feature set size and the classification error for each class as well. The fitness function is presented as follows:

$$\mathbf{f} = \arg \min_{\forall \mathbf{x} \in \mathscr{S}} \left(\sum_{i=1}^{M} w_i f_i(\mathbf{x}) + w_{M+1} f_{M+1}(\mathbf{x}) \right), \tag{7.14}$$

where $f_{M+1}(\mathbf{x})$ denotes the number of features. Figure 7.2 displays a detailed pipeline of the proposed approach MO-II.

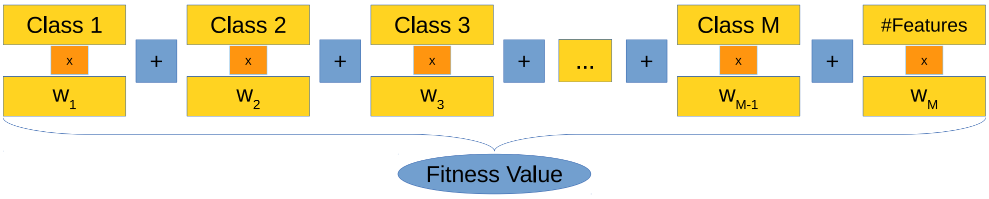


Figure 7.2: Proposed approach MO-II.

7.5 Experimental Setup

In this section, we present the experimental evaluation considering the proposed BABO and its multi- and many-objective versions for feature selection[2]. We set the number of possible solutions $m = 30$, and the number of iterations $T = 60$. Additionally, we used 50%, 30% and 20% for the training, evaluating and testing set percentages, respectively[3]. Regarding the source code, we used the LibOPT (Papa et al., 2017)[4]. We opted to evaluate the proposed techniques using the Optimum-Path Forest (OPF) classifier (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012) since it is parameterless and fast for training. Note the proposed approach can be used with any other supervised classification technique. Figure 7.3 displays the whole pipeline adopted in the paper.

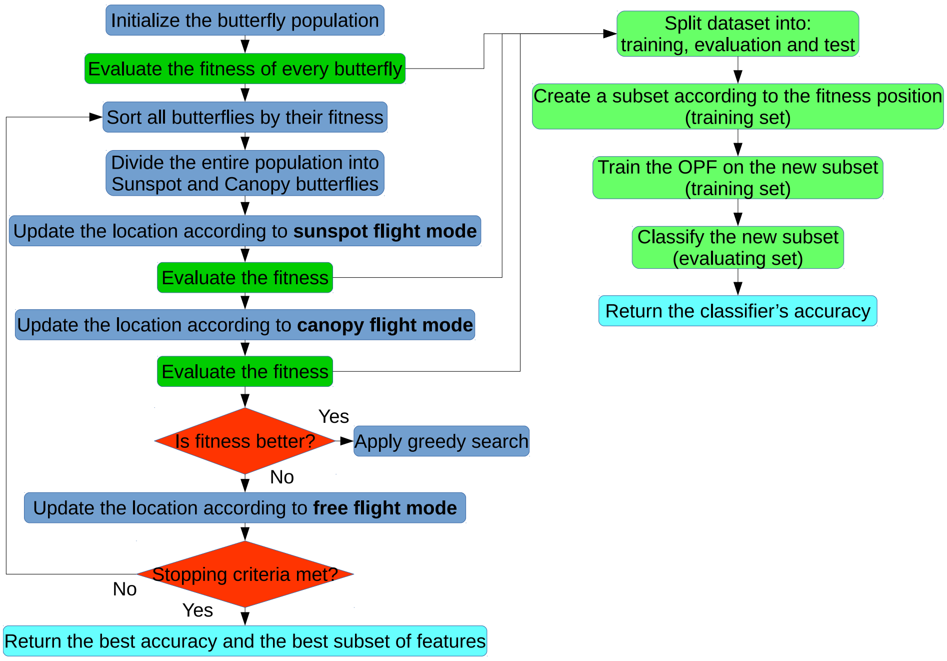

Initialize the butterfly population
Evaluate the fitness of every butterfly
Sort all butterflies by their fitness
Divide the entire population into Sunspot and Canopy butterflies
Update the location according to **sunspot flight mode**
Evaluate the fitness
Update the location according to **canopy flight mode**
Evaluate the fitness
Is fitness better?
Yes
Apply greedy search
No
Update the location according to **free flight mode**
Stopping criteria met?
No
Yes
Return the best accuracy and the best subset of features
Split dataset into: training, evaluation and test
Create a subset according to the fitness position (training set)
Train the OPF on the new subset (training set)
Classify the new subset (evaluating set)
Return the classifier's accuracy

Figure 7.3: Pipeline of the entire feature selection process.

[2] The experiments were executed in a computer with a Pentium Intel Core *i7*® 1.73Ghz processor, 6 GB of RAM and Linux Ubuntu Desktop LTS 13.04 as the operational system.

[3] These fold values were empirically chosen.

[4] `https://github.com/jppbsi/LibOPT`

We considered eight benchmark datasets from UCI repository[5]. Table 7.1 presents the number of samples, number of features, and the number of classes for each dataset. Notice we decided to evaluate the robustness of the proposed approaches under different scenarios. Table 7.2 exhibits the parameter configuration for every meta-heuristic technique[6].

Table 7.1: Dataset descriptions

Dataset	Samples	Classes	Features
German Numer (D1)	1000	2	24
Ionosphere (D2)	351	2	34
MPEG-7 (D3)	1,400	180	70
Pendigits (D4)	7494	10	16
Satimage (D5)	4,435	36	6
Sonar (D6)	208	2	60
Splice (D7)	1000	2	60
SVM Guide 2 (D8)	391	3	20

Table 7.2: Parameters used for each meta-heuristic algorithm.

Technique	Parameters
BHA	–
BSO	$k = 3 \mid p\_one\_cluster = 0.3$
	$p\_one\_center = 0.4 \mid p\_two\_centers = 0.3$
FPA	$\beta = 1.5 \mid p = 0.8$
PSO	$c1 = 1.7 \mid c2 = 1.7 \mid w = 0.7$
FA	$\alpha = 0.2 \mid \beta = 1.0 \mid \gamma = 1.0$
ABO	$ratio_e = 0.2 \mid step_e = 0.05$

7.5.1 Binary Single-objective ABO Evaluation

In this section, we evaluate the Binary ABO concerning the task of feature selection considering optimizing the OPF classification accuracy. Table 7.3 presents the mean OPF classification accuracy and standard deviation over the test set considering 15 runs for the single-objective binary version of PSO, FA, FPA, BSO, and BHA techniques. Notice that the values in bold stand for the best accuracy values.

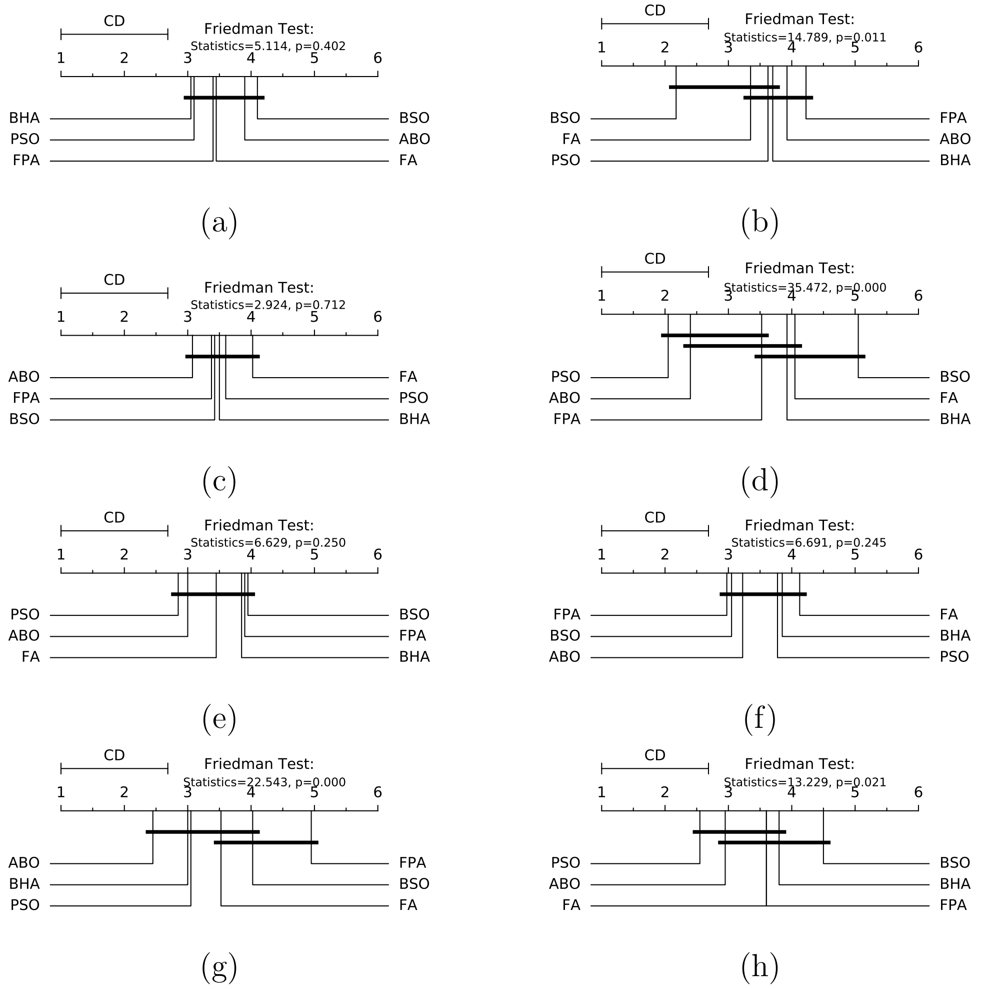
Additionally, we evaluated the results using the Friedman test (FRIEDMAN, 1937, 1940), which is a non-parametric statistical test used to detect differences on multiple groups. Following, we adopted the Nemenyi post-hoc test (NEMENYI, 1963) to identify groups that differ from each other as displayed in Figure 7.4. In this paper, we adopted 0.05 (5%) as the significance level.

[5] `http://archive.ics.uci.edu/ml`

[6] Note that these values were empirically chosen according to their author's definition.

Table 7.3: Average classification accuracy over the test set and the total number of selected features.

	BHA		BSO		FPA		PSO		FA		ABO	
Dataset	OPF Accuracy	#Feats	OPF Accuracy	#Feats	OPF Accuracy	#Feats	OPF Accuracy	#Feats	OPF Accuracy	#Feats	OPF Accuracy	#Feats
D1	0.586±0.029	16.2±2.7	0.573±0.033	16.8±1.7	**0.590±0.035**	15.7±2.2	0.589±0.039	15.3±1.6	0.583±0.038	13.8±2.5	0.584±0.023	15.7±2.6
D2	0.812±0.033	22.6±2.6	0.841±0.041	21.7±2.7	0.835±0.031	21.3±3.1	0.842±0.035	22.0±2.9	**0.842±0.048**	21.5±3.1	0.837±0.055	22.3±2.4
D3	0.911±0.071	120.8±5.0	0.912±0.060	116.9±7.0	0.915±0.008	119.1±4.3	**0.920±0.009**	118.2±4.9	0.914±0.008	119.5±2.7	0.913±0.009	114.3±6.2
D4	0.995±0.014	13.6±0.8	0.990±0.038	11.6±1.4	0.994±0.002	12.3±1.3	**0.995±0.002**	13.2±1.5	0.993±0.004	12.7±1.7	0.988±0.007	10.5±1.8
D5	**0.927±0.070**	26.4±1.4	0.924±0.071	23.6±2.9	0.923±0.006	25.2±2.0	0.926±0.005	23.9±1.5	0.921±0.007	25.7±2.5	0.924±0.007	23.0±2.7
D6	0.822±0.466	40.2±3.1	0.844±0.532	39.7±2.9	0.818±0.059	40.3±1.6	0.816±0.037	39.9±3.2	**0.851±0.042**	40.8±2.8	0.836±0.062	38.8±2.3
D7	**0.699±0.251**	40.6±3.6	0.680±0.273	39.1±3.4	0.698±0.043	39.5±3.1	0.695±0.042	39.4±2.8	0.695±0.038	39.7±4.1	0.679±0.032	40.5±1.8
D8	**0.734±0.300**	14.5±1.8	0.702±0.539	12.9±1.9	0.706±0.036	14.9±0.9	0.720±0.043	14.7±2.1	0.725±0.048	14.7±1.6	0.712±0.026	12.9±2.0



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 7.4: Friedman test followed by Nemenyi post-hoc test considering (a) D1, (b) D2, (c) D3, (d) D4, (e) D5, (f) D6, (g) D7, and (h) D8 datasets.

According to Figure 7.4, ABO demonstrated a different behavior in Pendigits and Splice datasets, since all techniques behave statistically similar in all other datasets. In a brief look at Table 7.3, one can notice that even ABO achieved the lowest accuracy value, all other techniques are still very close to each other. Additionally, one can also notice in Table 7.3 that ABO selected fewer features compared to the other techniques in all datasets, except for German Numer, Ionosphere and Splice datasets, where FA, FPA, and PSO selected fewer features, respectively. Figure 7.5 displays the average training time on each dataset. Notice the whole optimization step comes down to multiplying the classifier training time, the number of agents, and the number of iterations of each meta-heuristic technique.

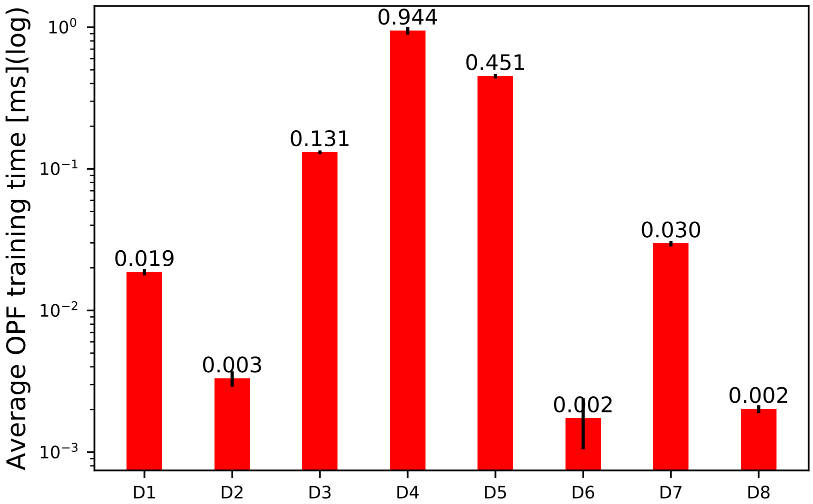


Figure 7.5: OPF training time considering each dataset.

Table 7.4 presents the classification time over the test set (milliseconds) regarding all single-objective meta-heuristic techniques. Notice that the values in bold stand for the lowest classification times. BSO was the fastest technique in six out of eight datasets, as presented in Table 7.4. FA and ABO achivied the lowest classification time in MPEG-7 and Pendigits datasets, respectively.

Table 7.4: Average classification time [ms] over the test set.

Dataset	BHA	BSO	FPA	PSO	FA	ABO
D1	74.920±0.498	**37.909±0.449**	63.788±1.065	43.573±1.229	43.173±0.9136	40.495±0.292
D2	10.630±0.051	**5.471±0.037**	9.022±0.475	6.600±0.224	6.344±0.2469	6.244±0.211
D3	471.709±1.433	242.691±1.093	281.627±7.688	245.074±7.161	**238.442±5.250**	241.467±7.024
D4	4216.023±0.311	**2140.515±0.276**	3573.888±30.457	2519.392±95.506	2417.530±56.800	2525.624±102.058
D5	1998.749±56.837	**1033.594±20.971**	1463.046±42.363	1093.093±35.372	1068.509±36.047	1097.053±30.139
D6	4.945±0.066	**2.614±0.056**	3.962±0.220	2.870±0.121	2.949±0.079	2.866±0.107
D7	103.648±0.177	**52.408±0.321**	71.757±0.378	56.773±2.106	58.409±1.735	54.436±0.296
D8	8.695±0.251	4.400±0.064	7.546±0.464	4.302±0.047	4.511±0.192	**4.272±0.034**

7.5.2 Multi- and Many-objective Binary ABO Evaluation

In this section, we evaluate the proposed multi- and many-objective Binary ABO in the context of feature selection. In order to compose an optimal Pareto front, 15 runs were considered for each approach. Also, random weight values were generated for both methods on each run. Since we have a search space of possible optimal solutions, the best result for each dataset depends entirely on the choice of the decision maker. However, to evaluate the techniques compared in this work, we used a measure F that considers the results of both fitness functions as follows:

$$F = \frac{\text{accuracy over the test set}}{\text{training set size}}. \tag{7.15}$$

Therefore, since one aims at using smaller training sets and obtaining higher accuracy rates, the greater the value of F, the better the technique is. Tables 7.5, 7.6, 7.7, 7.8, 7.9 and 7.10 present the F values for each optimization technique considered in this work, where the values in bold stand for the best results. It is worth noting that MO-I obtained the best F value in four datasets, i.e., German Numer, Ionosphere and Mpeg-7; while in Pendigits, Satimage, Sonar, Splice, and SVM-Guide 2 datasets, the best F values were achieved by MO-II. MOABO achieved good results in three datasets, i.e., German Numer, Satimage and Sonar, followed by MOPSO in Splice and SVM-Guide 2 datasets and MOFA in Ionosphere and Mpeg-7. Figure 7.6 summarizes the best F values of each technique considering each approach (MO-I and MO-II) in all databases.

Table 7.5: F values considering MOBHA using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.0357	0.0302	0.0271	0.0311	**0.0091**	0.0075	0.1226	0.0710	0.0368	0.0371	0.0221	0.0244	0.0195	0.0206	0.0762	0.1035
0.0446	0.0442	0.0273	0.0364	0.0073	0.0082	0.1537	0.1091	0.0402	**0.0580**	0.0232	0.0163	0.0146	0.0163	0.0592	0.0705
0.0454	0.0316	0.0342	0.0332	0.0082	0.0076	0.1097	0.1536	0.0414	0.0355	0.0165	0.0177	0.0197	0.0183	0.0856	0.0592
0.0413	0.0449	0.0346	**0.0420**	0.0076	0.0090	0.0828	0.1553	0.0395	0.0437	0.0220	0.0212	0.0148	0.0141	0.0782	0.0705
0.0266	0.0324	0.0356	0.0358	0.0072	0.0081	0.0982	0.0761	0.0416	0.0439	0.0191	0.0205	0.0193	0.0182	0.0637	0.0545
0.0470	0.0389	0.0321	0.0362	0.0083	0.0084	0.1581	0.1220	0.0401	0.0438	0.0162	0.0195	0.0185	0.0176	0.0708	0.0769
0.0427	0.0494	0.0402	0.0324	0.0070	0.0083	0.1575	0.1399	0.0353	0.0439	0.0169	0.0160	0.0191	0.0173	0.0511	0.0528
0.0333	0.0326	0.0351	0.0315	0.0088	0.0071	0.1379	0.2117	0.0461	0.0370	0.0200	0.0177	0.0178	0.0184	0.0479	0.0650
0.0524	0.0488	0.0314	0.0362	0.0082	0.0076	0.1376	0.1377	0.0407	0.0490	0.0170	0.0246	0.0141	0.0191	0.0573	0.0648
0.0454	0.0290	0.0319	0.0262	0.0070	0.0076	0.1786	0.1828	0.0354	0.0441	0.0180	0.0239	0.0156	0.0188	0.0672	0.0462
0.0505	0.0366	0.0306	0.0329	0.0077	0.0080	0.1389	0.1226	0.0387	0.0356	0.0237	0.0216	**0.0229**	0.0153	0.0575	0.0567
0.0442	0.0528	0.0327	0.0346	0.0078	0.0076	0.0892	0.0985	0.0388	0.0421	0.0225	0.0179	0.0217	0.0168	0.0509	0.0582
0.0259	0.0394	0.0278	0.0412	0.0080	0.0080	0.1390	0.1545	0.0387	0.0395	0.0238	0.0212	0.0134	0.0139	0.0665	**0.1156**
0.0452	0.0402	0.0361	0.0375	0.0078	0.0085	0.1378	0.1576	0.0340	0.0535	0.0220	**0.0283**	0.0172	0.0174	0.0747	0.0701
0.0555	0.0524	0.0343	0.0322	0.0091	0.0076	**0.2160**	0.1558	0.0320	0.0401	0.0261	0.0192	0.0198	0.0196	0.0669	0.0634

Tables 7.11, 7.12, 7.13, 7.16, 7.15 and 7.16 present the classification time over the testing set for each meta-heuristic technique considering both approaches MO-I and MO-II. One can observe that MO-I obtained the lowest classification time in six datasets, i.e., German Numer, Mpeg-7, Pendigits, Satimage, Splice and SVM-Guide 2. Concerning Ionosphere and Sonar datasets, MO-II obtained the lowest classification times. MOFA and MOABO were the fastest meta-heuristic techniques, achieving the best classification time in four datasets. These results indicate that MO-I was faster than MO-II, mainly due to the lower number of selected features.

Table 7.6: F values considering MOBSO using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.0389	0.0314	0.0412	0.0412	0.0081	0.0077	0.1228	0.1813	0.0371	0.0567	0.0194	0.0233	0.0196	0.0165	0.0498	0.0605
0.0274	0.0300	0.0418	0.0318	**0.0091**	0.0078	0.0712	0.0710	0.0418	0.0382	0.0181	0.0230	0.0164	0.0183	0.0734	**0.0896**
0.0346	0.0409	0.0383	0.0398	0.0073	0.0078	0.0828	0.0664	0.0419	0.0370	0.0232	0.0221	0.0161	0.0208	0.0646	0.0505
0.0428	0.0434	0.0387	0.0305	0.0068	0.0074	0.0980	0.1345	0.0397	0.0439	**0.0260**	0.0225	0.0157	0.0157	0.0573	0.0523
0.0476	0.0361	0.0423	0.0274	0.0075	0.0075	0.0710	0.1089	0.0424	0.0352	0.0178	0.0191	0.0192	0.0161	0.0573	0.0754
0.0315	0.0400	0.0356	0.0388	0.0078	0.0081	0.0902	0.1218	0.0486	0.0398	0.0184	0.0207	0.0179	0.0134	0.0702	0.0485
0.0429	0.0338	0.0369	**0.0498**	0.0083	0.0079	0.1370	**0.1906**	0.0398	0.0409	0.0151	0.0243	0.0160	0.0177	0.0560	0.0498
0.0368	0.0306	0.0324	0.0328	0.0081	0.0078	0.1569	0.1504	0.0341	0.0424	0.0185	0.0192	**0.0218**	0.0185	0.0561	0.0700
0.0281	**0.0560**	0.0387	0.0375	**0.0091**	0.0077	0.1352	0.0827	0.0487	0.0360	0.0214	0.0198	0.0169	0.0193	0.0613	0.0577
0.0347	0.0443	0.0393	0.0363	0.0073	0.0074	0.1217	0.1212	0.0423	0.0361	0.0184	0.0215	0.0186	0.0175	0.0499	0.0614
0.0369	0.0412	0.0299	0.0330	0.0073	0.0077	0.0898	0.1345	0.0383	0.0382	0.0217	0.0192	0.0200	0.0167	0.0662	0.0537
0.0352	0.0386	0.0370	0.0313	0.0082	0.0083	0.1100	0.0986	0.0380	0.0403	0.0220	0.0176	0.0164	0.0179	0.0627	0.0540
0.0359	0.0373	0.0355	0.0346	0.0079	0.0084	0.1231	0.0902	0.0387	0.0437	0.0189	0.0215	0.0161	0.0196	0.0510	0.0599
0.0327	0.0379	0.0393	0.0333	0.0070	**0.0091**	0.1086	0.0710	0.0329	**0.0571**	0.0191	0.0204	0.0203	0.0168	**0.0896**	0.0558
0.0327	0.0324	0.0334	0.0342	0.0085	0.0075	0.0828	0.1092	0.0363	0.0386	0.0193	0.0228	0.0207	0.0168	0.0588	0.0779

Table 7.7: F values considering MOFPA using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.0303	0.0346	0.0335	0.0458	0.0074	0.0076	0.1885	0.1494	0.0416	0.0419	0.0265	0.0230	0.0194	0.0176	0.0700	0.0693
0.0339	**0.0541**	0.0448	0.0380	0.0079	0.0079	0.1585	0.1387	0.0459	0.0420	0.0253	0.0199	0.0194	0.0150	0.0498	0.0585
0.0363	0.0324	0.0404	0.0412	0.0077	0.0072	0.0986	0.1690	0.0420	0.0513	0.0256	0.0198	0.0173	0.0163	0.0576	0.0483
0.0507	0.0326	0.0299	0.0388	0.0077	0.0078	0.0903	0.1385	0.0388	0.0442	0.0208	0.0176	0.0190	0.0183	0.0438	0.0597
0.0378	0.0437	**0.0486**	0.0343	0.0082	**0.0091**	0.1548	0.1606	0.0369	0.0367	0.0198	0.0262	**0.0215**	0.0202	0.0606	0.0616
0.0352	0.0384	0.0331	0.0371	0.0075	0.0081	0.0902	0.1556	0.0399	0.0483	0.0201	0.0221	0.0202	0.0169	0.0713	0.0531
0.0342	0.0426	0.0379	0.0345	0.0080	0.0088	0.0901	0.0901	0.0345	0.0390	0.0179	0.0183	0.0164	0.0160	0.0526	0.0434
0.0326	0.0346	0.0363	0.0361	0.0084	0.0081	0.2225	0.1564	0.0429	0.0355	0.0246	0.0187	0.0128	0.0204	0.0539	0.0701
0.0450	0.0335	0.0411	0.0343	0.0080	0.0076	0.1839	0.2061	**0.0652**	0.0444	0.0170	0.0246	0.0149	0.0202	0.0468	0.0654
0.0453	0.0408	0.0351	0.0418	0.0089	0.0076	0.1086	0.1533	0.0505	0.0386	**0.0289**	0.0172	0.0175	0.0159	0.0511	0.0609
0.0425	0.0366	0.0347	0.0410	0.0080	0.0071	0.1786	0.1815	0.0419	0.0299	0.0244	0.0221	0.0185	0.0160	0.0562	0.0568
0.0334	0.0245	0.0424	0.0429	0.0068	0.0074	0.1889	**0.2680**	0.0298	0.0399	0.0207	0.0200	0.0169	0.0188	0.0427	0.0458
0.0343	0.0396	0.0404	0.0456	0.0083	0.0086	0.1091	0.0708	0.0511	0.0343	0.0231	0.0202	0.0191	0.0167	0.0478	0.0589
0.0289	0.0340	0.0346	0.0356	0.0071	0.0074	0.1230	0.1587	0.0439	0.0458	0.0172	0.0247	0.0156	0.0173	0.0552	0.0656
0.0347	0.0361	0.0307	0.0334	0.0077	0.0083	0.1099	0.0984	0.0459	0.0376	0.0199	0.0220	0.0203	0.0173	0.0753	**0.0828**

Table 7.8: F values considering MOPSO using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.0423	0.0392	0.0299	0.0394	0.0075	0.0077	0.1823	0.0983	0.0371	0.0390	0.0198	0.0201	0.0179	0.0180	0.0470	0.0563
0.0431	0.0467	0.0369	0.0285	0.0077	0.0079	0.1368	0.1368	0.0342	0.0357	0.0217	0.0155	0.0171	0.0204	0.0547	0.0632
0.0590	0.0445	0.0474	0.0313	0.0080	0.0083	0.1214	0.0901	0.0512	0.0342	0.0193	0.0203	0.0171	0.0158	0.0584	0.0526
0.0438	0.0375	0.0315	0.0371	0.0071	0.0081	0.1582	0.1368	0.0321	0.0485	0.0207	0.0208	0.0155	0.0143	0.0522	0.0575
0.0394	0.0335	0.0335	0.0311	0.0074	0.0081	0.0828	0.1377	0.0385	0.0383	0.0213	0.0188	0.0152	0.0197	0.0613	0.0611
0.0420	0.0320	0.0354	0.0420	0.0080	0.0075	0.1801	0.1102	0.0423	0.0396	0.0169	0.0231	0.0179	0.0142	0.0574	0.0650
0.0570	0.0376	0.0331	0.0345	0.0075	0.0077	0.1572	0.1613	0.0299	0.0507	0.0206	0.0163	0.0222	0.0169	0.0580	0.0504
0.0400	0.0496	0.0346	0.0340	0.0076	0.0080	0.0710	0.0902	0.0419	0.0388	0.0182	0.0179	0.0183	0.0166	0.0695	0.0737
0.0428	0.0374	0.0371	0.0342	0.0086	0.0081	0.0987	0.1593	0.0343	0.0368	0.0214	0.0191	0.0154	0.0203	0.0540	0.0532
0.0391	0.0325	0.0375	0.0353	**0.0087**	0.0071	0.1547	0.0986	0.0386	0.0512	0.0218	0.0202	0.0181	0.0214	**0.1107**	0.0533
0.0319	0.0505	0.0308	0.0310	0.0074	0.0076	**0.1832**	0.0827	0.0357	0.0485	0.0177	0.0208	0.0179	0.0196	0.0559	0.0716
0.0436	0.0433	0.0319	0.0322	0.0074	0.0075	0.0710	0.1581	0.0483	0.0435	0.0185	0.0211	0.0167	0.0193	0.0697	0.0605
0.0321	0.0374	0.0309	0.0360	0.0067	0.0085	0.0901	0.1361	0.0484	0.0359	0.0245	0.0175	0.0164	0.0180	0.0473	0.0600
0.0397	0.0374	0.0388	0.0317	0.0070	0.0082	0.0665	0.0826	0.0439	0.0317	0.0239	0.0178	0.0221	**0.0243**	0.0633	0.0605
0.0369	0.0388	**0.0481**	0.0361	0.0083	0.0072	0.0767	0.1581	**0.0603**	0.0369	0.0194	**0.0264**	0.0169	0.0198	0.0643	0.0523

7.5.3 Single, Multi- and Many-objective Discussion

In this section, we discuss the results concerning the single-objective techniques against the multi- and many-objective versions. One can observe that MO-I and MO-II approaches obtained a better accuracy rate than their respective standard versions (i.e., single-objective) in all datasets. Also, one may observe that both MO-I and MO-II selected a

Table 7.9: F values considering MOFA using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.0337	0.0362	0.0411	0.0486	0.0081	0.0078	0.1226	0.1366	0.0393	0.0443	0.0261	0.0219	0.0186	0.0159	0.0495	0.0906
0.0481	0.0343	**0.0503**	0.0379	0.0080	0.0081	0.1583	0.1347	0.0372	0.0401	0.0186	0.0196	0.0161	0.0184	0.0685	0.0568
0.0335	0.0431	0.0287	0.0314	0.0086	0.0088	0.1806	0.1213	0.0484	0.0338	0.0168	0.0190	0.0170	0.0163	0.0537	0.0517
0.0431	0.0415	0.0309	0.0278	0.0084	0.0081	0.1346	0.1504	0.0396	0.0464	0.0231	0.0206	0.0210	0.0159	0.0593	0.0707
0.0296	0.0489	0.0323	0.0319	0.0079	0.0080	0.2125	0.0901	0.0403	0.0368	0.0208	0.0205	0.0199	0.0177	0.0660	0.0631
0.0525	0.0359	0.0323	0.0340	0.0079	0.0075	0.1213	0.1092	0.0457	0.0341	0.0190	0.0216	**0.0229**	0.0179	0.0582	0.0505
0.0515	0.0356	0.0348	0.0360	0.0086	0.0083	**0.2175**	0.1397	0.0383	0.0354	0.0237	0.0185	0.0189	0.0163	0.0483	0.0542
0.0563	0.0550	0.0358	0.0282	0.0072	0.0068	0.1324	0.1887	**0.0568**	0.0355	0.0191	0.0188	0.0158	0.0174	0.0514	0.0556
0.0345	0.0449	0.0361	0.0324	0.0085	0.0085	0.1789	0.1572	0.0353	0.0368	0.0232	**0.0302**	0.0145	0.0188	0.0775	0.0505
0.0323	0.0393	0.0363	0.0287	**0.0097**	0.0083	0.1564	0.1356	0.0364	0.0456	0.0198	0.0226	0.0192	0.0152	0.0529	0.0639
0.0402	0.0402	0.0324	0.0346	0.0069	0.0084	0.1353	0.1870	0.0358	0.0307	0.0181	0.0175	0.0160	0.0170	0.0712	0.0726
0.0428	0.0337	0.0411	0.0376	0.0079	0.0084	0.1845	0.1833	0.0419	0.0402	0.0226	0.0278	0.0201	0.0177	0.0494	0.0477
0.0402	0.0429	0.0450	0.0334	0.0088	0.0078	0.1868	0.1385	0.0370	0.0418	0.0174	0.0194	0.0196	0.0220	0.0642	**0.0959**
0.0298	0.0508	0.0361	0.0345	0.0076	0.0077	0.1363	0.1344	0.0329	0.0388	0.0166	0.0182	0.0185	0.0144	0.0797	0.0499
0.0300	0.0398	0.0277	0.0329	0.0083	0.0086	0.1363	0.0711	0.0369	0.0482	0.0244	0.0210	0.0177	0.0189	0.0687	0.0668

Table 7.10: F values considering MOABO using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.0324	0.0333	0.0361	0.0321	0.0080	0.0078	0.1086	0.1488	0.0422	0.0352	0.0183	0.0195	0.0165	0.0162	0.0522	0.0781
0.0612	0.0404	0.0337	0.0380	0.0081	0.0079	0.0827	0.1562	0.0415	0.0459	0.0188	0.0226	0.0156	0.0135	0.0537	0.0647
0.0365	0.0308	0.0264	0.0450	0.0073	0.0083	0.1393	0.1196	0.0400	0.0487	0.0207	0.0233	0.0145	0.0187	0.0681	**0.0905**
0.0354	0.0342	0.0354	0.0408	0.0084	0.0078	0.0895	**0.1848**	0.0424	0.0433	0.0207	**0.0308**	0.0158	0.0161	0.0593	0.0518
0.0340	0.0588	0.0299	0.0456	0.0076	0.0082	0.0991	0.1217	0.0487	0.0387	0.0211	0.0252	0.0196	0.0155	0.0839	0.0764
0.0362	0.0269	0.0321	0.0321	0.0076	0.0075	0.1094	0.1361	0.0444	0.0286	0.0203	0.0193	0.0194	0.0153	0.0888	0.0649
0.0364	0.0369	0.0299	0.0385	0.0088	0.0078	0.0765	0.0763	0.0386	0.0388	0.0192	0.0178	0.0177	0.0188	0.0463	0.0712
0.0444	0.0357	0.0416	0.0299	0.0085	0.0076	0.0824	0.0764	0.0400	0.0385	0.0225	0.0183	0.0189	0.0177	0.0546	0.0479
0.0282	0.0414	0.0458	0.0321	0.0072	0.0085	0.0902	0.0827	0.0357	**0.0658**	0.0238	0.0199	0.0192	0.0189	0.0547	0.0552
0.0402	0.0419	0.0306	**0.0496**	0.0078	0.0073	0.0990	0.0824	0.0373	0.0357	0.0220	0.0190	0.0162	**0.0223**	0.0686	0.0613
0.0310	0.0283	0.0465	0.0345	0.0072	0.0076	0.1219	0.1079	0.0367	0.0403	0.0214	0.0191	0.0169	0.0164	0.0813	0.0425
0.0399	0.0340	0.0331	0.0346	**0.0090**	0.0086	0.0825	0.0989	0.0435	0.0384	0.0203	0.0226	0.0184	0.0155	0.0584	0.0505
0.0467	0.0350	0.0348	0.0324	0.0075	0.0077	0.1224	0.0764	0.0403	0.0486	0.0218	0.0238	0.0211	0.0156	0.0540	0.0642
0.0413	0.0453	0.0403	0.0311	0.0079	0.0085	0.0896	0.0823	0.0371	0.0422	0.0166	0.0199	0.0153	0.0136	0.0503	0.0434
0.0296	0.0389	0.0396	0.0388	0.0084	0.0078	0.1784	0.1209	0.0344	0.0333	0.0220	0.0187	0.0159	0.0176	0.0499	0.0696

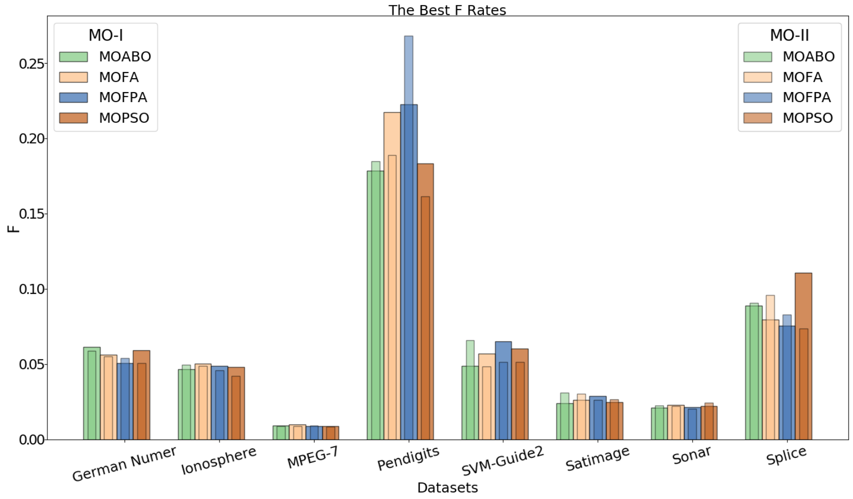


Figure 7.6: Best results concerning the F values.

lower number of features.

Notice that another main contribution of the paper is to model the problem of feature selection as a multi- and many-objective task. Therefore, we can observe that both MO-I and MO-II outperformed their counterpart single-objective versions in most datasets

Table 7.11: OPF classification time [ms] over the test set considering MOBHA using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
76.3525	73.9674	10.4353	10.9200	475.7919	**466.7612**	4468.6731	4960.8536	1861.6407	1862.5371	4.7852	4.7736	102.3345	102.0975	8.4965	9.4799
76.2478	76.1620	**10.3576**	11.5511	456.0902	470.4450	4488.7367	4805.6612	1858.1222	1877.8943	4.7985	4.8784	103.6787	105.4967	8.4665	8.8594
73.6071	78.4875	10.4316	10.8994	456.1306	465.5445	4490.5317	4904.4951	1854.1709	1866.0442	4.8425	4.8915	99.7426	99.1258	8.3894	8.7578
76.0670	**73.2821**	10.6193	10.3956	456.5861	492.5624	4499.1908	4563.1006	1922.3684	1999.0895	4.8718	4.8395	101.1446	104.7712	8.4343	8.6316
74.0452	73.9046	10.8389	10.4056	455.2781	478.4762	4486.0838	4497.5739	1888.7008	2087.3902	4.9159	4.8254	**99.0979**	99.2927	**8.3487**	9.6155
79.6690	76.4635	11.4443	10.6057	498.6324	478.2060	4438.7327	4666.2598	1939.4828	1997.8588	5.2098	4.9564	113.7183	104.5529	8.8860	9.3844
79.7321	77.1113	11.2945	10.9529	512.2357	476.6463	4478.4117	4852.3541	1987.5257	1963.7761	5.1840	**4.7196**	111.9663	106.9978	8.7900	8.6602
72.8248	80.8094	11.4344	10.8462	497.5652	516.4653	4500.0220	4875.6429	1877.6223	2812.3227	5.1915	5.0819	109.7311	99.3245	8.5980	8.8478
78.2009	77.5871	11.2526	10.5806	474.3244	532.1664	4499.1908	4498.2014	1890.9014	2001.5905	5.1251	4.7412	115.2204	99.5331	8.7863	8.6314
77.5113	74.2968	11.2783	10.5236	479.8628	501.2253	4432.2231	4445.7361	1984.4277	1974.2634	5.2670	4.8981	108.2694	102.0338	8.7879	8.8133
76.8732	75.8550	11.4274	10.9151	477.2765	502.0884	**4398.1311**	4977.8857	1965.7587	1947.0895	5.1985	5.0862	108.2057	102.5087	8.9218	8.5597
78.0177	75.2072	11.3500	10.4640	491.4626	488.2510	4423.7227	4409.5567	1889.7171	1882.5639	5.3856	5.1101	108.7937	103.6906	9.4309	9.2780
80.8148	76.1918	11.2947	10.5295	492.2315	479.2160	4491.2617	4915.4587	1843.7335	1848.7258	5.3038	5.0725	108.2708	102.3637	9.0284	9.1592
75.7043	75.3238	11.1212	10.7905	483.4474	475.4177	4499.1029	4622.3314	1848.3921	1857.9091	5.2311	5.0414	108.5772	103.6296	8.8264	9.7806
77.2615	75.7388	11.4949	10.5042	473.6927	503.1625	4494.0228	4501.6122	**1835.6977**	2015.7364	5.1540	4.7512	102.0545	102.6809	8.9079	9.2458

Table 7.12: OPF classification time [ms] over the test set considering MOBSO using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
37.4436	37.1843	5.3487	5.9227	238.5865	247.0244	2210.2719	2340.3538	947.7959	934.8670	2.4987	**2.4817**	50.6011	53.2433	**4.2513**	4.8580
37.6668	39.5510	5.2971	5.8937	**232.1547**	236.8789	2199.4560	2384.8294	945.1421	944.3814	2.5250	2.5223	50.9686	51.9140	4.2962	4.3810
37.6939	38.6387	5.4098	5.5706	234.4959	235.3043	2213.8505	2349.1429	938.3102	977.7015	2.5133	2.5346	50.3033	54.2999	4.2642	4.3530
38.5054	**36.6354**	5.4335	5.3979	231.1422	235.7493	2206.8422	2205.6344	987.8328	934.0697	2.5491	2.5151	**50.0868**	53.6708	4.2611	4.6338
37.4855	36.8153	5.4831	5.3897	232.7688	263.4341	2206.9115	2206.4279	942.1043	1060.3155	2.5710	2.5255	50.4176	50.4803	4.2569	4.7732
41.4105	38.3217	5.9092	5.7002	261.0667	242.2472	2209.4785	2210.2561	1003.7142	987.9454	2.7209	2.5337	56.3376	54.3662	4.4976	4.7543
37.3175	37.5140	5.8855	5.6506	262.3070	280.1310	2299.0017	2271.7945	1003.6290	1004.4524	2.7171	2.6048	54.4086	53.7047	4.4803	4.4293
39.8536	39.4400	5.9057	5.5780	244.8401	256.6322	2227.7804	2333.2611	950.1321	983.8173	2.7149	2.6547	54.7768	50.0529	4.3966	4.6182
37.1776	38.8507	5.8117	5.7133	244.3636	263.7904	2216.9227	2200.9783	984.4288	1010.2878	2.6654	2.5198	58.4994	53.7005	4.4172	4.3003
39.9638	37.1884	5.7322	5.4425	249.0203	271.5257	2201.8715	2254.3559	983.1052	999.4154	2.7465	2.6916	55.0096	51.4499	4.5848	4.4458
38.6065	38.5823	5.8473	5.3803	249.7323	254.8377	2247.5689	2327.1738	1057.1158	973.6856	2.7348	2.7049	55.4471	53.2821	4.3327	4.3087
39.0017	38.2520	5.8141	5.5020	247.8265	247.6033	**2198.8997**	2222.7446	934.4760	938.7620	2.7747	2.5015	55.3679	51.9893	4.8108	4.7565
37.0897	38.1658	5.7948	5.3709	248.8797	247.2195	2223.2496	2219.1685	934.6551	**930.9366**	2.7373	2.4963	55.5349	52.3685	4.5610	4.7370
39.0843	38.4441	5.7612	5.2795	252.6534	253.1957	2236.3485	2201.6687	938.3404	950.4469	2.7872	2.7085	55.7464	51.3100	4.4399	4.6690
38.9018	38.8150	5.8322	**5.2789**	242.9706	252.8292	2199.9188	2237.5824	933.1782	991.8629	2.6461	2.5548	55.8714	52.4966	4.9375	4.4772

Table 7.13: OPF classification time [ms] over the test set considering MOFPA using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
65.7947	61.1261	9.1780	8.8473	303.6099	289.6495	3775.7229	3971.0539	1466.5228	1569.9828	3.6727	3.6023	71.3449	71.9769	7.4410	7.3037
65.3047	61.6293	9.1167	8.8294	274.0305	290.9857	3773.2523	3857.8329	1467.4790	1479.9546	3.8452	**3.5609**	72.0789	72.1102	7.1770	7.1911
62.8358	61.2264	9.2044	9.7709	289.9417	301.6426	3782.4157	3781.4566	1534.0830	1454.2672	3.7185	3.7101	72.2523	70.9991	7.0798	7.1483
62.6570	60.9854	9.5727	8.7635	274.3998	275.3907	3764.5362	3773.3393	1489.5714	1632.1236	3.8630	3.7429	72.7717	72.2809	7.3440	7.1340
63.2830	61.6558	**8.3437**	9.8182	275.2336	274.1147	3841.4547	3760.4018	1510.0908	1446.4789	4.5745	3.6097	72.5370	**70.9497**	7.2031	7.2016
63.9723	60.9972	8.7783	8.8314	274.6536	273.9276	3780.4305	**3734.8424**	1534.0182	1446.4557	4.6223	3.6738	71.5267	71.3203	8.5571	7.0866
63.0067	61.7220	8.7477	8.8126	274.3322	305.3923	3787.6222	3758.9917	1514.4405	**1435.2239**	3.5877	3.6424	71.6620	71.6068	7.4267	7.0902
63.4399	61.6076	8.9113	9.5694	277.7789	275.8438	3796.8088	3741.1212	1489.0628	1458.2429	3.9263	3.7125	71.3808	72.0681	7.2509	7.1364
63.8668	61.1369	8.7102	9.4898	**270.1087**	273.3940	3798.3656	3799.0189	1455.1715	1444.9687	3.8740	3.6144	71.3232	71.1053	7.2325	7.1672
63.2419	61.0232	8.4665	8.8502	290.6199	276.6193	3801.4951	3790.5249	1457.1823	1440.4387	3.9172	3.6445	72.5876	71.8381	7.3168	**7.0280**
62.8268	60.5751	8.6267	9.3065	277.0646	276.4188	3803.1316	3789.2029	1457.0262	1451.6100	4.1304	3.7677	72.3342	71.1393	7.2616	7.0971
63.1409	61.0309	9.2845	8.6067	289.2688	273.3742	3861.6638	3838.4658	1470.8612	1464.9577	3.8068	3.8005	72.0629	71.4955	8.0016	7.1848
63.4634	**60.3784**	9.2421	9.6762	297.8688	275.9115	4020.4896	3765.2944	1461.3658	1447.1208	3.8312	3.8028	72.1101	71.3798	7.3535	7.1232
63.0574	61.4548	8.9452	8.6745	296.4165	276.7313	3775.8108	3763.7397	1450.2027	1443.5178	4.0003	3.9880	71.7332	72.3483	7.1480	7.1054
62.7453	61.8400	9.1438	9.4501	297.7156	277.2377	3855.0325	3755.8259	1460.9727	1453.7772	4.1253	3.6586	71.9537	72.3866	7.2059	7.1864

considering not only ABO, but also the other optimization techniques.

We can also highlight the gain in computational burden, where the proposed multi-objective approaches obtained better classification times than their respective versions, except for Pendigits dataset where FA overcame the MO-I and MO-II approaches.

Table 7.14: OPF classification time [ms] over the test set considering MOPSO using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
43.1008	41.5454	6.6941	6.6875	253.1043	247.8948	2687.6494	2695.1900	1070.3312	1106.9398	2.9407	3.0197	55.1498	56.0153	4.4256	4.4819
44.2937	42.1038	6.5561	6.5536	240.7723	242.9846	2568.3664	2680.4177	**1060.9201**	1064.4414	2.7893	2.7208	**54.7003**	55.8543	4.4350	4.4059
43.1313	42.2662	6.6660	6.4127	254.0697	241.2418	2572.7188	2730.5557	1087.4998	1079.4457	3.0224	2.8618	55.4225	56.1949	4.3971	4.4742
45.0615	42.2555	6.8277	**6.0505**	252.9682	237.5725	2576.6152	2704.5674	1070.1688	1133.4602	2.8158	**2.6829**	54.9762	55.8512	4.3764	4.4323
44.6118	41.6210	6.7682	6.7373	245.1446	**236.4427**	2558.7721	2734.9661	1091.7809	1153.3184	2.9500	2.8457	55.3140	55.4488	4.3405	4.4121
41.9591	41.8855	6.8804	6.7574	248.9938	237.2801	**2556.5096**	2873.7741	1109.0350	1138.3366	2.8654	2.7104	55.2025	56.1846	4.4096	4.4956
43.8881	41.5817	6.5012	6.3095	248.3832	242.3791	2562.9764	2857.5073	1075.0829	1091.5498	2.8246	2.6872	55.5342	55.7619	4.4216	4.4485
43.1556	41.5341	6.5937	6.3907	250.5298	241.6811	2602.1924	2758.8152	1165.1149	1083.4569	2.8753	2.7150	57.0716	55.5632	4.3700	4.3653
43.5439	41.8411	6.7581	6.3608	247.1172	238.6491	2786.5380	2582.6152	1160.1356	1084.3663	2.9010	2.7408	58.3450	55.8457	4.3963	4.4444
44.0741	42.0890	6.4708	6.3123	245.4849	240.2731	2725.2623	2570.9992	1092.0341	1082.2739	3.0015	3.0218	59.3706	56.4584	4.4129	4.4709
43.1574	**41.2618**	6.5292	6.1787	243.5540	240.9628	2623.2456	2584.7891	1098.2877	1085.4555	2.8608	2.7571	60.6200	55.6961	4.3434	4.4903
43.0059	41.8988	6.5056	6.0932	242.8412	240.5312	2594.2054	2569.7231	1089.6770	1117.2405	2.9060	2.9799	55.7556	55.6417	**4.3305**	4.4831
45.1643	41.6726	6.4208	6.1558	244.0754	238.7013	2669.7487	2569.9771	1089.6311	1101.3761	2.8873	2.9020	57.4034	56.1763	4.3648	4.4747
44.3816	41.4407	6.3584	6.1400	243.3132	239.4148	2580.7352	2565.2169	1105.3328	1088.3689	2.7450	2.6934	58.5355	55.7027	4.3608	4.4497
45.2889	41.6118	6.4028	6.2200	241.9221	241.3318	2575.2129	2658.8625	1149.7981	1081.0367	2.8243	2.7555	58.0263	55.6312	4.3544	4.4633

Table 7.15: OPF classification time [ms] over the test set considering MOFA using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
43.1817	44.5211	6.1777	6.4252	238.2406	252.1369	2624.5542	2528.4041	1041.1320	1163.5736	2.9952	2.7370	56.9876	59.1159	4.4611	4.8050
43.3810	42.3796	6.6032	6.2917	239.4010	255.4336	2592.9416	2532.3125	1039.1602	1055.5246	2.9542	2.7218	56.7942	61.8617	**4.4130**	4.7098
43.0340	41.3461	6.3279	6.4221	236.6228	264.1298	2522.3116	2526.4875	**1038.6646**	1044.9894	3.0307	**2.6796**	56.5298	61.3813	4.6329	4.6781
42.6865	41.8684	6.1019	6.2739	253.0374	269.9845	2527.7199	2615.8014	1048.2244	1047.6126	3.0076	2.7141	58.0889	62.4442	4.4245	4.7333
44.1988	42.3928	6.2436	**5.9860**	236.3712	244.3148	2531.6624	2704.3702	1101.3979	1040.9944	2.9660	2.9749	58.3262	56.9522	4.5431	4.8646
43.6889	**41.1775**	6.2225	6.2609	235.6218	238.7955	**2518.4205**	2652.2721	1068.9500	1045.3951	2.9777	3.0512	57.7545	58.7954	4.9403	4.7265
44.3378	42.3418	6.6045	6.3053	237.2773	252.1676	2525.7897	2584.8395	1051.3994	1044.4517	2.9390	2.7306	58.4448	54.6222	4.9072	4.7303
42.2208	42.3152	6.0694	6.3923	237.0011	239.3370	2524.1423	2723.9832	1056.1405	1047.4339	2.9803	2.8191	60.6923	54.1080	4.7407	4.6893
43.4751	47.4902	6.1026	6.4030	235.8586	260.3654	2527.6763	2575.2451	1066.4183	1087.2967	2.9295	2.7618	57.1188	54.2867	4.7446	5.0110
42.3835	45.4325	6.1752	6.6182	**233.5053**	257.8851	2726.2533	2546.2649	1110.3662	1049.4473	2.9660	2.7992	58.4260	54.5827	4.8154	4.9144
44.1114	45.3888	6.4491	6.1709	235.3105	260.3772	2529.5839	2550.6757	1174.7308	1044.4681	2.9560	2.8531	57.2868	54.8048	4.8346	4.6657
44.6251	46.0183	6.3026	6.4209	238.0078	260.8165	2519.3023	2548.2841	1113.7072	1042.7411	2.8966	2.8173	56.6994	54.4126	4.8968	4.5453
43.3657	44.5621	6.6032	6.3336	240.6807	258.5509	2528.8349	2570.3094	1060.3694	1044.3581	2.9049	3.0505	55.9386	54.1522	4.4710	4.8434
43.2035	44.7974	6.6549	7.0248	249.5390	255.0947	2521.3660	2571.4113	1060.3979	1046.8536	2.9311	2.8724	61.7011	54.3142	4.4778	4.7891
42.4509	44.6721	6.0434	6.6159	257.1939	245.4270	2540.0113	2577.1522	1057.4671	1050.1314	2.8820	2.7962	58.3008	**53.9834**	4.4595	4.7478

Table 7.16: OPF classification time [ms] over the test set considering MOABO using both methods.

D1		D2		D3		D4		D5		D6		D7		D8	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
40.7372	46.4140	6.5276	6.8306	243.9459	240.6160	2672.7928	2557.7211	1055.1687	1099.7347	3.0486	2.8063	53.8294	63.1823	4.4132	5.0338
40.3999	48.0638	6.5375	6.6341	245.5202	265.2359	2563.1109	2585.8427	1070.0457	1094.5302	2.8764	2.9622	**53.8060**	63.7138	4.4005	4.9431
40.6269	47.4357	6.2100	6.7017	245.9433	261.2216	2565.4186	2572.9847	1082.3806	1094.1100	2.9601	2.9060	54.9457	63.3305	4.3597	5.0363
40.8606	45.0828	6.1580	6.8156	247.7194	260.2824	2532.2921	2558.6317	**1051.8521**	1062.6643	2.9987	2.8568	54.6922	62.5069	4.4510	4.9833
40.2118	42.0907	6.0697	6.7175	244.1143	265.4362	2534.7349	2582.0807	1068.1780	1118.1347	2.9858	2.8083	54.4698	63.0499	4.3411	4.9799
40.5327	41.4456	6.1275	6.7165	243.1011	254.5836	**2526.2764**	2585.4187	1117.6240	1074.6288	2.9885	2.8267	54.0664	63.7024	4.3348	4.9051
40.9643	41.6122	**6.0065**	6.7948	237.0757	263.4672	2542.6458	2562.6190	1148.3540	1087.9209	2.7927	2.7546	54.2162	62.6687	**4.3238**	4.9342
40.7825	43.1946	6.0712	6.7033	235.9438	268.7633	2585.4452	2719.7555	1146.5517	1080.8486	2.7666	2.9672	54.4026	62.9117	4.3394	5.0131
40.7226	41.7344	6.1182	6.7295	235.5438	263.3332	2661.0683	2695.6028	1105.2047	1084.0722	2.7631	3.0044	53.9624	63.6939	4.4238	4.9586
40.7861	42.0674	6.1095	6.8234	232.6092	265.7857	2637.4384	2581.6904	1143.1002	1084.0208	2.7627	3.0275	54.5170	59.7343	4.3541	4.9149
41.4745	45.4255	6.1044	6.7923	238.7283	264.8787	2801.5214	2565.6339	1133.5473	1095.5595	2.7665	3.1144	54.7246	55.6820	4.3714	5.0084
40.6973	42.5494	6.0190	6.8165	236.7566	266.3220	2779.3517	2591.0062	1106.6383	1092.8316	2.8132	2.9606	54.7325	55.9784	4.3535	4.9743
40.6734	47.1110	6.0815	6.6273	234.7530	247.2989	2729.0272	2699.4595	1127.2353	1088.6165	2.9030	2.9969	54.9390	55.2044	4.3777	4.9849
40.9900	47.5641	6.3427	6.5245	235.5885	236.1931	2757.7060	2689.8728	1074.3758	1088.2703	**2.7345**	2.9803	54.5582	55.4309	4.3850	4.9366
40.4070	47.0076	6.0608	6.6283	**229.8755**	267.4954	2635.8116	2644.2503	1139.8569	1097.6119	2.7870	2.9904	54.3467	55.1849	4.3481	4.9017

7.6 Conclusions

In this work, we proposed the Binary Artificial Butterfly Algorithm and evaluated it for the task of feature selection concerning single, multi- and many-objective optimization. Also, two different approaches for feature selection were proposed in this work: (i) the first one (MO-I) concerns optimizing only the accuracy of each class separately, while (ii) the second approach (MO-II) optimizes the accuracy of each class along with the total number of features.

The experimental results demonstrated the single-objective binary version of ABO obtained good results and selected fewer features in five out of eight datasets, being the fastest technique when compared to FPA, PSO, and FA, making it a good option for feature selection purposes. Under the multi- and many-objective optimization, we evaluated the robustness of MO-I and MO-II, where both approaches obtained balanced results between them. We have also observed that MOABO achieved good results in three out of eight datasets followed by MOPSO and MOFA. Regarding the classification time, we noted that MO-I obtained the best results, being MOABO and MOFA the fastest techniques. Finally, we compared both MO-I and MO-II approaches against their single-objective versions. We observed that both approaches outperformed their single-objective variants in terms of accuracy, number of selected features, and classification time.

In regard to future works, we intend to devise a many-objective optimization model by adding more fitness functions, such as precision, recall, and F1-measure. We also intend to evaluate these approaches to real-world datasets with a larger number of classes and features.

7.A Preliminary Results

Tables 7.17, 7.18, 7.19, 7.20, 7.21, 7.22 present the results regarding the proposed MO-I and MO-II for each meta-heuristic technique, i.e., MOBHA, MOBSO, MOFPA, MOPSO, MOFA, and MOABO, respectively. In each column of the table, we have the OPF accuracy and the number of selected features concerning MO-I and MO-II approaches for each dataset.

Table 7.17: Trade-off between the classification accuracy over the test set and the total number of selected features concerning MOBHA.

German Numer		Ionosphere		MPEG-7		Pendigits		Satimage		Sonar		Splice		SVM-Guilde 2	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.500/14	0.513/17	0.731/27	0.808/26	0.906/100	0.911/121	0.981/8	0.994/14	0.921/25	0.928/25	0.863/39	0.782/32	0.722/37	0.681/33	0.686/9	0.621/6
0.580/13	0.531/12	0.736/27	0.874/24	0.920/126	0.911/111	0.922/6	0.982/9	0.926/23	0.928/16	0.788/34	0.735/45	0.718/49	0.670/41	0.651/11	0.705/10
0.590/13	0.537/17	0.854/25	0.862/26	0.926/113	0.929/123	0.988/9	0.922/6	0.911/22	0.924/26	0.745/45	0.816/46	0.731/37	0.641/35	0.685/8	0.592/10
0.579/14	0.494/11	0.797/23	0.882/21	0.924/121	0.922/103	0.993/12	0.932/6	0.909/23	0.918/21	0.860/39	0.825/39	0.679/46	0.663/47	0.860/11	0.705/10
0.532/20	0.550/17	0.854/24	0.788/22	0.909/126	0.913/113	0.982/10	0.989/13	0.915/22	0.923/21	0.860/45	0.760/37	0.714/37	0.672/37	0.701/11	0.709/13
0.611/13	0.545/14	0.739/23	0.904/25	0.926/111	0.917/109	0.948/6	0.976/8	0.922/23	0.921/21	0.810/50	0.878/45	0.720/39	0.687/39	0.637/9	0.692/9
0.598/14	0.593/12	0.805/20	0.874/27	0.908/130	0.918/110	0.945/6	0.979/7	0.918/26	0.922/21	0.828/49	0.673/42	0.707/37	0.707/41	0.766/15	0.739/14
0.533/16	0.554/17	0.808/23	0.788/25	0.920/105	0.908/127	0.965/7	0.847/4	0.922/20	0.926/25	0.782/39	0.763/43	0.677/38	0.699/38	0.719/15	0.650/10
0.471/9	0.586/12	0.785/25	0.832/23	0.926/113	0.899/119	0.963/7	0.964/7	0.936/23	0.932/19	0.766/45	0.788/32	0.621/44	0.705/37	0.630/11	0.648/10
0.545/12	0.523/18	0.862/27	0.709/27	0.908/129	0.926/122	0.893/5	0.914/5	0.922/26	0.926/21	0.738/41	0.860/36	0.687/44	0.716/38	0.672/10	0.601/13
0.606/12	0.621/17	0.766/25	0.854/26	0.929/121	0.926/116	0.972/7	0.981/8	0.929/24	0.926/26	0.760/32	0.863/40	0.711/31	0.644/42	0.632/11	0.737/13
0.531/12	0.581/11	0.851/26	0.901/26	0.911/117	0.891/117	0.981/11	0.985/10	0.931/24	0.927/22	0.878/39	0.735/41	0.781/36	0.655/39	0.610/12	0.641/11
0.493/19	0.592/15	0.777/28	0.865/21	0.904/113	0.908/114	0.973/7	0.927/6	0.930/24	0.908/23	0.832/35	0.828/39	0.658/49	0.568/41	0.665/10	0.694/6
0.543/12	0.563/14	0.865/24	0.788/21	0.906/116	0.897/106	0.965/7	0.945/6	0.919/27	0.910/17	0.835/38	0.763/27	0.687/40	0.712/41	0.747/10	0.701/10
0.611/11	0.576/11	0.893/26	0.805/25	0.906/100	0.913/120	0.864/4	0.935/6	0.927/29	0.922/23	0.860/33	0.807/42	0.692/35	0.706/36	0.669/10	0.697/11

Table 7.18: Trade-off between the classification accuracy over the test set and the total number of selected features concerning MOBSO.

German Numer		Ionosphere		MPEG-7		Pendigits		Satimage		Sonar		Splice		SVM-Guilde 2	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.583/15	0.533/17	0.824/20	0.865/21	0.926/114	0.926/121	0.983/8	0.907/5	0.927/25	0.908/16	0.757/39	0.816/35	0.707/36	0.710/43	0.598/12	0.666/11
0.520/19	0.511/17	0.835/20	0.827/26	0.918/101	0.911/117	0.996/14	0.993/14	0.920/22	0.917/24	0.813/45	0.807/35	0.689/42	0.713/39	0.734/10	0.627/7
0.519/15	0.532/13	0.805/21	0.835/21	0.891/122	0.913/117	0.993/12	0.996/15	0.921/22	0.926/25	0.813/35	0.928/42	0.677/42	0.708/34	0.776/12	0.707/14
0.599/14	0.607/14	0.813/21	0.824/27	0.909/133	0.922/125	0.980/10	0.942/7	0.914/23	0.922/21	0.885/34	0.857/38	0.692/44	0.629/40	0.688/12	0.785/15
0.571/12	0.577/16	0.846/20	0.766/28	0.928/124	0.908/121	0.994/14	0.980/9	0.932/22	0.915/26	0.729/41	0.763/40	0.653/34	0.676/42	0.688/12	0.754/10
0.567/18	0.640/16	0.854/24	0.854/22	0.902/115	0.917/113	0.992/11	0.974/8	0.924/19	0.914/23	0.885/48	0.850/41	0.696/39	0.668/50	0.631/9	0.680/14
0.601/14	0.608/18	0.813/22	0.846/17	0.915/110	0.920/117	0.959/7	0.953/5	0.916/23	0.940/23	0.770/51	0.900/37	0.673/42	0.726/41	0.616/11	0.698/14
0.588/16	0.612/20	0.810/25	0.821/25	0.909/112	0.908/116	0.941/6	0.903/6	0.922/27	0.933/22	0.757/41	0.885/46	0.762/35	0.665/36	0.729/13	0.700/10
0.589/21	0.560/10	0.851/22	0.862/23	0.926/102	0.906/117	0.946/7	0.993/12	0.925/19	0.937/26	0.835/39	0.810/41	0.677/40	0.656/34	0.674/11	0.749/13
0.590/17	0.532/12	0.785/20	0.835/23	0.920/126	0.906/122	0.974/8	0.969/8	0.930/22	0.939/26	0.903/49	0.838/39	0.687/37	0.682/39	0.749/15	0.676/11
0.590/16	0.577/14	0.808/27	0.758/23	0.900/124	0.917/119	0.987/11	0.942/7	0.919/24	0.916/24	0.932/43	0.807/42	0.721/36	0.685/41	0.662/10	0.752/14
0.529/15	0.540/14	0.851/23	0.720/23	0.922/112	0.917/111	0.990/9	0.986/10	0.911/24	0.926/23	0.900/41	0.882/50	0.639/39	0.681/38	0.690/11	0.702/13
0.575/16	0.523/14	0.816/23	0.865/25	0.915/116	0.920/110	0.985/8	0.992/11	0.928/24	0.918/21	0.813/43	0.882/41	0.659/41	0.725/37	0.663/13	0.719/12
0.556/17	0.606/16	0.865/22	0.766/23	0.926/132	0.926/102	0.977/9	0.994/14	0.923/28	0.913/16	0.803/42	0.835/41	0.731/36	0.653/39	0.716/8	0.782/14
0.556/17	0.486/15	0.835/25	0.785/23	0.911/107	0.920/122	0.994/12	0.983/9	0.907/25	0.927/24	0.828/43	0.820/36	0.683/33	0.705/42	0.705/12	0.623/8

Table 7.19: Trade-off between the classification accuracy over the test set and the total number of selected features concerning MOFPA.

German Numer		Ionosphere		MPEG-7		Pendigits		Satimage		Sonar		Splice		SVM-Guilde 2	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.575/19	0.519/15	0.736/22	0.824/18	0.920/125	0.915/120	0.942/05	0.897/06	0.915/22	0.921/22	0.900/34	0.850/37	0.716/37	0.758/43	0.700/10	0.693/10
0.542/16	0.595/11	0.851/19	0.835/22	0.922/117	0.899/114	0.951/06	0.971/07	0.919/20	0.925/22	0.810/32	0.875/44	0.717/37	0.661/44	0.697/14	0.702/12
0.508/14	0.519/16	0.808/20	0.865/21	0.922/119	0.902/125	0.986/10	0.845/05	0.923/22	0.924/18	0.716/28	0.832/42	0.727/42	0.716/44	0.749/13	0.677/14
0.507/10	0.587/18	0.808/27	0.893/23	0.931/121	0.900/116	0.994/11	0.969/07	0.931/24	0.929/21	0.791/38	0.757/43	0.648/34	0.642/35	0.656/15	0.716/12
0.567/15	0.568/13	0.827/17	0.824/24	0.908/111	0.924/101	0.929/06	0.963/06	0.922/25	0.918/25	0.853/43	0.760/29	0.711/33	0.687/34	0.727/12	0.678/11
0.599/17	0.499/13	0.728/22	0.854/23	0.911/122	0.926/114	0.993/11	0.934/06	0.918/23	0.918/19	0.785/39	0.773/35	0.688/34	0.708/42	0.713/10	0.638/12
0.546/16	0.596/14	0.797/21	0.862/25	0.928/116	0.919/104	0.991/11	0.991/11	0.933/27	0.935/24	0.788/44	0.785/43	0.672/41	0.671/42	0.683/13	0.695/16
0.555/17	0.588/17	0.835/23	0.758/21	0.929/110	0.911/113	0.890/04	0.938/06	0.943/22	0.923/26	0.935/38	0.785/42	0.524/41	0.715/35	0.700/13	0.631/09
0.585/13	0.569/17	0.862/21	0.789/23	0.913/114	0.911/120	0.920/05	0.825/04	0.913/14	0.933/21	0.766/45	0.910/37	0.688/46	0.687/34	0.655/14	0.720/11
0.635/14	0.652/16	0.808/23	0.835/20	0.911/102	0.899/119	0.977/09	0.920/06	0.909/18	0.926/24	0.810/28	0.791/46	0.735/42	0.730/46	0.665/13	0.731/12
0.638/15	0.621/17	0.832/24	0.901/22	0.915/115	0.917/129	0.893/05	0.908/05	0.923/22	0.926/31	0.878/36	0.841/38	0.703/38	0.658/41	0.675/12	0.625/11
0.535/16	0.489/20	0.805/19	0.816/19	0.922/135	0.920/124	0.945/05	0.804/03	0.925/31	0.917/23	0.788/38	0.760/38	0.659/39	0.732/39	0.640/15	0.687/15
0.514/15	0.555/14	0.808/20	0.912/20	0.911/110	0.909/106	0.982/09	0.991/14	0.920/18	0.925/27	0.832/36	0.828/41	0.745/39	0.686/41	0.669/14	0.766/13
0.550/19	0.544/16	0.797/23	0.747/21	0.891/126	0.900/121	0.984/08	0.952/06	0.921/21	0.916/20	0.810/47	0.766/31	0.641/41	0.604/35	0.718/13	0.656/10
0.625/18	0.542/15	0.890/29	0.835/25	0.920/120	0.893/107	0.989/09	0.984/10	0.918/20	0.939/25	0.857/43	0.878/40	0.731/36	0.658/38	0.603/08	0.745/09

Table 7.20: Trade-off between the OPF classification accuracy over the test set and the total number of selected features concerning MOPSO.

German Numer		Ionosphere		MPEG-7		Pendigits		Satimage		Sonar		Splice		SVM-Guilde 2	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.635/15	0.549/14	0.777/26	0.827/21	0.928/123	0.924/120	0.911/05	0.983/10	0.927/25	0.936/24	0.791/40	0.803/40	0.679/38	0.722/40	0.704/15	0.733/13
0.561/13	0.561/12	0.885/24	0.827/29	0.911/118	0.908/115	0.957/07	0.958/07	0.924/27	0.929/26	0.825/38	0.698/45	0.668/39	0.734/36	0.657/12	0.695/11
0.531/09	0.579/13	0.901/19	0.846/27	0.931/116	0.893/108	0.971/08	0.991/11	0.922/18	0.924/27	0.791/41	0.853/42	0.700/41	0.665/42	0.584/10	0.736/14
0.525/12	0.638/17	0.789/25	0.854/23	0.920/130	0.940/116	0.949/06	0.958/07	0.930/29	0.921/19	0.888/43	0.853/41	0.636/41	0.571/40	0.731/14	0.748/13
0.630/16	0.569/17	0.805/24	0.777/25	0.922/125	0.922/114	0.993/12	0.964/07	0.924/24	0.919/24	0.832/39	0.882/47	0.668/44	0.708/36	0.613/10	0.734/12
0.546/13	0.512/16	0.885/25	0.882/21	0.919/115	0.920/123	0.900/05	0.992/09	0.931/22	0.910/23	0.760/45	0.900/39	0.643/36	0.654/46	0.689/12	0.716/11
0.570/10	0.564/15	0.794/24	0.758/22	0.919/123	0.920/120	0.943/06	0.968/06	0.928/31	0.913/18	0.885/43	0.732/45	0.711/32	0.693/41	0.580/10	0.757/15
0.560/14	0.496/10	0.797/23	0.816/24	0.900/119	0.917/115	0.995/14	0.992/11	0.921/22	0.932/24	0.875/48	0.806/45	0.660/36	0.681/41	0.695/10	0.737/10
0.599/14	0.486/13	0.854/23	0.854/25	0.909/106	0.919/113	0.987/10	0.956/06	0.926/27	0.920/25	0.900/42	0.860/45	0.678/44	0.732/36	0.648/12	0.639/12
0.587/15	0.552/17	0.824/22	0.846/24	0.917/106	0.911/128	0.928/06	0.986/10	0.927/24	0.921/18	0.741/34	0.807/40	0.722/40	0.706/33	0.664/06	0.746/14
0.511/16	0.606/12	0.739/24	0.805/26	0.928/125	0.909/120	0.916/05	0.993/12	0.928/26	0.921/19	0.691/39	0.853/41	0.699/39	0.746/38	0.727/13	0.645/09
0.567/13	0.606/14	0.862/27	0.805/25	0.911/123	0.917/123	0.993/14	0.949/06	0.917/19	0.913/21	0.760/41	0.782/37	0.701/42	0.637/33	0.627/09	0.726/12
0.513/16	0.598/16	0.835/27	0.827/23	0.899/135	0.913/107	0.991/11	0.953/07	0.920/19	0.933/26	0.906/37	0.803/46	0.688/42	0.755/42	0.758/16	0.720/12
0.595/15	0.486/13	0.854/22	0.824/26	0.933/133	0.913/112	0.998/15	0.991/12	0.923/21	0.920/29	0.885/37	0.785/44	0.731/33	0.705/29	0.696/11	0.787/13
0.554/15	0.582/15	0.865/18	0.865/24	0.933/113	0.917/127	0.997/13	0.948/06	0.904/15	0.923/25	0.757/39	0.950/36	0.677/40	0.652/33	0.772/12	0.733/14

Table 7.21: Trade-off between the OPF classification accuracy over the test set and the total number of selected features concerning MOFA.

German Numer		Ionosphere		MPEG-7		Pendigits		Satimage		Sonar		Splice		SVM-Guilde 2	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.606/18	0.543/15	0.904/22	0.923/19	0.906/112	0.906/116	0.981/8	0.956/07	0.943/24	0.930/21	0.913/35	0.853/39	0.670/36	0.717/45	0.694/14	0.634/07
0.577/12	0.583/17	0.854/17	0.797/21	0.920/115	0.904/112	0.950/6	0.943/07	0.929/25	0.922/23	0.835/45	0.903/46	0.741/46	0.756/41	0.616/09	0.738/13
0.602/18	0.474/11	0.805/28	0.816/26	0.915/106	0.920/105	0.903/5	0.971/08	0.919/19	0.946/28	0.741/44	0.853/45	0.679/40	0.684/42	0.698/13	0.673/13
0.517/12	0.581/14	0.835/27	0.777/28	0.915/109	0.919/114	0.942/7	0.902/06	0.911/23	0.929/20	0.878/38	0.782/38	0.672/32	0.637/40	0.771/13	0.707/10
0.592/20	0.538/11	0.808/25	0.797/25	0.920/116	0.909/113	0.850/4	0.992/11	0.927/23	0.920/25	0.853/41	0.860/42	0.755/38	0.691/39	0.726/11	0.694/11
0.577/11	0.610/17	0.808/25	0.816/24	0.913/116	0.913/121	0.970/8	0.983/09	0.914/20	0.920/27	0.857/45	0.863/40	0.688/30	0.663/37	0.582/10	0.656/13
0.618/12	0.605/17	0.904/26	0.827/23	0.920/107	0.917/111	0.870/4	0.978/07	0.920/24	0.921/26	0.807/34	0.832/45	0.681/36	0.715/44	0.725/15	0.650/12
0.507/09	0.605/11	0.789/22	0.789/28	0.911/126	0.919/136	0.927/7	0.944/05	0.909/16	0.923/26	0.785/41	0.828/44	0.648/41	0.732/42	0.617/12	0.667/12
0.621/18	0.539/12	0.865/24	0.874/27	0.928/109	0.922/109	0.894/5	0.943/06	0.919/26	0.920/25	0.860/37	0.907/30	0.609/42	0.733/39	0.620/08	0.656/13
0.646/20	0.629/16	0.835/23	0.805/28	0.909/094	0.909/110	0.938/6	0.949/07	0.911/25	0.913/20	0.810/41	0.860/38	0.729/38	0.637/42	0.688/13	0.767/12
0.523/13	0.604/15	0.874/27	0.865/25	0.906/131	0.926/110	0.947/7	0.935/05	0.931/26	0.922/30	0.813/45	0.803/46	0.686/43	0.696/41	0.712/10	0.653/09
0.556/13	0.539/16	0.862/21	0.865/23	0.915/116	0.915/109	0.923/5	0.917/05	0.922/22	0.924/23	0.903/40	0.863/31	0.745/37	0.672/38	0.741/15	0.668/14
0.563/14	0.601/14	0.854/19	0.835/25	0.919/104	0.911/117	0.934/5	0.970/07	0.926/25	0.920/22	0.835/48	0.853/44	0.706/36	0.725/33	0.642/10	0.671/07
0.595/20	0.610/12	0.758/21	0.862/25	0.908/119	0.913/118	0.954/7	0.941/07	0.921/28	0.932/24	0.763/46	0.763/42	0.686/37	0.618/43	0.637/08	0.649/13
0.541/18	0.518/13	0.747/27	0.854/26	0.886/107	0.913/106	0.954/7	0.995/14	0.922/25	0.916/19	0.928/38	0.882/42	0.691/39	0.700/37	0.755/11	0.601/09

Table 7.22: Trade-off between the OPF classification accuracy over the test set and the total number of selected features concerning MOABO.

German Numer		Ionosphere		MPEG-7		Pendigits		Satimage		Sonar		Splice		SVM-Guilde 2	
MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II	MO-I	MO-II
0.616/19	0.566/17	0.794/22	0.739/23	0.928/116	0.926/118	0.977/09	0.893/06	0.929/22	0.916/26	0.788/43	0.857/44	0.741/45	0.632/39	0.730/14	0.703/09
0.612/10	0.566/14	0.843/25	0.874/23	0.926/114	0.917/116	0.992/12	0.937/06	0.913/22	0.917/20	0.845/45	0.882/39	0.687/44	0.673/50	0.752/14	0.712/11
0.583/16	0.554/18	0.766/29	0.854/19	0.915/125	0.920/111	0.975/07	0.957/08	0.919/23	0.926/19	0.785/38	0.863/37	0.639/44	0.673/36	0.681/10	0.724/08
0.567/16	0.581/17	0.744/21	0.816/20	0.917/109	0.928/119	0.985/11	0.924/05	0.932/22	0.910/21	0.788/38	0.863/28	0.710/45	0.660/41	0.711/12	0.726/14
0.578/17	0.588/10	0.777/26	0.912/20	0.915/120	0.909/111	0.991/10	0.974/08	0.925/19	0.930/24	0.928/44	0.832/33	0.686/35	0.636/41	0.755/09	0.764/10
0.580/16	0.457/17	0.739/23	0.835/26	0.929/123	0.915/122	0.985/09	0.953/07	0.931/21	0.915/32	0.813/40	0.832/43	0.736/38	0.673/44	0.710/08	0.649/10
0.582/16	0.591/16	0.777/26	0.884/23	0.911/104	0.935/120	0.995/13	0.992/13	0.928/24	0.931/24	0.766/40	0.785/44	0.745/42	0.677/36	0.694/15	0.784/11
0.577/13	0.643/18	0.874/21	0.777/26	0.913/108	0.909/120	0.989/12	0.993/13	0.921/23	0.925/24	0.832/37	0.788/43	0.701/37	0.672/38	0.819/15	0.671/14
0.536/19	0.538/13	0.962/21	0.769/24	0.900/125	0.929/109	0.993/11	0.993/12	0.928/26	0.921/14	0.832/35	0.857/43	0.635/33	0.662/35	0.656/12	0.718/13
0.563/14	0.629/15	0.797/26	0.843/17	0.920/118	0.902/124	0.990/10	0.989/12	0.932/25	0.929/26	0.882/40	0.835/44	0.630/39	0.736/33	0.686/10	0.736/12
0.558/18	0.537/19	0.697/15	0.827/24	0.911/126	0.917/121	0.975/08	0.971/09	0.918/25	0.926/23	0.791/37	0.803/42	0.627/37	0.623/38	0.650/08	0.723/17
0.558/14	0.646/19	0.827/25	0.797/23	0.922/103	0.928/108	0.989/12	0.989/10	0.913/21	0.920/24	0.770/38	0.882/39	0.698/38	0.575/37	0.759/13	0.707/14
0.561/12	0.560/16	0.835/24	0.843/26	0.924/123	0.909/118	0.979/08	0.993/13	0.926/23	0.924/19	0.807/37	0.832/35	0.716/34	0.623/40	0.702/13	0.770/12
0.495/12	0.589/13	0.846/21	0.747/24	0.904/115	0.915/108	0.985/11	0.987/12	0.929/25	0.929/22	0.782/47	0.835/42	0.688/45	0.668/49	0.754/15	0.651/15
0.562/19	0.544/14	0.832/21	0.816/21	0.929/110	0.935/120	0.892/05	0.967/08	0.928/27	0.931/28	0.882/40	0.860/46	0.682/43	0.686/39	0.699/14	0.696/10

Chapter 8

Conclusions

The present thesis focuses on meta-heuristic techniques for single, multi- and many-objective optimization applied to pattern recognition and computer vision areas. The number of works using meta-heuristic algorithms for solving single-objective problems is expressive. However, when it comes to optimizing more objective functions, the number of publications decreases. The study of algorithms involving optimization of many-objectives is still recent, and there is very little work, even in other areas of knowledge.

Single-objective meta-heuristic algorithms were employed for the task of feature selection where the objective function to be minimized was given by the OPF classifier error rate. Two studies were carried out, being: (i) characterization of irregular consumers using BHA, and (ii) channel selection in encephalogram examination to identify people in the biometric area using FPA. Also, a study in image reconstruction area was conducted where single-objective Cuckoo Search was employed to fine-tuning DBN's parameters.

Since many problems in pattern recognition and computer vision areas naturally have more than one objective function to be optimized, it is necessary to use multi- and many-objective optimization techniques. In this thesis, we proposed a multi-objective optimization pruning considering the OPF classifier where the idea is to obtain compact and representative training sets without the need for the desired loss and the maximum number of iteration parameters. Also, we proposed two multi- and many-objective feature selection approaches: (i) to minimize the classifier error for each class over the evaluating set, and (ii) to minimize the classifier error for each class over the evaluating set and also minimizing the number of selected features.

The results confirm the hypothesis of this work, evidencing that the use of single, multi- and many-objective meta-heuristic optimization algorithms improve the performance of machine learning techniques. However, such strategies still little explored in this area of knowledge.

Table 8.1 presents the works produced during the study period.

Name	**Type**	**Qualis**	**Year**	**Status**
On the Study of Commercial Losses in Brazil: A Binary Black Hole Algorithm for Theft Characterization (RAMOS et al., 2016)	Journal	A1	2016	Published
Unsupervised Non-Technical Losses Identification Through Optimum-Path Forest (JÚNIOR et al., 2016)	Journal	B3	2016	Published
EEG-based person identification through Binary Flower Pollination Algorithm (RODRIGUES et al., 2016)	Journal	A1	2016	Published
Social-Spider Optimization-based Support Vector Machines Applied for Energy Theft Detection (PEREIRA et al., 2016)	Journal	B1	2016	Published
Meta-heuristic Multi- and Many-objective Optimization Techniques for Solution of Machine Learning Problems (RODRIGUES; PAPA; ADELI, 2017)	Journal	-	2017	Published
A Multi-Objective Artificial Butterfly Optimization Approach for Class-Oriented Feature Selection	Journal	A1	2019	Submitted
Binary Flower Pollination Algorithm and Its Application to Feature Selection	Book	-	2015	Published
Fine-tuning deep belief networks using cuckoo search	Book	-	2016	Published
Fine-Tuning Restricted Boltzmann Machines using Quaternion-based Flower Pollination Algorithm	Book	-	2019	Submitted
Black Hole Algorithm for Non-Technical Losses Characterization (RODRIGUES et al., 2015)	Conference	B2	2015	Published
Pruning Optimum-Path Forest Classifiers Using Multi-Objective Optimization (RODRIGUES et al., 2015)	Conference	B1	2017	Published
Fine Tuning Deep Boltzmann Machines Through Meta-Heuristic Approaches (PASSOS; RODRIGUES; PAPA, 2018)	Conference	B1	2018	Published
Quaternion-Based Backtracking Search Optimization Algorithm	Conference	A1	2019	Published

Table 8.1: Works developed during the study period

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

ABBASS, H. A. Speeding up backpropagation using multiobjective evolutionary algorithms. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 15, n. 11, p. 2705–2726, 2003.

ABDULLAH, M. K. et al. Analysis of effective channel placement for an eeg-based biometric system. In: *Proc. of the IEEE Conference on Biomedical Engineering and Sciences*. [S.l.: s.n.], 2010. p. 303–306.

ABEDINPOURSHOTORBAN, H. et al. Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. *Swarm and Evolutionary Computation*, v. 26, p. 8–22, 2016.

ACKLEY, D.; HINTON, G.; SEJNOWSKI, T. J. A learning algorithm for boltzmann machines. In: WALTZ, D.; FELDMAN, J. (Ed.). *Connectionist Models and Their Implications: Readings from Cognitive Science*. Norwood, NJ, USA: Ablex Publishing Corp., 1988. p. 285–307. ISBN 0-89391-456-8.

AGUIRRE, H.; TANAKA, K. Adaptive ε-ranking on many-objective problems. *Evolutionary Intelligence*, v. 2, n. 4, p. 183, 2009.

AHMAD, S. R.; BAKAR, A. A.; YAAKUB, M. R. Metaheuristic algorithms for feature selection in sentiment analysis. In: *Science and Information Conference (SAI)*. [S.l.: s.n.], 2015. p. 222–226.

AKBARI, R. et al. A multi-objective artificial bee colony algorithm. *Swarm and Evolutionary Computation*, v. 2, p. 39–52, 2012.

ALBUKHANAJER, W. A.; BRIFFA, J. A.; JIN, Y. Evolutionary multiobjective image feature extraction in the presence of noise. *IEEE Transactions on Cybernetics*, PP, n. 99, p. 1–1, 2014.

ALBUKHANAJER, W. A. et al. Evolutionary multi-objective optimization of trace transform for invariant feature extraction. In: *IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2012. p. 1–8.

ALLèNE, C. et al. Some links between extremum spanning forests, watersheds and min-cuts. *Image Vision Computing*, Butterworth-Heinemann, Newton, MA, USA, v. 28, n. 10, p. 1460–1471, 2010.

ALVES, R. et al. Reduction of non-technical losses by modernization and updating of measurement systems. In: *Proceedings of the IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*. [S.l.: s.n.], 2006. p. 1–5.

ANEEL. *Irregular power consumption generates loss of* R*$ 8,1 billion per year*. 2011. Clic Energia.

aO, A. X. F.; STOLFI, J.; LOTUFO, R. A. The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Washington, DC, USA, v. 26, n. 1, p. 19–29, 2004.

ARTHANARI, T. S.; RAMAMURTHY, K. G. An extension of two machines sequencing problem. *Operations Research*, v. 8, n. 4, p. 10–22, 1971.

ATHAN, T. W.; PAPALAMBROS, P. Y. A note on weitherd criteria methods for compromise solutions in multi-objective optimization. *Engineering Optimization*, v. 27, n. 2, p. 155–176, 1996.

AYDIN, I.; KARAKOSE, M.; AKIN, E. A multi-objective artificial immune algorithm for parameter optimization in support vector machine. *Applied Soft Computing*, v. 11, n. 1, p. 120–129, 2011.

BANDYOPADHYAY, S. et al. A simulated annealing-based multiobjective optimization algorithm: Amosa. *IEEE Transactions on Evolutionary Computation*, v. 12, n. 3, p. 269–283, June 2008.

BEIJSTERVELDT, C.; BOOMSMA, D. Genetics of the human electroencephalogram (EEG) and event-related brain potentials (ERPs): a review. *Human Genetics*, Springer-Verlag, v. 94, n. 4, p. 319–330, 1994. ISSN 0340-6717.

BELKADI, K.; GOURGAND, M.; BENYETTOU, M. Parallel genetic algorithms with migration for the hybrid flow shop scheduling problem. *Journal of Applied Mathematics and Decision Sciences*, v. 2006, p. 17, 2006.

BEUME, N.; NAUJOKS, B.; EMMERICH, M. Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, v. 181, n. 3, p. 1653–1669, 2007.

BHUVANESWARI, M. C. *Application of Evolutionary Algorithms for Multi-Objective Optimization in VLSI and Embedded Systems*. [S.l.]: Springer India, 2015.

BOWMAN, V. J. On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. In: THIRIEZ, H.; ZIONTS, S. (Ed.). *Multiple Criteria Decision Making: Proceedings of a Conference Jouy-en-Josas, France May 21–23, 1975*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1976. p. 76–86.

BRIDGMAN, P. *Dimensional Analysis*. [S.l.]: Yale University Press, 1922.

CAMPISI, P.; ROCCA, D. L. Brain waves for automatic biometric-based user recognition. *Information Forensics and Security, IEEE Transactions on*, v. 9, n. 5, p. 782–800, May 2014. ISSN 1556-6013.

CARPENTER, G. A. et al. Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, v. 3, n. 5, p. 698–713, Sep. 1992.

CHAND, S.; WAGNER, M. Evolutionary many-objective optimization: A quick-start guide. *Surveys in Operations Research and Management Science*, v. 20, n. 2, p. 35–42, 2015.

CHANKONG, V.; HAIMES, Y. *Multiobjective decision making: theory and methodology.* [S.l.]: North Holland, 1983.

COELLO, C. A. C.; LAMONT, G. B.; VELDHUIZEN, D. A. V. *Evolutionary Algorithms for Solving Multi-Objective Problems.* New York, NY, USA: Springer-Verlag, 2007.

CORNE, D. W. et al. Pesa-ii: Region-based selection in evolutionary multiobjective optimization. In: *Proceedings of the Genetic and Evolutionary Computation Conference.* [S.l.]: Morgan Kaufmann Publishers, 2001. p. 283–290.

CORTES, C.; VAPNIK, V. Support vector networks. *Machine Learning*, v. 20, p. 273–297, 1995.

DAS, I.; DENNIS, J. E. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM J. on Optimization*, v. 8, n. 3, p. 631–657, 1998.

DEB, K. *Multi-Objective Optimization Using Evolutionary Algorithms.* New York, NY, USA: John Wiley & Sons, Inc., 2001. ISBN 047187339X.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, v. 6, n. 2, p. 182–197, 2002.

DENIZ, A. et al. Robust multiobjective evolutionary feature subset selection algorithm for binary classification using machine learning techniques. *Neurocomputing*, v. 241, p. 128–146, 2017.

DIAO, R.; SHEN, Q. Nature inspired feature selection meta-heuristics. *Artificial Intelligence Review*, Kluwer Academic Publishers, Norwell, MA, USA, v. 44, n. 3, p. 311–340, 2015. ISSN 0269-2821.

EPE. *Statistical Yearbook of Electricity 2014 - 2013 Baseline Year.* [S.l.], 2014.

ERFANI, T.; UTYUZHNIKOV, S. Directed search domain: A method for even generation of pareto frontier in multiobjective optimization. *Engineering Optimization*, v. 43, n. 5, p. 467–484, 2011.

FABRE, M. G. *Optimización de Problemas Com Más de Tres Objetivos Mediante Algoritmos Evolutivos.* Tese (Doutorado) — Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, 2009.

FALCÓN, R.; M., A.; NAYAK, A. Fault identification with binary adaptive fireflies in parallel and distributed systems. In: *Proceedings of the IEEE Congress on Evolutionary Computation.* [S.l.]: IEEE, 2011. p. 1359–1366.

FANG, X. et al. Smart grid – the new and improved power grid: A survey. *IEEE Communications Surveys and Tutorials*, v. 14, n. 4, p. 944–980, 2012.

FARINA, M.; AMATO, P. On the optimal solution definition for many-criteria optimization problems. In: *Proceedings of Annual Meeting of the North American Fuzzy Information Processing Society*. [S.l.: s.n.], 2002. p. 233–238.

FEDOROVICI, L. et al. Embedding gravitational search algorithms in convolutional neural networks for OCR applications. In: *7th IEEE International Symposium on Applied Computational Intelligence and Informatics*. [S.l.: s.n.], 2012. p. 125–130.

FIELDSEND, J. E.; SINGH, S. Pareto multiobjective nonlinear regression modelling to aid capm analogous forecasting. In: *Proceedings of the International Joint Conference on Neural Networks*. [S.l.: s.n.], 2002. v. 1, p. 388–393.

FIELDSEND, J. E.; SINGH, S. Pareto evolutionary neural networks. *IEEE Transactions on Neural Networks*, v. 16, n. 2, p. 338–354, March 2005.

FIRPI, H. A.; GOODMAN, E. Swarmed feature selection. In: *Proceedings of the 33rd Applied Imagery Pattern Recognition Workshop*. Washington, DC, USA: IEEE Computer Society, 2004. p. 112–118.

FONSECA, C. M.; FLEMING, P. J. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: *Proceedings of the Fifth International Conference in Genetic Algorithm*. [S.l.: s.n.], 1993. p. 416–423.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, American Statistical Association, Taylor & Francis, Ltd., v. 32, n. 200, p. 675–701, 1937.

FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 11, n. 1, p. 86–92, 1940.

GARCÍA-PEDRAJAS, N.; HERVÁS-MARTÍNEZ, C.; PÉREZ, J. Muñoz. Multi-objective cooperative coevolution of artificial neural networks (multi-objective cooperative networks). *Neural Networks*, v. 15, n. 10, p. 1259–1278, 2002.

GASSER, T. et al. Development of the eeg of school-age children and adolescents ii. topography. *Electroencephalography and clinical neurophysiology*, v. 69, n. 2, p. 100–109, 1988.

GEEM, Z. W. *Music-Inspired Harmony Search Algorithm: Theory and Applications*. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2009. ISBN 364200184X, 9783642001840.

GEN, M.; YUN, Y. Soft computing approach for reliability optimization: State-of-the-art survey. *Reliability Engineering and System Safety*, v. 91, n. 9, p. 1008–1026, 2006.

GOLDBERGER, A. L. et al. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, v. 101, n. 23, p. e215–e220, 2000.

GUARRACINO, M. et al. Supervised classification of distributed data streams for smart grids. *Energy Systems*, Springer-Verlag, v. 3, n. 1, p. 95–108, 2012.

GUO, L. et al. Automatic feature extraction using genetic programming: An application to epileptic EEG classification. *Expert Systems with Applications*, v. 38, n. 8, p. 10425–10436, 2011. ISSN 0957-4174.

HAIMES, Y. Y.; LASDON, L. S.; WISMER, D. A. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-1, n. 3, p. 296–297, 1971.

HAMDANI, T. M. et al. Multi-objective feature selection with NSGA-II. In: BELICZYNSKI, B. et al. (Ed.). *Adaptive and Natural Computing Algorithms*. [S.l.]: Springer Berlin Heidelberg, 2007, (Lecture Notes in Computer Science, v. 4431). p. 240–247.

HANDL, J.; KNOWLES, J. Evolutionary multiobjective clustering. In: YAO, X. et al. (Ed.). *Parallel Problem Solving from Nature - PPSN VIII*. [S.l.]: Springer Berlin Heidelberg, 2004, (Lecture Notes in Computer Science, v. 3242). p. 1081–1091.

HARRIS, T.; HARDIN, J. W. Exact wilcoxon signed-rank and wilcoxon mann-whitney ranksum tests. *Stata Journal*, Stata Press, College Station, TX, v. 13, n. 2, p. 337–343, 2013.

HARVEY, D. Y.; TODD, M. D. Automated feature design for numeric sequence classification by genetic programming. *IEEE Transactions on Evolutionary Computation*, v. 19, n. 4, p. 474–489, 2015.

HATAMLOU, A. Black hole: A new heuristic optimization approach for data clustering. *Information Sciences*, v. 222, n. 0, p. 175–184, 2013.

HATANAKA, T.; KONDO, N.; UOSAKI, K. Multi-objective structure selection for radial basis function networks based on genetic algorithm. In: *The Congress on Evolutionary Computation*. [S.l.: s.n.], 2003. v. 2, p. 1095–1100.

HAYKIN, S. *Neural Networks: A Comprehensive Foundation (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2007. ISBN 0131471392.

HEGAZY, A. E.; MAKHLOUF, M. A.; EL-TAWEL, G. S. Improved salp swarm algorithm for feature selection. *Journal of King Saud University - Computer and Information Sciences*, 2018.

HINTON, G. E. Training products of experts by minimizing contrastive divergence. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 14, n. 8, p. 1771–1800, 2002. ISSN 0899-7667.

HINTON, G. E. A practical guide to training restricted boltzmann machines. In: MONTAVON, G.; ORR, G.; MÜLLER, K.-R. (Ed.). *Neural Networks: Tricks of the Trade*. [S.l.]: Springer Berlin Heidelberg, 2012, (Lecture Notes in Computer Science, v. 7700). p. 599–619.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 18, n. 7, p. 1527–1554, 2006.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems: An introductory analysis with applications to biology, control and artificial intelligence.* [S.l.]: Oxford, England: U Michigan Press, 1975.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems.* Cambridge, MA, USA: MIT Press, 1992.

HORN, J. Multicriterion decision making. *Handbook of evolutionary computation*, v. 1, p. F1, 1997.

HORN, J.; NAFPLIOTIS, N.; GOLDBERG, D. E. A niched pareto genetic algorithm for multiobjective optimization. In: *Proceedings of the IEEE Conference on Evolutionary Computation.* [S.l.: s.n.], 1994. v. 1, p. 82–87.

HUANG, S.-C.; LO, Y.-L.; LU, C.-N. Non-technical loss detection using state estimation and analysis of variance. *IEEE Transactions on Power Systems*, v. 28, n. 3, p. 2959–2966, Aug 2013. ISSN 0885-8950.

HUGHES, E. J. Evolutionary many-objective optimisation: many once or one many? In: *Proceedings of the IEEE Congress on Evolutionary Computation.* [S.l.: s.n.], 2005. v. 1, p. 222–227.

IGEL, C. Multi-objective model selection for support vector machines. In: *Proceedings of the Third International Conference on Evolutionary MultiCriterion Optimization.* [S.l.]: SpringerVerlag, 2005. p. 534–546.

ISHIBUCHI, H.; TSUKAMOTO, N.; NOJIMA, Y. Evolutionary many-objective optimization: A short review. In: *IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence).* [S.l.: s.n.], 2008. p. 2419–2426.

JAIN, A. K.; ROSS, A. A.; NANDAKUMAR, K. *Introduction to Biometrics.* [S.l.]: Springer, 2011.

JAIN, S.; DESHPANDE, G. Parametric modeling of brain signals. In: *Biotechnology and Bioinformatics, 2004. Proceedings. Technology for Life: North Carolina Symposium on.* [S.l.: s.n.], 2004. p. 85–91.

JANKOWSKI, N.; GROCHOWSKI, M. Comparison of instances seletion algorithms i. algorithms survey. In: ______. *Artificial Intelligence and Soft Computing - ICAISC 2004: 7th International Conference.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 598–603.

JEET, K.; DHIR, R. Software clustering using hybrid multi-objective black hole algorithm. In: *International Conference on Software Engineering and Knowledge Engineering.* [S.l.: s.n.], 2016. p. 650–653.

JIN, Y. Neural network regularization and ensembling using multi-objective evolutionary algorithms. In: *Proceedings of IEEE Congress on Evolutionary Computation.* [S.l.]: IEEE Press, 2004. p. 1–8.

JIN, Y.; SENDHOFF, B. Pareto-based multiobjective machine learning: An overview and case studies. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 38, n. 3, p. 397–415, May 2008.

JONES, D.; TAMIZ, M. *Practical Goal Programming*. [S.l.]: Springer, 2010.

JOZEFOWIEZ, N.; SEMET, F.; TALBI, E. Multi-objective vehicle routing problems. *European Journal of Operational Research*, v. 189, n. 2, p. 293–309, 2008.

JÚNIOR, L. A. P. et al. Unsupervised non-technical losses identification through optimum-path forest. *Electric Power Systems Research*, Elsevier, v. 140, p. 413–423, 2016.

JURCAK, V.; TSUZUKI, D.; DAN, I. 10/20, 10/10, and 10/5 systems revisited: Their validity as relative head-surface-based positioning systems. *NeuroImage*, v. 34, n. 4, p. 1600–1611, 2007.

KAHRAMAN, C. et al. Multiprocessor task scheduling in multistage hybrid flow-shops: A parallel greedy algorithm approach. *Applied Soft Computing*, v. 10, n. 4, p. 1293–1300, 2010.

KAVEH, A.; TALATAHARI, S. A novel heuristic optimization method: charged system search. *Acta Mechanica*, Springer Wien, v. 213, n. 3, p. 267–289, 2010.

KENNEDY, J.; EBERHART, R. *Swarm Intelligence*. [S.l.]: M. Kaufman, 2001.

KHALILI-DAMGHANI, K.; ABTAHI, A.-R.; TAVANA, M. A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems. *Reliability Engineering and System Safety*, v. 111, n. 0, p. 58–75, 2013.

KHALILI-DAMGHANI, K.; ABTAHI, A. R.; TAVANA, M. A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems. *Reliability Engineering and System Safety*, v. 111, n. 0, p. 58–75, 2013.

KHALOULI, S.; GHEDJATI, F.; HAMZAOUI, A. A meta-heuristic approach to solve a jit scheduling problem in hybrid flow shop. *Engineering Applications of Artificial Intelligence*, v. 23, n. 5, p. 765–771, 2010.

KIZILOZ, H. E. et al. Novel multiobjective tlbo algorithms for the feature subset selection problem. *Neurocomputing*, v. 306, p. 94–107, 2018.

KNOWLES, J.; CORNE, D. The pareto archived evolution strategy: a new baseline algorithm for pareto multiobjective optimisation. In: *Proceedings of the Congress on Evolutionary Computation*. [S.l.: s.n.], 1999. v. 1, p. 98–105.

KONAK, A.; COIT, D. W.; SMITH, A. E. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, v. 91, n. 9, p. 992–1007, 2006.

KOSTÍLEK, M.; ŜTÁSTNÝ, J. EEG biometric identification: Repeatability and influence of movement-related EEG. In: *Proceedings of the International Conference on Applied Electronics*. [S.l.: s.n.], 2012. p. 147–150.

KOTTATHRA, K.; ATTIKIOUZEL, Y. A novel multicriteria optimization algorithm for the structure determination of multilayer feedforward neural networks. *Journal of Network and Computer Applications*, v. 19, n. 2, p. 135–147, 1996.

KOZA, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992. ISBN 0-262-11170-5.

KOZODOI, N. et al. A multi-objective approach for profit-driven feature selection in credit scoring. *Decision Support Systems*, v. 120, p. 106–117, 2019.

LATIFF, N. M. A. et al. Dynamic clustering using binary multi-objective particle swarm optimization for wireless sensor networks. In: *IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*. [S.l.: s.n.], 2008. p. 1–5.

LAU, H. C. et al. A multi-objective memetic algorithm for vehicle resource allocation in sustainable transportation planning. In: *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2013. p. 2833–2839.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, 1998.

LI, B. et al. Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys*, ACM, New York, NY, USA, v. 48, n. 1, p. 13:1–13:35, 2015.

LI, W.; LIU, L.; GONG, W. Multi-objective uniform design as a svm model selection tool for face recognition. *Expert Systems with Applications*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 38, n. 6, p. 6689–6695, jun. 2011.

LI, Z.; LIAO, H.; COIT, D. W. A two-stage approach for multi-objective decision making with applications to system reliability optimization. *Reliability Engineering and System Safety*, v. 94, n. 10, p. 1585–1592, 2009.

LIU, G. P.; KADIRKAMANATHAN, V. Learning with multi-objective criteria. In: *Artificial Neural Networks, 1995., Fourth International Conference on*. [S.l.: s.n.], 1995. p. 53–58.

LIU, L.; DENG, M. An evolutionary artificial neural network approach for breast cancer diagnosis. In: *Proceedings of Third International Conference on Knowledge Discovery and Data Mining*. [S.l.: s.n.], 2010. p. 593–596.

MAHDAVI, M.; FESANGHARY, M.; DAMANGIR, E. An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, v. 188, n. 2, p. 1567 – 1579, 2007.

MARICHELVAM, M. K.; PRABAHARAN, T.; YANG, X.-S. A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems. *IEEE Transactions on Evolutionary Computation*, v. 18, n. 2, p. 301–305, April 2014.

MARLER, R. T.; ARORA, J. S. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, v. 26, n. 6, p. 369–395, 2004.

MESSAC, A. Physical programming - effective optimization for computational design. *AIAA Journal*, v. 34, n. 1, p. 149–158, 1996.

MESSAC, A.; ISMAIL-YAHAYA, A.; MATTSON, C. A. The normalized normal constraint method for generating the pareto frontier. *Structural and Multidisciplinary Optimization*, v. 25, n. 2, p. 86–98, 2003.

MIETTINEN, K. *Nonlinear Multiobjective Optimization.* [S.l.]: Springer US, 1998.

MIETTINEN, K.; MäKELä, M. M. Interactive bundle-based method for nondifferentiable multiobjeective optimization: nimbus. *Optimization*, v. 34, n. 3, p. 231–246, 1995.

MILLARD, R.; EMMERTON, M. Non-technical losses - how do other countries tackle the problem? In: *AMEU Proceedings 2009.* [S.l.: s.n.], 2009. p. 67–81.

MIRANDA, P. B. C. et al. Combining meta-learning with multi-objective particle swarm algorithms for svm parameter selection: An experimental analysis. In: *Brazilian Symposium on Neural Networks.* [S.l.: s.n.], 2012. p. 1–6.

MIRJALILI, S. et al. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, v. 114, p. 163–191, 2017.

MIRJALILI, S.; HASHIM, S. Z. M. BMOA: Binary magnetic optimization algorithm. In: *3rd IEEE International Conference on Machine Learning and Computing.* Singapore: [s.n.], 2011. v. 1, p. 201–206.

MIRJALILI, S.; LEWIS, A. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, v. 9, p. 1–14, 2013.

MISRA, K. B.; LJUBOJEVIC, M. D. Optimal reliability design of a system: A new look. *IEEE Transactions on Reliability*, R-22, n. 5, p. 255–258, 1973.

MONEDERO, I. et al. Midas: Detection of non-technical losses in electrical consumption using neural networks and statistical techniques. In: *Proc. of the Intl. Conference on Computational Science and Applications.* Springer Berlin/Heidelberg: Lecture Notes in Computer Science, 2006. v. 3984.

MORITA, M. et al. Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition. In: *Seventh International Conference on Document Analysis and Recognition.* [S.l.: s.n.], 2003. p. 666–670.

MUELLER-GRITSCHNEDER, D.; GRAEB, H.; SCHLICHTMANN, U. A successive approach to compute the bounded pareto front of practical multiobjective optimization problems. *SIAM J. on Optimization*, v. 20, n. 2, p. 915–934, 2009.

MUKHOPADHYAY, A.; MAULIK, U.; BANDYOPADHYAY, S. A survey of multiobjective evolutionary clustering. *ACM Comput. Surv.*, v. 47, n. 4, p. 1–46, 2015.

NAGI, J. et al. Nontechnical loss detection for metered consumers in power utility using support vector machines. *IEEE Transactions on Power Delivery*, v. 25, p. 1162–1171, 2010.

NAGI, J. et al. Improving svm-based nontechnical loss detection in power utility using the fuzzy inference system. *Power Delivery, IEEE Transactions on*, v. 26, n. 2, p. 1284–1285, 2011.

NAKAMURA, R. et al. Optimum-path forest pruning parameter estimation through harmony search. In: *24th SIBGRAPI Conference on Graphics, Patterns and Images.* [S.l.: s.n.], 2011. p. 181–188.

NAKAYAMA, H.; SAWARAGI, Y. Satisficing trade-off method for multiobjective programming. In: _____. *Interactive Decision Analysis.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1984. p. 113–122.

NAKIB, A.; OULHADJ, H.; SIARRY, P. Non-supervised image segmentation based on multiobjective optimization. *Pattern Recognition Letters*, v. 29, n. 2, p. 161–172, 2008.

NAKIB, A.; OULHADJ, H.; SIARRY, P. Image thresholding based on pareto multiobjective optimization. *Engineering Applications of Artificial Intelligence*, v. 23, n. 3, p. 313–320, 2010.

NEMATI, M.; MOMENI, H.; BAZRKAR, N. Article: Binary black holes algorithm. *International Journal of Computer Applications*, v. 79, n. 6, p. 36–42, 2013. Published by Foundation of Computer Science, New York, USA.

NEMENYI, P. *Distribution-free Multiple Comparisons.* [S.l.]: Princeton University, 1963.

NIZAR, A.; DONG, Z.; WANG, Y. Power utility nontechnical loss analysis with extreme learning machine method. *IEEE Transactions on Power Systems*, v. 23, p. 946–955, 2008.

NIZAR, A.; ZHAO, J. H.; DONG, Z. Y. Customer information system data pre-processing with feature selection techniques for non-technical losses prediction in an electricity market. In: *International Conference on Power System Technology.* [S.l.: s.n.], 2006. p. 1–7.

NOCEDAL, J.; WRIGHT, S. *Numerical Optimization.* New York, NY, USA: Springer-Verlag, 2006.

NUNES, T. M. et al. EEG signal classification for epilepsy diagnosis via optimum path forest – a systematic assessment. *Neurocomputing*, v. 136, p. 103–123, 2014. ISSN 0925-2312.

NUWER, M. R. et al. IFCN standards for digital recording of clinical EEG. *Electroencephalography and Clinical Neurophysiology*, v. 106, n. 3, p. 259–261, 1998.

OCAK, H. Automatic detection of epileptic seizures in EEG using discrete wavelet transform and approximate entropy. *Expert Systems with Applications*, v. 36, n. 2, Part 1, p. 2027–2036, 2009. ISSN 0957-4174.

OLIVEIRA, L. S. et al. Feature selection using multi-objective genetic algorithms for handwritten digit recognition. In: *Object recognition supported by user interaction for service robots.* [S.l.: s.n.], 2002. v. 1, p. 568–571.

OLIVEIRA, M. de; BOSON, D.; PADILHA-FELTRIN, A. A statistical analysis of loss factor to determine the energy losses. In: *Proceedings of the IEEE/PES Transmission and Distribution Conference and Exposition: Latin America.* [S.l.: s.n.], 2008. p. 1–6.

OMKAR, S. N. et al. Artificial bee colony (abc) for multi-objective design optimization of composite structures. *Applied Soft Computing*, v. 11, n. 1, p. 489–499, 2011.

ONETY, R. E. et al. Multiobjective optimization of mpls-ip networks with a variable neighborhood genetic algorithm. *Applied Soft Computing*, v. 13, n. 11, p. 4403–4412, 2013.

OĝUZ, C.; ERCAN, M. F. A genetic algorithm for hybrid flow-shop scheduling with multiprocessor tasks. *Journal of Scheduling*, v. 8, n. 4, p. 323–351, 2005.

PALANIAPPAN, R. Method of identifying individuals using VEP signals and neural network. *IEE Proceedings - Science, Measurement and Technology*, v. 151, n. 1, p. 16–20, 2004.

PALANIAPPAN, R.; MANDIC, D. EEG based biometric framework for automatic identity verification. *The Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, Springer US, v. 49, n. 2, p. 243–250, 2007.

PALIT, S. et al. A cryptanalytic attack on the knapsack cryptosystem using binary firefly algorithm. In: *Computer and Communication Technology (ICCCT), 2011 2nd International Conference on*. [S.l.: s.n.], 2011. p. 428–432.

PAPA, J. et al. Robust pruning of training patterns for optimum-path forest classification applied to satellite-based rainfall occurrence estimation. *IEEE Geoscience and Remote Sensing Letters*, v. 7, n. 2, p. 396–400, 2010.

PAPA, J. P. et al. Efficient supervised optimum-path forest classification for large datasets. *Pattern Recognition*, Elsevier Science Inc., New York, NY, USA, v. 45, n. 1, p. 512–520, 2012.

PAPA, J. P.; FALCÃO, A. X.; SUZUKI, C. T. N. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, John Wiley & Sons, Inc., New York, NY, USA, v. 19, n. 2, p. 120–131, 2009. ISSN 0899-9457.

PAPA, J. P.; FERNANDES, S. E. N.; FALCÃO, A. X. Optimum-path forest based on k-connectivity: Theory and applications. *Pattern Recognition Letters*, v. 87, p. 117–126, 2017.

PAPA, J. P. et al. Feature selection through gravitational search algorithm. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2011. p. 2052–2055.

PAPA, J. P. et al. On the model selection of bernoulli restricted boltzmann machines through harmony search. In: *Proceedings of the Genetic and Evolutionary Computation Conference*. New York, NY, USA: ACM, 2015. p. 1449–1450.

PAPA, J. P. et al. Model selection for discriminative restricted boltzmann machines through meta-heuristic techniques. *Journal of Computational Science*, v. 9, p. 14–18, 2015. ISSN 1877-7503.

Papa, J. P. et al. Libopt: An open-source platform for fast prototyping soft optimization techniques. *ArXiv e-prints*, 2017. Http://adsabs.harvard.edu/abs/2017arXiv170405174P.

PAPA, J. P. et al. Feature selection through binary brain storm optimization. *Computers & Electrical Engineering*, v. 72, p. 468 – 481, 2018.

PAPA, J. P.; SCHEIRER, W.; COX, D. D. Fine-tuning deep belief networks using harmony search. *Applied Soft Computing*, p. –, 2015. ISSN 1568-4946.

PARUCHURI, V.; DUBEY, S. An approach to determine non-technical energy losses in india. In: *2012 14th International Conference on Advanced Communication Technology (ICACT)*. [S.l.: s.n.], 2012. p. 111–115.

PASSOS, L. A.; RODRIGUES, D. R.; PAPA, J. P. Fine tuning deep boltzmann machines through meta-heuristic approaches. In: IEEE. *2018 IEEE 12th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. [S.l.], 2018. p. 000419–000424.

PEIMANKAR, A. et al. Evolutionary multi-objective fault diagnosis of power transformers. *Swarm and Evolutionary Computation*, v. 36, p. 62–75, 2017.

PELLETIER, J. L.; VEL, S. S. Multi-objective optimization of fiber reinforced composite laminates for strength, stiffness and minimal mass. *Computers & Structures*, v. 84, n. 29–30, p. 2065–2080, 2006.

PEREIRA, D. R. et al. Social-spider optimization-based support vector machines applied for energy theft detection. *Computers & Electrical Engineering*, v. 49, p. 25–38, 2016.

PETROU, M.; KADYROV, A. The trace transform and its applications. In: MARSHALL, S.; HARVEY, N.; SHAH, D. (Ed.). *Noblesse Workshop on Non-Linear Model Based Image Analysis*. [S.l.]: Springer London, 1998. p. 207–214.

PETTERSSON, F.; CHAKRABORTI, N.; SAXéN, H. A genetic algorithms based multi-objective neural net applied to noisy blast furnace data. *Applied Soft Computing*, v. 7, n. 1, p. 387–397, 2007.

POLLOCK, V.; SCHNEIDER, L.; LYNESS, S. Reliability of topographic quantitative EEG amplitude in healthy late-middle-aged and elderly subjects. *Electroencephalography and Clinical Neurophysiology*, v. 79, n. 1, p. 20–26, 1991.

PORRAS, J. et al. Identification of non-technical electricity losses in power distribution systems by applying techniques of information analysis and visualization. *IEEE Latin America Transactions (Revista IEEE America Latina)*, v. 13, n. 3, p. 659–664, March 2015. ISSN 1548-0992.

POULOS, M.; RANGOUSSI, M.; ALEXANDRIS, N. Neural network based person identification using EEG features. In: *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. [S.l.: s.n.], 1999. p. 1117–1120.

POULOS, M. et al. Parametric person identification from the EEG using computational geometry. In: *Proceedings of IEEE International Conference on Electronics, Circuits and Systems, 1999*. [S.l.: s.n.], 1999. v. 2, p. 1005–1008.

POURPANAH, F. et al. Feature selection based on brain storm optimization for data classification. *Applied Soft Computing*, v. 80, p. 761–775, 2019.

PęKALSKA, E.; DUIN, R. P. W.; PACLíK, P. Prototype selection for dissimilarity-based classifiers. *Pattern Recognition*, v. 39, n. 2, p. 189–208, 2006.

QI, X.; ZHU, Y.; ZHANG, H. A new meta-heuristic butterfly-inspired algorithm. *Journal of Computational Science*, v. 23, p. 226–239, 2017.

QIAN, X. et al. Unsupervised texture image segmentation using multiobjective evolutionary clustering ensemble algorithm. In: *IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2008. p. 3561–3567.

RADTKE, P. V.; WONG, T.; SABOURIN, R. A multi-objective memetic algorithm for intelligent feature extraction. In: COELLO, C. A. C.; AGUIRRE, A. H.; ZITZLER, E. (Ed.). *Evolutionary Multi-Criterion Optimization*. [S.l.]: Springer Berlin Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3410). p. 767–781.

RAMOS, C. C. O. et al. On the study of commercial losses in brazil: A binary black hole algorithm for theft characterization. *IEEE Transactions on Smart Grid*, PP, n. 99, p. 1–1, 2016.

RAMOS, C. C. O. et al. A novel algorithm for feature selection using harmony search and its application for non-technical losses detection. *Computers & Electrical Engineering*, v. 37, n. 6, p. 886–894, 2011.

RAMOS, C. C. O. et al. New insights on non-technical losses characterization through evolutionary-based feature selection. *IEEE Transactions on Power Delivery*, v. 27, n. 1, p. 140–146, 2012.

RAMOS, C. C. O. et al. Fast non-technical losses identification through optimum-path forest. In: *Proceedings of The 15th International Conference on Intelligent System Applications to Power Systems (ISAP)*. [S.l.: s.n.], 2009. p. 1–5.

RAMOS, C. C. O. et al. A new approach for nontechnical losses detection based on optimum-path forest. *IEEE Transactions on Power Systems*, v. 26, p. 181–189, 2011.

RANI, K. N. A. et al. Modified cuckoo search algorithm in weighted sum optimization for linear antenna array synthesis. In: *IEEE Symposium on Wireless Technology and Applications*. [S.l.: s.n.], 2012. p. 210–215.

RAO, S. S. Game theory approach for multiobjective structural optimization. *Computers & Structures*, v. 25, n. 1, p. 119–127, 1987.

RASHEDI, E.; Nezamabadi-pour, H.; SARYAZDI, S. GSA: A gravitational search algorithm. *Information Sciences*, Elsevier Science Inc., New York, NY, USA, v. 179, n. 13, p. 2232–2248, 2009.

RASHEDI, E.; NEZAMABADI-POUR, H.; SARYAZDI, S. BGSA: Binary gravitational search algorithm. *Natural Computing*, Springer Netherlands, v. 9, n. 3, p. 727–745, 2010.

RASHIDI, E.; JAHANDAR, M.; ZANDIEH, M. An improved hybrid multi-objective parallel genetic algorithm for hybrid flow shop scheduling with unrelated parallel machines. *The International Journal of Advanced Manufacturing Technology*, v. 49, n. 9, p. 1129–1139, 2010.

ROCCA, D. L. et al. Human brain distinctiveness based on eeg spectral coherence connectivity. *Biomedical Engineering, IEEE Transactions on*, v. 61, n. 9, p. 2406–2412, Sept 2014. ISSN 0018-9294.

RODRIGUES, D.; PAPA, J. P.; ADELI, H. Meta-heuristic multi- and many-objective optimization techniques for solution of machine learning problems. *Expert Systems*, v. 34, n. 6, p. e12255, 2017.

RODRIGUES, D. et al. A wrapper approach for feature selection based on bat algorithm and optimum-path forest. *Expert Systems with Applications*, v. 41, n. 5, p. 2250–2258, 2013. ISSN 0957-4174.

RODRIGUES, D. et al. Optimizing feature selection through binary charged system search. In: *Proceedings of 15th International Conference on Computer Analysis of Images and Patterns*. [S.l.: s.n.], 2013. p. 377–384.

RODRIGUES, D. et al. Black hole algorithm for non-technical losses characterization. In: *IEEE 6th Latin American Symposium on Circuits and Systems (LASCAS2015)*. [S.l.: s.n.], 2015. p. 1–4.

RODRIGUES, D. et al. Eeg-based person identification through binary flower pollination algorithm. *Expert Systems with Applications*, v. 62, p. 81–90, 2016.

RODRIGUES, D.; SOUZA, A. N.; PAPA, J. P. Pruning optimum-path forest classifiers using multi-objective optimization. In: *30th SIBGRAPI Conference on Graphics, Patterns and Images*. [S.l.: s.n.], 2017. p. 127–133.

RODRIGUES, D.; YANG, X.-S.; PAPA, J. P. Fine-tuning deep belief networks using cuckoo search. In: YANG, X.-S.; PAPA, J. P. (Ed.). *Bio-Inspired Computation and Applications in Image Processing*. [S.l.]: Academic Press, 2016. p. 47–59.

RODRIGUES, D. et al. Binary flower pollination algorithm and its application to feature selection. In: YANG, X.-S. (Ed.). *Recent Advances in Swarm Intelligence and Evolutionary Computation*. [S.l.]: Springer International Publishing, 2015, (Studies in Computational Intelligence, v. 585). p. 85–100.

ROSA, G. H. et al. Fine-tuning convolutional neural networks using harmony search. In: PARDO, A.; KITTLER, J. (Ed.). *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. [S.l.]: Springer International Publishing, 2015, (Lecture Notes in Computer Science, v. 9423). p. 683–690.

ROSALES-PÉREZ, A. et al. Multi-objective model type selection. *Neurocomputing*, v. 146, n. 0, p. 83–94, 2014.

RUIZ, R.; MAROTO, C. A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. *European Journal of Operational Research*, v. 169, n. 3, p. 781–800, 2006.

SAFONT, G. et al. Combination of multiple detectors for eeg based biometric identification/authentication. In: *Proc. of the IEEE Intl. Carnahan Conference on Security Technology*. [S.l.: s.n.], 2012. p. 230–236.

SAHA, S.; BANDYOPADHYAY, S. A generalized automatic clustering algorithm in a multiobjective framework. *Applied Soft Computing*, v. 13, n. 1, p. 89–108, 2013.

SANEI, S.; CHAMBERS, J. *EEG signal processing*. Chichester, England; Hoboken, NJ: John Wiley & Sons, 2007.

SATO, H.; AGUIRRE, H. E.; TANAKA, K. Controlling dominance area of solutions and its impact on the performance of moeas. In: *Proceedings of the 4th International Conference on Evolutionary Multi-criterion Optimization*. Berlin, Heidelberg: Springer-Verlag, 2007. p. 5–20.

SCHAFFER, J. D. Multiple objective optimization with vector evaluated genetic algorithms. In: *Proceedings of the 1st International Conference on Genetic Algorithms*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1985. p. 93–100.

SCHALK, G. et al. BCI2000: a general-purpose brain-computer interface (bci) system. *IEEE Transactions on Biomedical Engineering*, v. 51, n. 6, p. 1034–1043, 2004.

SHIAU, D.-F.; CHENG, S.-C.; HUANG, Y.-M. Proportionate flexible flow shop scheduling via a hybrid constructive genetic algorithm. *Expert Systems with Applications*, v. 34, n. 2, p. 1133–1143, 2008.

SHIRAKAWA, S.; NAGAO, T. Evolutionary image segmentation based on multiobjective clustering. In: *IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2009. p. 2466–2473.

SHUKLA, P. K.; DEB, K.; TIWARI, S. Comparing classical generating methods with an evolutionary multi-objective optimization method. In: COELLO, C. A. C.; AGUIRRE, A. H.; ZITZLER, E. (Ed.). *Evolutionary Multi-Criterion Optimization: Third International Conference, EMO 2005, Guanajuato, Mexico, March 9-11, 2005. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 311–325.

SINGH, M. R.; MAHAPATRA, S. S. A swarm optimization approach for flexible flow shop scheduling with multiprocessor tasks. *The International Journal of Advanced Manufacturing Technology*, v. 62, n. 1, p. 267–277, 2012.

SIVASUBRAMANI, S.; SWARUP, K. S. Multi-objective harmony search algorithm for optimal power flow problem. *International Journal of Electrical Power and Energy Systems*, v. 33, n. 3, p. 745–752, 2011.

SMITH, C.; JIN, Y. Evolutionary multi-objective generation of recurrent neural network ensembles for time series prediction. *Neurocomputing*, v. 143, p. 302–311, 2014.

SODHRO, A. H. et al. Artificial intelligence based qos optimization for multimedia communication in iov systems. *Future Generation Computer Systems*, v. 95, p. 667–680, 2019.

SODHRO, A. H. et al. An adaptive qos computation for medical data processing in intelligent healthcare applications. *Neural Computing and Applications*, Jan 2019.

SODHRO, A. H.; PIRBHULAL, S.; ALBUQUERQUE, V. H. C. Artificial intelligence-driven mechanism for edge computing-based industrial applications. *IEEE Transactions on Industrial Informatics*, v. 15, n. 7, p. 4235–4243, July 2019.

SODHRO, A. H. et al. Towards an optimal resource management for iot based green and sustainable smart cities. *Journal of Cleaner Production*, v. 220, p. 1167–1179, 2019.

SODHRO, A. H. et al. *Medical-QoS Based Telemedicine Service Selection Using Analytic Hierarchy Process*. Cham: Springer International Publishing, 2017.

SPOLAOR, N.; LORENA, A. C.; LEE, H. D. Use of multiobjective genetic algorithms in feature selection. In: *Eleventh Brazilian Symposium on Neural Networks*. [S.l.: s.n.], 2010. p. 146–151.

SRINIVAS, N.; DEB, K. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, v. 2, p. 221–248, 1994.

STORN, R.; PRICE, K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, v. 11, n. 4, p. 341–359, 1997.

STRAFFIN, P. D. *Game Theory and Strategy*. [S.l.]: Mathematical Association of America, 1993.

SUBASI, A. EEG signal classification using wavelet feature extraction and a mixture of expert model. *Expert Systems with Applications*, v. 32, n. 4, p. 1084–1093, 2007. ISSN 0957-4174.

SUTTORP, T.; IGEL, C. Multi-objective optimization of support vector machines. In: JIN, Y. (Ed.). *Multi-Objective Machine Learning*. [S.l.]: Springer Berlin Heidelberg, 2006, (Studies in Computational Intelligence, v. 16). p. 199–220.

TANG, L.; WANG, X. An improved particle swarm optimization algorithm for the hybrid flowshop scheduling to minimize total weighted completion time in process industry. *IEEE Transactions on Control Systems Technology*, v. 18, n. 6, p. 1303–1314, 2010.

TARADEH, M. et al. An evolutionary gravitational search-based feature selection. *Information Sciences*, v. 497, p. 219 – 239, 2019.

TAU, G. Z.; PETERSON, B. S. Normal development of brain circuits. *Neuropsychopharmacology*, v. 35, n. 1, p. 147–168, 2009.

TSENG, C.-T.; LIAO, C.-J. A particle swarm optimization algorithm for hybrid flow-shop scheduling with multiprocessor tasks. *International Journal of Production Research*, v. 46, n. 17, p. 4655–4670, 2008.

VAPNIK, V. N. *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

VATOLKIN, I.; PREUSS, M.; RUDOLPH, G. Multi-objective feature selection in music genre and style recognition tasks. In: *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*. New York, NY, USA: ACM, 2011. p. 411–418.

WANG, H.; TAN, L.; NIU, B. Feature selection for classification of microarray gene expression cancers using bacterial colony optimization with multi-dimensional population. *Swarm and Evolutionary Computation*, v. 48, p. 172–181, 2019.

WANG, H.-M.; CHOU, F.-D.; WU, F.-C. A simulated annealing for hybrid flow shop scheduling with multiprocessor tasks to minimize makespan. *The International Journal of Advanced Manufacturing Technology*, v. 53, n. 5, p. 761–776, 2011.

WANG, L. et al. Feature selection based on meta-heuristics for biomedicine. *Optimization Methods and Software*, Taylor & Francis, v. 29, n. 4, p. 703–719, 2014.

WANG, X.; TANG, L. A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers. *Computers & Operations Research*, v. 36, n. 3, p. 907–918, 2009.

WANG, Y.; GAO, S.; GAO, X. Common spatial pattern method for channel selection in motor imagery based brain-computer interface. In: *Engineering in Medicine and Biology 27th Annual IEEE Conference*. [S.l.: s.n.], 2005. p. 5392–95.

WIEGAND, S.; IGEL, C.; HANDMANN, U. Evolutionary multi-objective optimization of neural networks for face detection. *International Journal of Computation Intelligence and Applications*, v. 4, n. 3, p. 2004, 2004.

WILCOXON, F. Individual comparisons by ranking methods. *Biometrics Bulletin*, International Biometric Society, v. 1, n. 6, p. 80–83, 1945.

WILSON, D. R.; MARTINEZ, T. R. Reduction techniques for instance-based learning algorithms. *Machine Learning*, v. 38, n. 3, p. 257–286, 2000.

WOLPERT, D. H.; MACREADY, W. G. No free lunch theorems for optimization. *Trans. Evol. Comp*, IEEE Press, Piscataway, NJ, USA, v. 1, n. 1, p. 67–82, 1997. ISSN 1089-778X.

XUE, B.; ZHANG, M.; BROWNE, W. N. Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE Transactions on Cybernetics*, v. 43, n. 6, p. 1656–1671, 2013.

YANG, X.-S. Firefly algorithm, stochastic test functions and design optimisation. *International Journal Bio-Inspired Computing*, Inderscience Publishers, v. 2, n. 2, p. 78–84, 2010.

YANG, X.-S. Firefly algorithm, stochastic test functions and design optimisation. *International Journal Bio-Inspired Computing*, Inderscience Publishers, v. 2, n. 2, p. 78–84, 2010.

YANG, X.-S. Flower pollination algorithm for global optimization. In: *Proceedings of the 11th International Conference on Unconventional Computation and Natural Computation*. Berlin, Heidelberg: Springer-Verlag, 2012. (UCNC'12), p. 240–249.

YANG, X.-S. Multiobjective firefly algorithm for continuous optimization. *Engineering with Computers*, v. 29, n. 2, p. 175–184, 2013.

YANG, X.-S.; S., D. Engineering optimisation by cuckoo search. *International Journal of Mathematical Modelling and Numerical Optimisation*, v. 1, p. 330–343, 2010.

YANG, Y. et al. Subject-specific channel selection for classification of motor imagery electroencephalographic data. In: *Acoustics, Speech and Signal Processing (ICASP), 2013 IEEE International Conference*. [S.l.: s.n.], 2013. p. 1159–1169.

YOU, D.; BENITEZ-QUIROZ, C. F.; MARTINEZ, A. M. Multiobjective optimization for model selection in kernel methods in regression. *IEEE Transactions on Neural Networks and Learning Systems*, v. 25, n. 10, p. 1879–1893, Oct 2014.

YU, P. L.; LEITMANN, G. Compromise solutions, domination structures, and salukvadze's solution. *Journal of Optimization Theory and Applications*, v. 13, n. 3, p. 362–378, 1974.

YUSIONG, J. P. T.; JR., P. C. N. Training neural networks using multiobjective particle swarm optimization. In: JIAO, L. et al. (Ed.). *Advances in Natural Computation*. [S.l.]: Springer Berlin Heidelberg, 2006, (Lecture Notes in Computer Science, v. 4221). p. 879–888.

ZAVALA, G. R. et al. A survey of multi-objective metaheuristics applied to structural optimization. *Structural and Multidisciplinary Optimization*, v. 49, n. 4, p. 537–558, 2014.

ZELENY, M. *Multiple Criteria Decision Making*. [S.l.]: McGraw-Hill, 1982.

ZHANG, H. et al. A hybrid multi-objective artificial bee colony algorithm for burdening optimization of copper strip production. *Applied Mathematical Modelling*, v. 36, n. 6, p. 2578–2591, 2012.

ZHANG, Q.; MAHFOUF, M. A new reduced space searching algorithm (rssa) and its application in optimal design of alloy steels. In: *IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2007. p. 1815–1822.

ZHANG, Q.; MAHFOUF, M. A nature-inspired multi-objective optimisation strategy based on a new reduced space searching algorithm for the design of alloy steels. *Engineering Applications of Artificial Intelligence*, v. 23, n. 5, p. 660–675, 2010.

ZHANG, Y.; ROCKETT, P. I. Feature extraction using multi-objective genetic programming. In: JIN, Y. (Ed.). *Multi-Objective Machine Learning*. [S.l.]: Springer Berlin Heidelberg, 2006, (Studies in Computational Intelligence, v. 16). p. 75–99.

ZHANG, Y.; ROCKETT, P. I. Multiobjective genetic programming feature extraction with optimized dimensionality. In: SAAD, A. et al. (Ed.). *Soft Computing in Industrial Applications*. [S.l.]: Springer Berlin Heidelberg, 2007, (Advances in Soft Computing, v. 39). p. 159–168.

ZHENG, Y.-J.; SONG, Q.; CHEN, S.-Y. Multiobjective fireworks optimization for variable-rate fertilization in oil crop production. *Applied Soft Computing*, v. 13, n. 11, p. 4253–4263, 2013.

ZITZLER, E. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. Tese (Doutorado) — Institut für Technische Informatik und Kommunikationsnetze, 1999.

ZITZLER, E.; KÜNZLI, S. Indicator-based selection in multiobjective search. In: YAO, X. et al. (Ed.). *Parallel Problem Solving from Nature - PPSN VIII*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 832–842.

ZITZLER, E.; THIELE, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, v. 3, n. 4, p. 257–271, Nov 1999.

şERIFOğLU, S. F.; ULUSOY, G. Multiprocessor task scheduling in multistage hybrid flow-shops: a genetic algorithm approach. *Journal of the Operational Research Society*, v. 55, n. 5, p. 504–512, 2004.