UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ON THE ADVANCES IN PATTERN RECOGNITION USING OPTIMUM-PATH FOREST

LUIS CLAUDIO SUGI AFONSO

ORIENTADOR: PROF. DR. JOÃO PAULO PAPA

São Carlos – SP

Setembro/2020

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ON THE ADVANCES IN PATTERN RECOGNITION USING OPTIMUM-PATH FOREST

LUIS CLAUDIO SUGI AFONSO

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: Processamento de Imagens e Sinais
Orientador: Prof. Dr. João Paulo Papa

São Carlos – SP

Setembro/2020

UNIVERSIDADE FEDERAL DE SÃO CARLOS

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

Folha de Aprovação

Defesa de Tese de Doutorado do candidato Luis Claudio Sugi Afonso, realizada em 24/09/2020.

Comissão Julgadora:

Prof. Dr. João Paulo Papa (UFSCar)

Prof. Dr. Alexandre Luis Magalhães Levada (UFSCar)

Prof. Dr. Aparecido Nilceu Marana (UNESP)

Prof. Dr. Moacir Antonelli Ponti (ICMC/USP)

Prof. Dr. Anderson de Rezende Rocha (UNICAMP)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Aos meus pais.

AGRADECIMENTOS

Gostaria de agradecer aos meus pais por todo o apoio durante toda a caminhada não apenas nesta etapa, mas em toda a vida.

Gostaria de agradecer ao meu orientador, Prof. João Paulo Papa, pelas oportunidades, confiança e orientação desde o mestrado.

Gostaria de agradecer ao Prof. Nilceu pelas oportunidades, confiança e orientação desde a graduação. Ao Prof. Ivan Guilherme da UNESP/Rio Claro pelas oportunidades e confiança e aos membros do grupo do projeto Radiar, Prof. Daniel Pedronette, Prof. Alexandro Baldassin, Gustavo Sousa, Pedro Ivo e Rafael Murari.

Gsotaria de agradecer aos Professores Marcos Cavenaghi e Roberta Spolon do Departamento de Computação da UNESP/Bauru pela oportunidade que me foi dada ainda na graduação. Ali foi o início do trajeto até este momento.

Gostaria de agradecer aos membros do Recogna pela amizade e companheirismo em mais esta etapa. Obrigado, Clayton, Leandro (Lelis), Douglas, Luiz Felix, Rafael Pires, e todos os demais membros.

Gostaria de agradecer ao Prof. Alexandre Vidal (IG-Unicamp) por ter me aceitado como membro de seu grupo nas primeiras etapas do doutorado, e também por todo o conhecimento e apoio durante o período. Também gostaria de agradecer a Michelle Kuroda, Bruno Honório, Ulisses, Leandro, Gui, Mateus e Aline pela amizade, companheirismo e tudo que me ensinaram.

Gostaria de agradecer a Giedry Fernanda Trindade por todos os momentos que me proporcionou, o amor, o carinho, as alegrias, as lições que, mesmo que indiretamente, fizeram toda a diferença durante mais esta etapa.

Também gostaria de agradecer ao Luis Martins, Alan Peixinho, Azael, Otávio, Samuel, Fabi, e Helena pela amizade no período em que estive em Campinas.

Por último, mas não menos importante, agradeço a UNESP e UFSCar.

Be the change you want see in the world.

Mahatma Gandhi

RESUMO

Técnicas de reconhecimento de padrões (RP) têm sido de grande importância para a solução de muitos problemas de diversos níveis de complexidade e áreas de estudo. A ideia por trás das técnicas de RP está em criar modelos capazes de classificar elementos nunca vistos. Basicamente, os problemas de reconhecimento de padrões podem ser divididos em duas categorias: problemas de aprendizado (i) supervisionado e (ii) não-supervisionado. Essas categorias estão relacionadas com a existência ou não de elementos rotulados para auxiliar no "aprendizado" dos algoritmos de RP. Um conjunto de elementos de treinamento é fundamental para que as técnicas de RP sejam capazes de identificar padrões existentes, e a presença de dados rotulados pode auxiliar na criação de modelos mais robutos. Muitas técnicas foram desenvolvidas para lidar com tais problemas e estão bem-estabelecidas na literatura. Uma técnica desenvolvida recentemente diz respeito ao classificador baseado em grafos denominado Floresta de Caminhos Ótimos (OPF - *Optimum-Path Forest*), o qual possui as versões de aprendizado supervisionado, semi-supervisionado e não-supervisionado. OPF modela as amostras de um conjunto de dados como sendo os nós de um grafo e as conexões (arestas) são definidas a partir de uma relação de adjacência pré-definida. Apesar de ser uma abordagem recente, OPF já foi empregado em inúmeras aplicações distintas e tem apresentado resultados promissores e superando até mesmo técnicas bem estabelecidas na literatura. Contudo, ainda há muito a ser estudado, avaliado e proposto com relação ao uso e desempenho do classificador em questão. Este trabalho de qualificação investiga e propõe variações e alterações no algoritmo tradicional do OPF das versões de aprendizado supervisionado e não-supervisionado com os objetivos de avaliar seu desempenho em pontos ainda não explorados e superar algumas de suas deficiências.

Palavras-chave: Floresta de Caminhos Ótimos, Reconhecimento de padrões, Aprendizado de máquina

ABSTRACT

Pattern recognition (PR) techniques have been paramount to solve different and complex problems in many fields of study. The basic idea behind PR techniques is to compute a model capable of classifying unknown samples. Pattern recognition can be categorized as problems of (i) supervised, and (ii) unsupervised learning. This categorization is related to the existence or absence of labeled data to support the learning process. The learning process is mandatory for PR techniques to learn the data distribution, and the existence of labeled data is an additional information that helps to build more robust models. Many techniques were proposed and are well-established in the literature. The Optimum-Path Forest (OPF) is a graph-based classifier proposed recently, which comprises the models for supervised, semi-supervised and unsupervised learning. The OPF models dataset samples as nodes of a graph and their connections (edges) are defined by some pre-defined adjacency relation. Although very recent, OPF has already been employed in numerous applications and showed promising results, and even outperformed other well-known classifiers. Nonetheless, there is still a lot to be investigated, evaluated and proposed concerning the use and performance of the OPF classifier. This dissertation investigates e proposes variations and modifications to the traditional OPF algorithms concerning supervised and unsupervised learning aiming the assessment of its performance in not yet explored scenarios and to overcome its drawbacks.

Keywords: Optimum-Path Forest, Pattern Recognition, Machine Learning

List of Figures

LIST OF TABLES

CONTENTS

Chapter 1

INTRODUCTION

Pattern recognition (PR) techniques have been paramount to solve different and complex problems in many fields of study. The motivation in the development of such techniques is to perform recognition tasks more accurately, or faster, or just to aid in the mechanical and repetitively ones (PAL; PAL, 2011). The basic idea behind PR techniques is to compute a model capable of classifying unknown samples by learning data distribution over the feature space. The learning process requires a training set that might carry information (i.e., label) that helps to minimize the classification error in the training set. Therefore, the existence of such labeled training data creates two fundamental problems in pattern recognition: (i) supervised and (ii) unsupervised learning (KPALMA; RONSIN, 2007).

Artificial Neural Networks using Multi-Layer Perceptrons (ANN-MLP) , Support Vector Machines (SVM) and Naïve Bayes classifier (BC) figure among the most popular supervised learning-based algorithms, which take advantage of a full labeled dataset. Although well-established in the literature, the mentioned classifiers have their drawbacks, such as to find an optimum set of parameters for better accuracy rates, computational cost, and especially the difficult to handle non-separable classes in the feature space. For instance, ANN-MLP could have its performance improved if more layers are added, but that comes with the increase of computational cost. Non-linear problems require SVM to map data into higher-dimensional spaces, which also makes the method costly, and BC makes a decision based on the probability density of each class. If such information is not available, one must be estimated.

In the opposite way, unsupervised problems do not have any labeled information regarding the training samples at their disposal, which makes the learning task more difficult. Hence, the fundamental problem in unsupervised learning is to identify clusters in an unlabeled dataset, such that samples from the same cluster should share some level of similarity. Many methods were proposed where the learning problem is addressed with different perspectives, such as data clustering and density estimation, just to mention a few (SCHWENKER; TRENTIN, 2014). Self-Organizing Maps, *k*-means, and Hidden Markov Models figure among the most common unsupervised algorithms. The Self-Organizing Maps (KOHONEN, 2001) is a popular unsupervised neural-network model for the analysis of high-dimensional input data. Difficulties on using SOM comprise to define the map size, which is related to the number of input data, and hierarchical representations are hard to be identified (RAUBER; MERKL; DITTENBACH, 2002). The *k*-means (JAIN, 2010) is one of the simplest clustering algorithms that partitions data in an iterative fashion using *k*-centroids. The parameter *k* is defined a priori, which is not a straightforward task. Also, its random initialization has a considerable impact on the final result.

Graph-based machine learning techniques have their appeal as well. The Optimum-Path Forest (OPF) is a framework for the design of graph-based classifiers that comprises three models: (i) supervised (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012; PAPA; FERNANDES; FALCÃO, 2017), (ii) semi-supervised (AMORIM; FALCÃO; CARVALHO, 2014; AMORIM et al., 2016) and (iii) unsupervised (ROCHA; CAPPABIANCO; FALCÃO, 2009). The traditional OPF algorithm models samples as the nodes of a graph, which are connected to each other based on some predefined adjacency relation. Learning and classification phases are carried out by a competition-based process ruled by a few “key” samples called *prototypes*. The prototypes are responsible for conquering the remaining samples of a dataset by offering them optimum-path costs. The outcome of such a process is a set of optimum-path trees (forest) being each rooted by a different prototype. As to be further discussed, the set of prototypes are defined based on the model employed.

Proposed in 2009 (PAPA; FALCÃO, 2009a), OPF has been applied in a wide variety of applications, such as network security, image and video analysis, disease identification, just to mention a few. Moreover, it has shown competitive accuracy rates outperforming some well-known and state-of-art classifiers (e.g., SVM, *k*-NN, and BC), it is quite fast in both training and classification process since it does not interpret pattern recognition problems as a separating hyperplanes task, and is parameterless (PAPA; FALCÃO, 2009a). Despite the numerous successful applications, there is still room for investigations and improvements to be proposed concerning different performance aspects of the OPF. A few works proposed to improve the supervised learning algorithm. Papa and Falcão (PAPA; FALCÃO, 2009b) proposed a pruning algorithm to eliminate samples considered irrelevant without compromising the accuracy in dynamic applications. Castelo et al. (CASTELO-FERNÁNDEZ et al., 2010) proposed the detection and elimination of outliers in the training set using a penalty computed based on the frequency a sample is classified as either false positive or false negative (FERNANDES; PAPA, 2017).

This thesis is driven by studying OPF behavior under some problems and proposing techniques that can be incorporated into it or modifying its own to improve accuracy. Hence, many different problems were studied along with the work. The main questions to be answered in the thesis are: *Although OPF obtained interesting results in many applications, is it possible to improve its accuracy? If so, which of its steps can be modified to do so?* Hence, the main contribution of the thesis is to foster the research on the Optimum-Path Forest classifier by: (i) proposing and investigating its behavior over changes performed on its working mechanisms, and (ii) applying in problems that it has not been explored.

This thesis is organized as a collection of papers that investigate and evaluate the supervised and unsupervised OPF classifiers. Chapter 2 introduces the supervised and unsupervised OPF learning models. Chapters 3, 4, 5, 6, and 7 present works concerning the supervised approach. Chapters 8, 9, 10, and 11 present the studies related to the unsupervised model. Finally, conclusions and future works are stated in Chapter 12. The remainder of this chapter is divided into two sections (supervised and unsupervised applications) that provide an overview of each published work and the questions we are willing to answer. Figure 1.1 depicts how the chapters are related to each other, and the following sections provide a summary of the problems considered in this work.

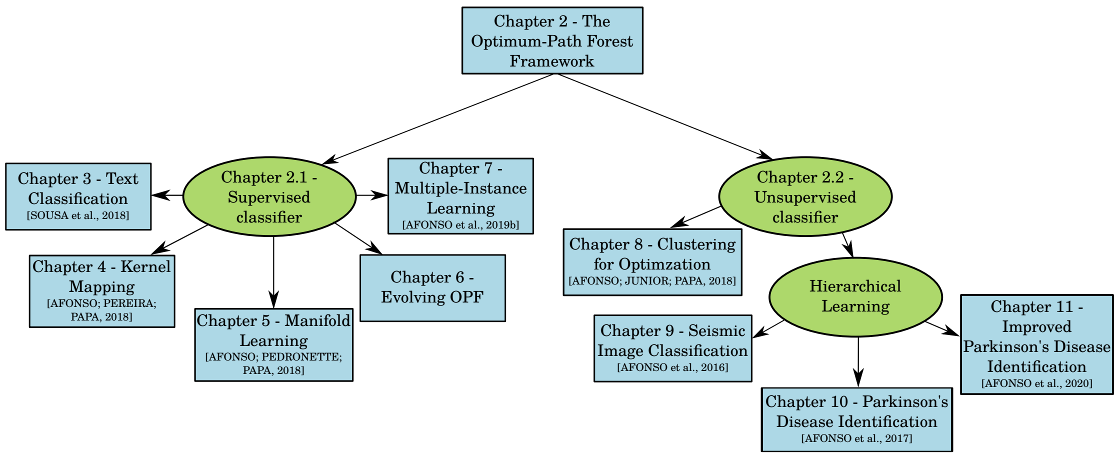

Chapter 2 - The Optimum-Path Forest Framework
Chapter 2.1 - Supervised classifier
Chapter 2.2 - Unsupervised classifier
Chapter 3 - Text Classification [SOUSA et al., 2018]
Chapter 7 - Multiple-Instance Learning [AFONSO et al., 2019b]
Chapter 4 - Kernel Mapping [AFONSO; PEREIRA; PAPA, 2018]
Chapter 6 - Evolving OPF
Chapter 5 - Manifold Learning [AFONSO; PEDRONETTE; PAPA, 2018]
Chapter 8 - Clustering for Optimzation [AFONSO; JUNIOR; PAPA, 2018]
Hierarchical Learning
Chapter 9 - Seismic Image Classification [AFONSO et al., 2016]
Chapter 11 - Improved Parkinson's Disease Identification [AFONSO et al., 2020]
Chapter 10 - Parkinson's Disease Identification [AFONSO et al., 2017]

Figure 1.1: Chapter organization.

1.1 Supervised learning-based applications

Well drilling monitoring is an essential task to prevent faults, save resources, and take care of environmental and eco-planning businesses. During drilling, it is required that staff fill out a log to keep track of the activities that are currently occurring. With such data analyzed and processed, it is possible to learn how to prevent faults and take corrective actions in real-time. However, the most important information is usually stored in a free-text format, thus complicating the task of automated text mining. Despite the complex architecture and being computational costly, many works employed the well-known deep-learning algorithms for text mining. In the work presented in Chapter 3, we aim at answering the question *how would a more straightforward and faster learner pattern recognition algorithm such as Optimum-Path Forest classifier behave in such context?* Experiments showed that OPF combined with text-based features are a compelling source to learn patterns in drilling reports.

Despite the interesting accuracy in many applications, OPF's learning process can still be improved. Chapters 4 and 5 investigate two modifications in the learning algorithm. The former work deals with kernel functions. The modeling of real-world problems as graphs along with the problem of non-linear distributions comes up with the idea of applying kernel functions in feature spaces. Roughly speaking, the idea is to seek for well-behaved samples in higher dimensional spaces, where the assumption of linearly separable samples is stronger. The current OPF algorithm implementation works naturally with non-linear situations, but it does not map samples from one space to another. The question to be answered in the work presented in Chapter 4 is *how much the application of kernel functions could benefit OPF's learning process and consequently improve its accuracy?* The proposed technique was evaluated over a benchmark comprised of 11 datasets, whose results outperformed the well-known Support Vector Machines and the standard OPF classifier for some situations.

Appropriate metrics are paramount for machine learning and pattern recognition. In Content-based Image Retrieval-oriented applications, low-level features and pairwise-distance metrics are usually not capable of representing similarity among the objects as observed by humans. Therefore, metric learning from available data has become crucial in such applications, but just a few related approaches take into account the contextual information inherent from the samples for a better accuracy performance. The work presented in Chapter 5 answers the question *would the combination of an unsupervised manifold learning algorithm with the Optimum-Path Forest provide more accurate recognition rates?* The proposed approach was evaluated in some public datasets and evidenced the validity of metric learning in the context of OPF classifiers. Also, it was showed the proposed approach could outperform standard OPF-based classifiers that are trained over the original manifold.

As to be further described, the prototypes are computed through a minimum spanning tree over the training set and promotes the samples nearby the decision boundary as prototypes. Although such methodology has obtained promising results in the past year, it can be prone to overfitting. The work presented in Chapter 6 aims to answer the question: *can an evolutionary-based approach estimate better prototypes than using the minimum spanning tree?* To do so, it is proposed a metaheuristic-based approach (OPF$_{mh}$) for the selection of prototypes, being such a task modeled as an optimization problem whose goal is to improve accuracy. The experimental results showed the OPF$_{mh}$ can reduce overfitting, as well as the number of prototypes in many situations. Moreover, OPF$_{mh}$ achieved competitive accuracies and outperformed OPF in the experimental scenarios.

Multiple-instance (MI) learning aims at modeling problems that are better described by several instances of a given sample instead of individual descriptions often employed by standard machine learning approaches. In binary-driven MI problems, the entire bag is considered positive if one (at least) sample is labeled as positive. On the other hand, a bag is considered negative if it contains all samples labeled as negative as well. In the work presented in Chapter 7, we are willing to *introduce the Optimum-Path Forest (OPF) classifier to the context of multiple-instance learning paradigm, and to evaluate it in different scenarios that range from molecule description, text categorization, and anomaly detection in well-drilling report classification.* The experimental results showed that two different OPF classifiers can be suitable to handle problems in the multiple-instance learning paradigm.

1.2 Unsupervised learning-based applications

As aforementioned, the fundamental problem in unsupervised learning is the identification of clusters. Due to the lack of labeled information, clustering techniques have been paramount in the last years. Selecting the clustering algorithm and the number of clusters can be paramount for obtaining better results in any problem, as shown in Afonso et al. (AFONSO et al., 2012). Therefore, the works presented in Chapters 8, 9, 10, and 11 were dedicated to the investigation of the unsupervised OPF performance under a few problems and to raise points to be improved and explored.

The work presented in Chapter 8 proposes a modification in the Brain Storm Optimization (BSO) working mechanism using the unsupervised Optimum-Path Forest. Among the many interesting meta-heuristic optimization algorithms, one can find those inspired by both the swarm and social behavior of human beings. The BSO is motivated by the brainstorming process performed by human beings to find solutions and solve problems. Such a process involves clustering the possible solutions, which can be sensitive to the number of groupings and the clustering technique itself. In this work, we are willing to answer if *the performance of*

BSO using OPF is as good as to its traditional algorithm and how much the usage of OPF can benefit the optimization process. The proposed approach is evaluated in a set of six benchmarking functions and showed promising results, outperforming the traditional BSO and a second variant makes use of the well-known Self-Organizing Maps clustering technique.

The works presented in Chapters 9, 10, and 11 were inspired by the deep learning phenomenon. In its traditional approach, the unsupervised OPF could be compared to a shallow neural network (i.e., which learns simpler features), and yet has shown promising results. So, *what if we develop a deep-driven OPF-based architecture to learn features? Will such representations provide better recognition rates than those from a "shallow" OPF?* These questions were answered in three distinct applications where each of them is a modification of its previous.

The work presented in Chapter 9 proposed to fill a gap in OPF-based works by introducing a multi-scale approach to obtain more refined cluster representations through the OPF classifier. The proposed approach was validated in the context of high-resolution seismic images aiming at petroleum exploration, as well as in general-purpose applications. Chapter 10 deals with the automatic Parkinson's disease (PD) identification. Approximately $50,000$ to $60,000$ new cases of PD are diagnosed yearly. Despite being non-lethal, PD shortens the life expectancy of the ones affected with such disease. As such, researchers from different fields of study have put great effort in order to develop methods aiming the identification of PD in its early stages. The proposed work uses handwriting dynamics data acquired by a series of tasks and proposed the application of the deep-driven OPF clustering algorithm to learn a dictionary-like representation of each individual to automatic identify Parkinson's disease. Experimental results showed promising results that are comparable to some state-of-the-art approaches in the literature. Finally, Chapter 11 extended the work of the previous by learning representations from all layers instead of just from the last layer as in the work in Chapter 10. The proposed approach was also applied in the context automatic identification of Parkinson's disease

Chapter 2

THE OPTIMUM-PATH FOREST FRAMEWORK

This chapter presents the theoretical background behind the Optimum-Path Forest framework concerning supervised and unsupervised learning. The content was extracted from the paper "A Survey on Optimum-Path Forest Classification", which was submitted to ACM Computing Surveys and is currently under review. The survey provides an overview of the fundamentals and algorithms behind the Optimum-Path Forest framework concerning supervised, semi-supervised, and unsupervised learning, as well as their many applications.

2.1 Theoretical Background

A weighted graph $\mathcal{G} = (\mathcal{C}, \mathcal{E}, w)$ is formally defined as a triplet, in which $\mathcal{C}$ denotes the set of vertices (nodes) and $\mathcal{E}$ stands for the set of edges or arcs. Besides, $w : \mathcal{C} \times \mathcal{C} \rightarrow \Re^+$ stands for a function that weights each arc of the graph. An edge defines a connection between any two adjacent nodes and the connections are established according to an *adjacency relation* $\mathcal{A}$ (i.e., an irreflexive binary relation between nodes). The edges allow to perform the so-called "walks on the graph" and to find paths among nodes. A path $\pi_{\boldsymbol{s}}$ is defined as a sequence of adjacent nodes starting from any node and with terminus at node $\boldsymbol{s} \in C$, and it is called trivial when composed of a single node $\boldsymbol{s}$ being represented as $\langle \boldsymbol{s} \rangle$. An optimum-path is the one with a value $f(\pi_{\boldsymbol{s}})$ that satisfies $f(\pi_{\boldsymbol{s}}) \leq f(\tau_{\boldsymbol{s}})$, being $\tau_{\boldsymbol{s}}$ any other path in the graph with terminus at sample $\boldsymbol{s}$, and f a real-valued path-cost function. Figure 2.1 depicts a few possible paths in a graph.[1]

The main idea behind OPF is to partition a graph into a forest (i.e., set of trees), such that all trees are comprised of optimum paths, and the nodes within a tree share similar properties.

[1]In the pattern recognition context, the nodes encode the feature vectors.

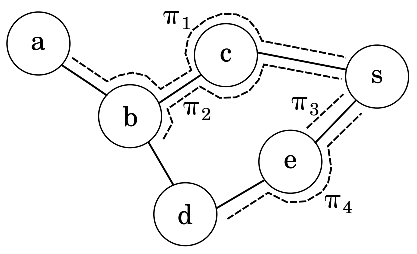


Figure 2.1: The dashed lines represent four possible paths to node s: $\pi_1 = \{a, b, c, s\}$, $\pi_2 = \{b, c, s\}$, $\pi_3 = \{e, s\}$, and $\pi_4 = \{d, e, s\}$.

In such a way, OPF can be understood as a generalization of Dijkstra's algorithm that works with any smooth path-cost function and is capable of finding optimum-paths from multiple sources (FALCÃO; STOLFI; LOTUFO, 2004). *Why does the OPF working mechanism is restricted to smooth path-cost functions?* The answer comes from the work by Falcão et al. (FALCÃO; STOLFI; LOTUFO, 2004) that describes the generalization of Dijkstra's algorithm works essentially with monotonic-incremental cost functions (i.e., a function whose values always increase within a certain interval). However, a few non-monotonic-incremental functions are also eligible as a path-cost function (e.g., the 4-connected adjacency and $f_{euc}(\pi)$, which defines the Euclidean distance between the endpoints of π). Hence, the authors defined more general conditions to cover such cases, which are:

- $f(\tau) \leq f(\pi)$,
- τ is optimum, and
- for any optimum path τ' ending at s, $f(\tau' \cdot \langle s, t \rangle) = f(\pi)$.

The conditions capture the essential features of such functions and the ones that satisfy them are called *smooth*. Moreover, the path-cost functions are comprised of two terms: (i) the initialization term $f(\langle s \rangle)$ that assigns an initial value to the trivial paths, usually according to the type of node (i.e., prototype or non-prototype); and (ii) the propagation term $f(\pi_s \cdot \langle s, t \rangle)$, which defines the path cost to be offered to a node during the bidding process.

Let $\mathcal{Z} = \{z_1, z_2, \ldots, z_m\}$ be a dataset of samples such that $z_i \in \mathcal{R}^n$. The Optimum-Path Forest encodes each sample as a node in the graph $\mathcal{G} = (\mathcal{Z}, \mathcal{A})$, and the graph-partitioning task is performed in a bidding-like process where the bid is the cost defined by a smooth-path cost function. As aforementioned, the OPF is capable of computing optimum-path trees from multiple sources, called *prototypes*, which are a subset of samples $\mathcal{P} \in \mathcal{Z}$. The heuristic implemented to select the set of prototypes varies according to the application. For instance, they can be selected at random, by computing minimum spanning trees, or by some probability density function. For the sake of explanation, let us consider the set $\mathscr{P}$ is already defined. After defining the path-cost function and the set of prototypes, the next step concerns computing the optimum-path trees (OPTs).

The optimum-path trees are computed in a bidding-like process where the prototypes play as bidders and the remaining samples (i.e., the non-prototype samples) are the prize. In the first step, all paths are trivial and initialized with a cost according to the type of node they are associated (i.e., prototype or non-prototype).

In the OPF context, the bidding order is defined by a priority queue whose ordering is defined by the trivial-path cost. The targets (i.e., prize-nodes) of each node are its adjacent ones and the value to be offered is a path cost. The winner is the one that offers the optimum-path cost and the prize-node is added to its "collection" (i.e., optimum-path tree). *What if multiple prototypes offer the same optimum-path cost?* In this case, OPF applies the first-takes-the-prize policy (i.e., first-in-first-out - FIFO), but any policy can be implemented as well. Figure 2.2 provides an overview of the process for a given adjacency relation $\mathscr{A}$.

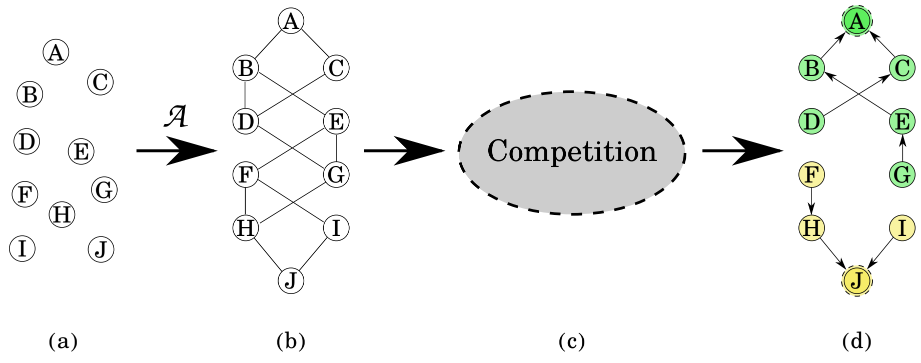


Figure 2.2: Overview of the OPF working mechanism: (a) initial dataset, (b) derived graph $\mathscr{G}$ according to $\mathscr{A}$, (c) computing the optimum-path trees, and (d) the resulting optimum-path forest with the prototypes highlighted (dashed circles).

Following, we present a toy example that computes optimum-paths from multiple sources considering the f_{sum} path-cost function:

$$\begin{aligned} f_{sum}(\langle \boldsymbol{s} \rangle) &= \begin{cases} 0 & \text{if } \boldsymbol{s} \in \mathscr{P} \\ +\infty & \text{otherwise,} \end{cases} \\ f_{sum}(\pi_{\boldsymbol{s}} \cdot (\boldsymbol{s}, \boldsymbol{t})) &= f_{sum}(\pi_{\boldsymbol{s}}) + d(\boldsymbol{s}, \boldsymbol{t}). \end{aligned} \tag{2.1}$$

Figure 2.3 illustrates the entire process from the (a) initialization of the graph to (l) its optimum-path forest. The values in red stand for the trivial-path cost of the nodes, in which each node $\boldsymbol{s} \in \mathscr{X}$ is initialized with a value according to its type and stored in a priority queue $\mathscr{Q}$.

The values in green stand for the weight of the edges defined by $d(\boldsymbol{s},\boldsymbol{t})$, which can be any distance or similarity metric computed over the arc $(\boldsymbol{s},\boldsymbol{t}) \in \mathscr{A}$. The prototypes are represented as dashed-circled nodes (i.e., nodes $\boldsymbol{a}$ and $\boldsymbol{j}$), and on the right side of each graph is shown the current priority queue $\mathscr{Q}$. The node in yellow is the one that has been removed from $\mathscr{Q}$, and the pink ones are its adjacent that are still in $\mathscr{Q}$. The priority queue stores the nodes in increasing order of costs and the ones with minimum values are the first to be popped out. Notice that $\mathscr{Q}$ is implemented using a binary heap. For the sake of explanation, we have a min-heap since we are trying to minimize the cost of each sample.

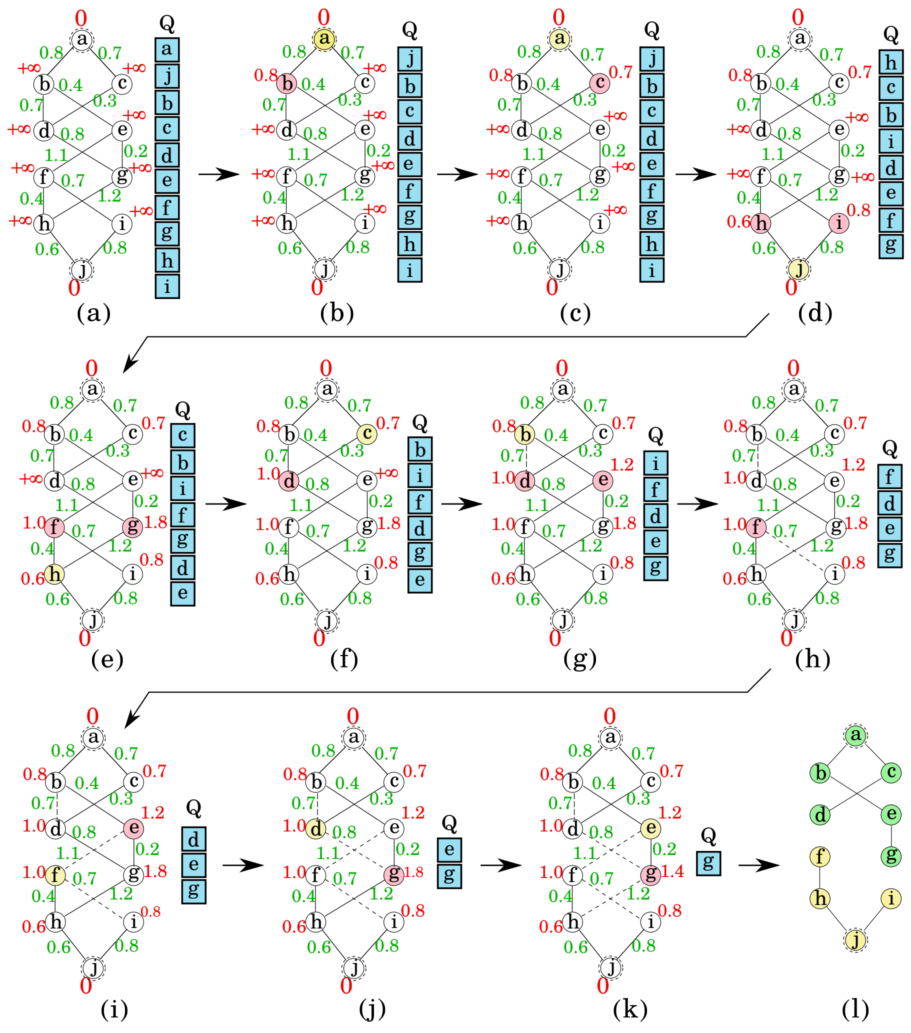


Figure 2.3: General working mechanism of OPF from: (a) initialization, (b)-(k) bidding process to (l) optimum-path forest.

In the toy example, node $\boldsymbol{a}$ is the first to be removed and tries to conquer its adjacent node $\boldsymbol{b}$ (Figure 2.3b) by offering a cost $C_a(\boldsymbol{b}) = 0.0 + 0.8$. Since $C_a(\boldsymbol{b}) < C(\boldsymbol{b})$, where $C(\boldsymbol{b})$ stands for the current cost of node $\boldsymbol{b}$, node $\boldsymbol{b}$ is conquered by node $\boldsymbol{a}$. Now, $\mathscr{O}(\boldsymbol{b}) = \{\boldsymbol{a}\}$ and $\boldsymbol{b}$ points to $\boldsymbol{a}$ where $\mathscr{O}$ is the predecessor map. The process is repeated for the next adjacent node of $\boldsymbol{a}$ (i.e., node $\boldsymbol{c}$). Similarly to node $\boldsymbol{b}$, the node $\boldsymbol{c}$ is also conquered by $\boldsymbol{a}$ with a cost of 0.7. Notice the priority queue is updated every time a node is conquered.

The next node in the queue is $\boldsymbol{j}$, and its adjacency is analyzed in the same way as it was done for $\boldsymbol{a}$. The bidding is performed until $\mathscr{Q}$ is empty. Notice the case of a tie in (j) where $\boldsymbol{d}$ tries to conquer $\boldsymbol{g}$ but it does not due to the FIFO policy. Also, notice in (k) where the conqueror of $\boldsymbol{g}$ changes to $\boldsymbol{e}$. The entire process is implemented in Algorithm 1, in which f sums up all arc-weights along a path.

Algorithm 1: Optimum-Path Forest Algorithm for Shortest-Path Computation

```
Input: A set of samples $\mathscr{X}$, set of prototypes $\mathscr{P} \subset \mathscr{X}$, a path-cost function $f$, and an
       adjacency relation $\mathscr{A}$.
Output: Predecessor map $\mathscr{O}$ and path-cost map $\mathscr{C}$.
Auxiliary: Priority queue $\mathscr{Q}$, and variable $cst$.
1  $\mathscr{Q}(\boldsymbol{s}) \leftarrow 0$
2  for all $\boldsymbol{s} \in \mathscr{X}$ do
3      $\mathscr{O}(\boldsymbol{s}) \leftarrow nil, \mathscr{C}(\boldsymbol{s}) \leftarrow +\infty, \mathscr{Q} \leftarrow \{\boldsymbol{s}\}$;
4  if $\boldsymbol{s} \in \mathscr{P}$ then
5      $\mathscr{C}(\boldsymbol{s}) \leftarrow 0$
6  while $\mathscr{Q} \neq \{\}$ do
7      Remove from $\mathscr{Q}$ a sample $\boldsymbol{s}$ such that $\mathscr{C}(\boldsymbol{s})$ is minimum;
8      for each sample $\boldsymbol{t} \in \mathscr{A}(\boldsymbol{s})$ and $\boldsymbol{t} \in \mathscr{Q}$ do
9          $cst = f(\mathscr{O}(\boldsymbol{s}), d(\boldsymbol{s}, \boldsymbol{t}))$;
10         if $cst < \mathscr{C}(\boldsymbol{t})$ then
11             $\mathscr{O}(\boldsymbol{t}) \leftarrow \boldsymbol{s}, \mathscr{C}(\boldsymbol{t}) \leftarrow cst$;
12             Update $\mathscr{Q}$;
13 return $\mathscr{O}, \mathscr{C}$
```

One of the main applications of the Optimum-Path Forest concerns general pattern recognition problems, but it is not restricted to them. By performing a few modifications in its general formulation, OPF becomes a powerful classification algorithm capable of learning under different assumptions (i.e., supervised, semi-supervised, and unsupervised). OPF-based classifiers explore the connectivity strength among samples to group the similar ones and their main advantages are two-fold: (i) the native treatment for multi-class problems, and (ii) the fact the attribute space geometry is not considered, which provides better results for datasets comprised of outliers or overlapping classes.

2.2 Supervised learning

This section introduces two strategies for supervised learning using OPF. They differ from each other on the adjacency relation, path-cost function, and the heuristic to select the prototypes.

2.2.1 OPF using Complete graph

The strategy introduced by Papa et al. (PAPA; FALCÃO; SUZUKI, 2009) implements the complete graph as an adjacency relation, i.e., there is an arc connecting any pair of nodes. Let $\mathcal{Z}$ be a λ-labeled dataset, and $\mathcal{Z}_1$, $\mathcal{Z}_2$, and $\mathcal{Z}_3$ stand for the training, validation, and testing sets, respectively, such that $\mathcal{Z} = \mathcal{Z}_1 \cup \mathcal{Z}_2 \cup \mathcal{Z}_3$, and $\mathcal{Z}_1 \cap \mathcal{Z}_3 = \emptyset$. The OPF with complete graph (OPF$_{cg}$) builds $\mathcal{P} \subseteq \mathcal{Z}_1$ by computing a minimum spanning tree (MST) over the complete graph $\mathcal{G} = (\mathcal{Z}_1, \mathcal{A})$. The outcome is an undirected acyclic graph where the edges are weighted by the distance between the adjacent feature vectors. Such spanning tree is optimum in the sense the sum of the weights of its edges is minimum if compared to any other spanning tree of the complete graph. The MST holds some theoretical properties, and it guarantees no error during training under certain circumstances (ALLèNE et al., 2010).

Since training aims at minimizing the classification error, the prototypes will be the samples in the resulting MST that are connected and belong to distinct classes, i.e., samples located at the decision boundaries. The option for such samples is based on the fact they are more likely to be misclassified due to the proximity to the influence region of other classes. By disconnecting such samples, we obtain the set of prototypes $\mathcal{P}$ and the bidding is performed through Algorithm 2, which uses the $f_{\max}$ as path-cost function:

$$
\begin{aligned}
f_{max}(\langle \boldsymbol{s} \rangle) &= \begin{cases} 0 & \text{if } \boldsymbol{s} \in \mathcal{P} \\ +\infty & \text{otherwise,} \end{cases} \\
f_{max}(\pi_{\boldsymbol{s}} \cdot (\boldsymbol{s}, \boldsymbol{t})) &= \max\{f_{max}(\pi_{\boldsymbol{s}}), d(\boldsymbol{s}, \boldsymbol{t})\}.
\end{aligned}
\tag{2.2}
$$

Lines 1–4 initialize the cost map by assigning cost 0 to the prototypes and $+\infty$ to the remaining samples. All samples have their predecessors set as *nil*, and the prototypes are inserted into the priority queue $\mathcal{Q}$. The loop at Line 5 iterates over all samples $\boldsymbol{s} \in \mathcal{P}$ first to start the competition process: if the cost offered *cst* is lower than the current cost of the prize-node $\boldsymbol{t}$, the sample $\boldsymbol{t}$ is labeled with the same label as sample $\boldsymbol{s}$ and added to its tree (Line 12). Notice the classes can be represented by multiple optimum-path trees, and there must be at least one per class.

Algorithm 2: Supervised training using complete graph

```
Input: A $\lambda$-labeled training set $\mathscr{Z}_1$, set of prototypes $\mathscr{P} \subset \mathscr{Z}_1$ and a function $d$ for
        distance computation.
Output: Predecessor map $\mathscr{O}$, path-cost map $\mathscr{C}$ and label map $\mathscr{L}$.
Auxiliary: Priority queue $\mathscr{Q}$, and variable $cst$.
for all $\boldsymbol{s} \in \mathscr{Z}_1$ do
    $\mathscr{O}(\boldsymbol{s}) \leftarrow nil, \mathscr{C}(\boldsymbol{s}) \leftarrow +\infty$;
for all $\boldsymbol{s} \in \mathscr{P}$ do
    $\mathscr{C}(\boldsymbol{s}) \leftarrow 0$, $\mathscr{L}(\boldsymbol{s}) = \lambda(\boldsymbol{s})$, $\mathscr{Q} \leftarrow \boldsymbol{s}$
while $\mathscr{Q} \neq \{\}$ do
    Remove from $\mathscr{Q}$ a sample $\boldsymbol{s}$ such that $\mathscr{C}(\boldsymbol{s})$ is minimum;
    for each sample $\boldsymbol{t} \in \mathscr{Z}_1$ such that $\boldsymbol{s} \neq \boldsymbol{t}$ and $\mathscr{C}(\boldsymbol{t}) > \mathscr{C}(\boldsymbol{s})$ do
        $cst \leftarrow \max\{\mathscr{C}(\boldsymbol{s}), d(\boldsymbol{s},\boldsymbol{t})\}$;
        if $cst < \mathscr{C}(\boldsymbol{t})$ then
            if $\mathscr{C}(\boldsymbol{t}) \neq +\infty$ then
                Remove $\boldsymbol{t}$ from $\mathscr{Q}$;
            $\mathscr{L}(\boldsymbol{t}) \leftarrow \mathscr{L}(\boldsymbol{s})$, $\mathscr{O}(\boldsymbol{t}) \leftarrow \boldsymbol{s}$, $\mathscr{C}(\boldsymbol{t}) \leftarrow cst$;
            $\mathscr{Q} \leftarrow \boldsymbol{t}$;
return $\mathscr{O}, \mathscr{C}, \mathscr{L}$;
```

The classification of a sample $\boldsymbol{t} \in \mathscr{Z}_3$ is performed by connecting it to all samples $\boldsymbol{s} \in \mathscr{Z}_1$ and making $\boldsymbol{t}$ part of the graph. By considering all possible paths between $\mathscr{P}$ and $\boldsymbol{t}$, we aim at finding the optimum-path to $\boldsymbol{t}$ with the same class of its most strongly connected prototype. Such path can be found by evaluating the optimum-path cost $\mathscr{C}(\boldsymbol{t})$ as follows:

$$\mathscr{C}(\boldsymbol{t}) = \min\{\max\{\mathscr{C}(\boldsymbol{s}), d(\boldsymbol{s},\boldsymbol{t})\}\}, \forall \boldsymbol{s} \in \mathscr{Z}_1. \tag{2.3}$$

Let $\boldsymbol{s}^* \in \mathscr{Z}_1$ be the node that satisfies Equation 2.3. The classification assigns $\mathscr{L}(\boldsymbol{s}^*)$ as the label of $\boldsymbol{t}$. A misclassification happens when $\mathscr{L}(\boldsymbol{s}^*) \neq \lambda(\boldsymbol{t})$. Figure 2.4 depicts the training (a)–(c) and classification (d)–(f) phases.

The OPF with complete graph works similarly to the nearest-neighbor (NN) classifier only when all training samples are prototypes, which is very rare and indicates that the set of attributes may not be the most appropriate to represent the samples. Moreover, OPF-based classifiers differentiate from NN-based ones on how decisions are made. The latter makes local-based decisions whereas the former is capable of global solutions based on the connectivity strength among samples.

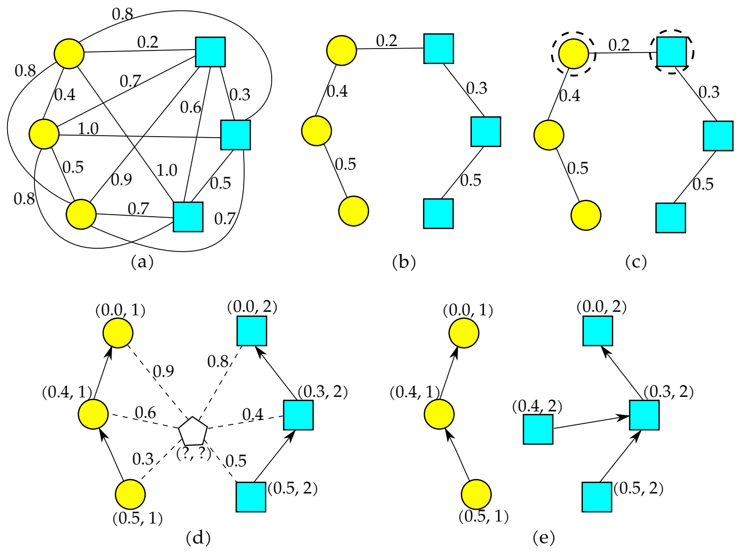


Figure 2.4: OPF$_{cg}$: training and classification phases.

2.2.2 OPF using k-nn graph

The second approach for supervised classification (OPF$_{knn}$) was proposed by Papa et al. (PAPA; FALCÃO, 2008; PAPA; FALCÃO, 2009a; PAPA; FERNANDES; FALCÃO, 2017), whose primary motivation was to explore different path-cost functions, adjacency relations, and heuristics to select prototypes. Their approach also encodes samples as nodes in a graph, but it employs a k-NN adjacency relation ($\mathscr{A}_k$). Since a k-NN graph does not guarantee a connected graph, computing the prototypes via MST is no longer an option. Hence, OPF$_{knn}$ implements a strategy similar to selecting elements nearby the centroids of clusters (ROSA et al., 2014), where the nodes located in areas of higher concentration of samples are chosen to build $\mathscr{P}$.

The density of each node $\boldsymbol{s} \in \mathscr{Z}_1$ is computed by a probability density function (pdf) as follows:

$$\rho(\boldsymbol{s}) = \frac{1}{\sqrt{2\pi\sigma^2}|\mathscr{A}_k(\boldsymbol{s})|} \sum_{\forall \boldsymbol{t} \in \mathscr{A}_k(\boldsymbol{s})} \exp\left(\frac{-d^2(\boldsymbol{s},\boldsymbol{t})}{2\sigma^2}\right), \tag{2.4}$$

where $\sigma = d_{max}/3$, and $d_{max} = \max\{d(\boldsymbol{s},\boldsymbol{t}) \in (\mathscr{Z}_1, \mathscr{A}_k)\}$. A coverage of 99.7% of the nodes within $d(\boldsymbol{s},\boldsymbol{t}) \in [0, 3\sigma]$ can be guaranteed in the computation of $\rho(\boldsymbol{s})$ by applying a Gaussian function. Moreover, the pdf is a Parzen-window estimation based on isotropic Gaussian kernel when the arcs are defined by $(\boldsymbol{s},\boldsymbol{t}) \in \mathscr{A}$ if $d(\boldsymbol{s},\boldsymbol{t}) \leq d_f$, being $d_{\max}$.

Imagine that the plotting of the density values gives us a surface where the variations of density create valleys and peaks, and the regions with the same values define plateaus. In cases where the plateau is a maximum of the pdf, it is urged to guarantee the connectivity between any pair of nodes at that maximum, so that any node can reach the remaining ones of the plateau and their influence zone via an optimum path. This condition is essential to avoid the segmentation of the plateau in too many OPTs (i.e., too many prototypes), where there should be only a few or even a single one. One can overcome this issue by modifying the adjacency relation $\mathscr{A}_k$ to a symmetric one of type $\mathscr{A}_2$:

$$\text{if } \boldsymbol{t} \in (\mathscr{A}_k(\boldsymbol{s})), \text{ and } \boldsymbol{s} \notin \mathscr{A}_k(\boldsymbol{t}), \text{and } \rho(\boldsymbol{s}) = \rho(\boldsymbol{t}), \text{ then } \mathscr{A}_2(\boldsymbol{t}) \leftarrow \mathscr{A}_k(\boldsymbol{t}) \cup \{\boldsymbol{s}\}. \tag{2.5}$$

In a theoretical situation where each maximum is comprised of a single sample, the following path-cost function can be applied:

$$\begin{aligned} f_1(\langle \boldsymbol{s} \rangle) &= \begin{cases} \rho(\boldsymbol{s}) & \text{if } \boldsymbol{s} \in \mathscr{P} \\ -\infty & \text{otherwise,} \end{cases} \\ f_1(\pi_{\boldsymbol{s}} \cdot \langle \boldsymbol{s}, \boldsymbol{t} \rangle) &= \min\{f_1(\pi_{\boldsymbol{s}}), \rho(\boldsymbol{t})\}. \end{aligned} \tag{2.6}$$

Every sample $\boldsymbol{s} \in \mathscr{P}$ defines a trivial path $\langle \boldsymbol{s} \rangle$ since it is not possible to reach $\boldsymbol{s}$ from any other maximum of the pdf without going through nodes of values lower than $\rho(\boldsymbol{s})$. Hence, any path originated in $\mathscr{P}$ will have a greater value since the remaining trivial paths are initialized with a value $-\infty$. Considering all possible paths from $\mathscr{P}$ to all samples $\boldsymbol{t} \notin \mathscr{P}$, the optimum path $\mathscr{O}^*(\boldsymbol{t})$ will be the one whose lowest density value is the maximum.

However, in a practical situation, a maximum may be represented by a set of nodes leading us to change the connectivity function in such way that an initial value h defines the relevant pdf maxima (i.e., a single node is selected per maximum). By applying $f_1(\langle \boldsymbol{s} \rangle) = h(\boldsymbol{s}) < \rho(\boldsymbol{s}), \forall \boldsymbol{s} \in \mathscr{Z}_1$, a few pdf maxima are preserved and others are reached by paths from other maxima, whose values are greater than their initial ones. Given that $h(\boldsymbol{s})$ can be computed as follows:

$$\begin{aligned} h(\boldsymbol{t}) &= \rho(\boldsymbol{t}) - \delta \\ \delta &= \min_{(\boldsymbol{s},\boldsymbol{t}) \in \mathscr{A} | \rho(\boldsymbol{t}) \neq \rho(\boldsymbol{s})} |\rho(\boldsymbol{t}) - \rho(\boldsymbol{s})|, \end{aligned} \tag{2.7}$$

all maxima of ρ are preserved and the regions with a value lower than δ will not define influence zones. According to Rocha et al. (ROCHA; FALCÃO; MELONI, 2008), the number of maxima can be reduced by:

- increasing the value of δ or by computing an anti-extensive morphological operation: the plateaus of height below δ are removed;
- applying connected filters, such as area and volume openings as anti-extensive operators: the plateaus of area or volume below δ are removed.

As aforementioned, it is desired to avoid any division of the influence zone of a maximum into multiple influence zones each rooted by one sample from that maximum. In this sense, $\mathscr{P}$ is built by adding to it a single node per maximum, which is the first one detected by OPF$_{knn}$ in each maximum. The nodes in $\mathscr{P}$ will have their cost exchanged from $h(\boldsymbol{s})$ to $\rho(\boldsymbol{s})$ and they will be able to conquer the remaining ones in their maximum zone as well. Thus, the final connectivity function f_2 is given by

$$\begin{aligned} f_2(\langle \boldsymbol{s} \rangle) &= \begin{cases} \rho(\boldsymbol{s}) & \text{if } \boldsymbol{s} \in \mathscr{P} \\ h(\boldsymbol{s}) & \text{otherwise} \end{cases} \\ f_2(\pi_{\boldsymbol{s}} \cdot \langle \boldsymbol{s}, \boldsymbol{t} \rangle) &= \min\{f(\pi_{\boldsymbol{s}}), \rho(\boldsymbol{t})\}, \end{aligned} \tag{2.8}$$

and training can be performed.

The OPF$_{knn}$ accuracy is influenced by the value $\boldsymbol{k}$ for $\mathscr{A}_k$. Papa et al. (PAPA et al.,) proposed an additional step prior the final training that computes the best value $k^* \in [1, k_{\max}]$ that maximizes the accuracy Acc over an evaluating set $\mathscr{Z}_2$. The idea is to obtain more relevant samples to the training by swapping samples from $\mathscr{Z}_2$ that were incorrectly classified for any randomly selected samples from $\mathscr{Z}_1$, except the prototypes, and minimize the misclassification rate. The accuracy Acc takes into account unbalanced datasets (i.e., a dataset whose classes are comprised of a different amount of samples) and defined as follows:

$$Acc = \frac{2c - \sum_{i=1}^{c} E(i)}{2c} = 1 - \frac{\sum_{i=1}^{c} E(i)}{2c}, \tag{2.9}$$

where i stands for the class, c is the number of classes, and the term $E(i)$ is given by:

$$E(i) = \frac{FP(i)}{|\mathscr{Z}_2| - |\mathscr{Z}_2(i)|} + \frac{FN(i)}{|\mathscr{Z}_2(i)|}, \text{ i = 1, 2, \dots, c}, \tag{2.10}$$

where $FP(i)$ and $FN(i)$ are the values of false positive and false negative for class i, respectively, and $\mathscr{Z}_2(i)$ stands for the number of samples in class i. In summary, it is computed a classification model for each $k \in [1, k_{\max}]$ with accuracy evaluated over $\mathscr{Z}_2$. Then, the model that achieves the highest accuracy is used to classify $\mathscr{Z}_3$.

The resulting optimum-path forest from $\mathscr{Z}_1$ must have each class represented by at least one maximum of the pdf. However, such condition may not be satisfied if Equation 2.8 is applied. To ensure such property, k^* is computed using Equation 2.8 and then, the final training using the best model is performed through the training Algorithm 3 applying $f_{\min}$:

$$f_{\min}(\langle \boldsymbol{s} \rangle) = \begin{cases} \rho(\boldsymbol{s}) & \text{if } \boldsymbol{s} \in \mathscr{P} \\ h(\boldsymbol{s}) & \text{otherwise,} \end{cases}$$

$$f_{\min}(\pi_{\boldsymbol{s}} \cdot \langle \boldsymbol{s}, \boldsymbol{t} \rangle) \;=\; \begin{cases} -\infty & \text{if } \lambda(\boldsymbol{t}) \neq \lambda(\boldsymbol{s}) \\ \min\{f_2(\pi_{\boldsymbol{s}}), \rho(\boldsymbol{s})\} & \text{otherwise,} \end{cases} \tag{2.11}$$

By assigning $-\infty$ to all edges $(\boldsymbol{s}, \boldsymbol{t}) \in \mathscr{A}_k$ where $\lambda(\boldsymbol{t}) \neq \lambda(\boldsymbol{s})$, we avoid such arcs to be part of an optimum path.

Concerning the training algorithm, the loop defined in Line 1 initializes the trivial paths and predecessor map, and add all nodes to the priority queue $\mathscr{Q}$. At first, the trivial paths have assigned a cost $f(\langle \boldsymbol{s} \rangle) = \rho(\boldsymbol{s}) - \delta$, and none of them have a predecessor node. Differently from OPF$_{cg}$, the priority queue in OPF$_{knn}$ is in decreasing order, being the nodes of higher cost the ones with higher priority. The main loop that begins in Line 4 iterates over all nodes that will try to conquer its adjacent ones (i.e., loop in Line 8). If the node $\boldsymbol{t}$ is conquered by a node $\boldsymbol{s}$, $\boldsymbol{s}$ becomes the predecessor of $\boldsymbol{t}$, and $\boldsymbol{s}$ assigns its label to node $\boldsymbol{t}$. The cost and position of node $\boldsymbol{t}$ in $\mathscr{Q}$ are also updated. Notice the maxima are defined in Line 6. The outcomes of the algorithm are the optimum-path forest (i.e., predecessor map), path cost map, and label map.

Algorithm 3: Supervised training using k-nn graph

```
Input: A k-nn graph $(\mathscr{Z}_1, \mathscr{A}_k)$, $\lambda(\boldsymbol{s})$ for all $\boldsymbol{s} \in \mathscr{Z}_1$ and a path-cost function $f_1$.
Output: Predecessor map $\mathscr{O}$, path cost map $\mathscr{C}$ and label map $\mathscr{L}$.
Auxiliary: Priority queue $\mathscr{Q}$ and variable $cst$.
1  for all $\boldsymbol{s} \in \mathscr{Z}_1$ do
2      $\mathscr{O}(\boldsymbol{s}) \leftarrow nil$, $\mathscr{C}(\boldsymbol{s}) \leftarrow \rho(\boldsymbol{s}) - \delta$, $\mathscr{L}(\boldsymbol{s}) \leftarrow \lambda(\boldsymbol{s})$;
3      Insert $\boldsymbol{s}$ in $\mathscr{Q}$;
4  while $\mathscr{Q} \neq \{\}$ do
5      Remove from $\mathscr{Q}$ a sample $\boldsymbol{s}$ such that $\mathscr{C}(\boldsymbol{s})$ is maximum;
6      if $\mathscr{O}(\boldsymbol{s}) = nil$ then
7          $\mathscr{C}(\boldsymbol{s}) \leftarrow \rho(\boldsymbol{s})$;
8      for each sample $\boldsymbol{t} \in \mathscr{A}_k(\boldsymbol{s})$ and $\mathscr{C}(\boldsymbol{t}) < \mathscr{C}(\boldsymbol{s})$ do
9          $cst \leftarrow \min\{\mathscr{C}(\boldsymbol{s}), \rho(\boldsymbol{t})\}$;
10         if $cst > \mathscr{C}(\boldsymbol{t})$ then
11             $\mathscr{L}(\boldsymbol{t}) \leftarrow \mathscr{L}(\boldsymbol{s})$, $\mathscr{O}(\boldsymbol{t}) \leftarrow \boldsymbol{s}$, $\mathscr{C}(\boldsymbol{t}) \leftarrow cst$;
12             Update the position of $\boldsymbol{t}$ in $\mathscr{Q}$;
13 return $\mathscr{O}, \mathscr{C}, \mathscr{L}$;
```

The classification of samples in $\mathscr{Z}_3$ is performed similarly to the conquering process (Figure 2.5). The first step computes the k-nearest neighbors of t and then, it is verified which node $\boldsymbol{s}^* \in \mathscr{Z}_1$ satisfies the equation below:

$$C(\boldsymbol{t}) = \arg\max_{\boldsymbol{s} \in \mathscr{Z}_1} \min\{C(\boldsymbol{s}), \rho(\boldsymbol{t})\}. \tag{2.12}$$

Therefore, OPF$_{knn}$ can be understood as a dual version of OPF$_{cg}$ (minimization problem) since it aims at maximizing the cost of each sample, as follows:

$$\max f(\pi_{\mathbf{s}}),\ \forall \mathbf{s} \in \mathscr{Z}_1. \tag{2.13}$$

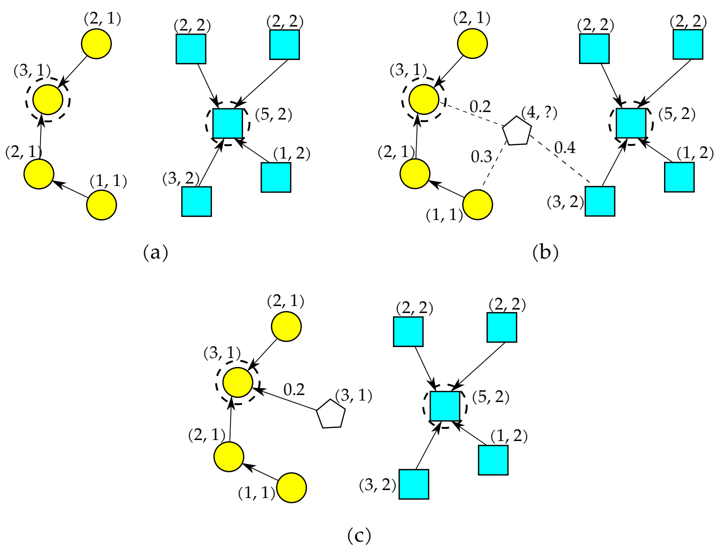


Figure 2.5: OPF k-nn: (a) optimum-path forest after training, (b) sample to be classified, and (c) sample after classification

.

2.3 Unsupervised learning

The unsupervised version of the Optimum-Path Forest classifier was proposed by Rocha et al. (ROCHA; FALCÃO; MELONI, 2008) based on the well-known mean-shift algorithm that computes clusters as influence zones of the maxima of a probability density function. The unsupervised OPF (OPF$_{uns}$) also exploits the identification of natural groups in a dataset using the pdf of the samples, but with the advantages of being less sensitive to the pdf's gradient estimation and finding clusters more closely to the desired amount.

OPF$_{uns}$ works similarly to OPF$_{knn}$ by performing two major steps: (i) defining the set of prototypes $\mathscr{P}$, and (ii) computing the influence zones (i.e., clusters), which are rooted in $\mathscr{P}$. The maxima of a pdf can be implicitly found by computing the density of each node through Equation 2.4 for a neighborhood defined by a k-NN adjacency relation. As mentioned in Section 2.2.2, a maxima of the pdf may be comprised of multiple nodes and many of them are irrelevant for clustering (i.e., over-clustering issue rises if all nodes are taken as maxima). Hence, the filter defined in Equation 2.8 is applied to obtain the relevant maxima that will compose $\mathscr{P}$, and a distinct label is assigned to each maximum.

The next step is to define the influence zones through Algorithm 4. The relevant maxima are computed on-the-fly through Lines 6 and 7, where a new label is assigned to the sample $\boldsymbol{s}$ that does not have a predecessor, and its cost changed to $\rho(\boldsymbol{s})$. By updating the cost, we allow that the first sample popped out from the maximum propagates its influence zone throughout its maximum. The label propagation is performed in the loop in Lines 8 – 12 where a samples $\boldsymbol{s}$ tries to conquer its neighbors. The outputs are an optimum-path cost map $\mathscr{V}$ and a predecessor map $\mathscr{P}$.

Algorithm 4: Clustering by OPF

Input: Graph $(\mathscr{Z}, \mathscr{A}_k)$ and functions h and ρ, such that $h(\boldsymbol{s}) < \rho(\boldsymbol{s}), \forall \boldsymbol{s} \in \mathscr{Z}$.
Output: Predecessor map $\mathscr{O}$, path-value map $\mathscr{C}$ and label map $\mathscr{L}$.
Auxiliary: Priority queue $\mathscr{Q}$, variables tmp and $l \leftarrow 1$.

```
1  for all $\boldsymbol{s} \in \mathscr{Z}$ do
2      $\mathscr{O}(\boldsymbol{s}) \leftarrow nil, \mathscr{C}(\boldsymbol{s}) \leftarrow \rho(\boldsymbol{s}) - \delta$;
3      Insert $\boldsymbol{s}$ in $\mathscr{Q}$;
4  while $\mathscr{Q} \neq \{\}$ do
5      Remove from $\mathscr{Q}$ a sample $\boldsymbol{s}$ such that $\mathscr{C}(\boldsymbol{s})$ is maximum;
6      if $\mathscr{O}(\boldsymbol{s}) = nil$ then
7          $\mathscr{L}(\boldsymbol{s}) \leftarrow l, l \leftarrow l+1, \mathscr{C}(\boldsymbol{s}) \leftarrow \rho(\boldsymbol{s})$;
8      for each sample $\boldsymbol{t} \in \mathscr{A}_k(\boldsymbol{s})$ and $\mathscr{C}(\boldsymbol{t}) < \mathscr{C}(\boldsymbol{s})$ do
9          $tmp \leftarrow \min\{\mathscr{C}(\boldsymbol{s}), \rho(\boldsymbol{t})\}$;
10         if $tmp > \mathscr{C}(\boldsymbol{t})$ then
11             $\mathscr{L}(\boldsymbol{t}) \leftarrow \mathscr{L}(\boldsymbol{s}), \mathscr{O}(\boldsymbol{t}) \leftarrow \boldsymbol{s}, \mathscr{C}(\boldsymbol{t}) \leftarrow tmp$;
12             Update the position of $\boldsymbol{t}$ in $\mathscr{Q}$;
13 return $\mathscr{O}, \mathscr{C}, \mathscr{L}$;
```

Similarly as in OPF$_{knn}$, the value of k also has some influence over the final model in the OPF$_{uns}$ version. Therefore, Rocha et al. (ROCHA; FALCÃO; MELONI, 2008) proposed finding the best value $k^* \in [1, k_{\max}]$ by computing the minimum graph cut provided by clustering results for $k \in [1, k_{max}]$ according to a measurement suggested by Shi and Malik based on graph cuts (SHI; MALIK, 2000).

Chapter 3

Pattern Analysis in Drilling Reports using Optimum-Path Forest

This chapter introduces the Optimum-Path Forest for text mining and event classification in drilling reports. The work proposed by Sousa et al. (SOUSA et al., 2018) was accepted for presentation at the International Joint Conference on Neural Networks (IJCNN), 2018 (Qualis-CC B2).

3.1 Introduction

Automatically classifying text documents has been actively pursued in the last decades mainly due to the massive amount of text that is generated and consumed daily. Applications vary from World Wide Web, social media message, electronic mail, medical patient records, and digital libraries recommendation. Despite all these seemingly distinct purposes, text learning algorithms rely on learning text patterns that may be used to automatically classifying text tasks, such as the reading interests of users, catalog news articles, and social media sentiment analysis.

Drilling activities in the oil and gas industry are of great concern from energy companies, government agencies, and the general public since they affect the economy and have a strong environmental appealing. To fulfill certain regulations and also to monitor drilling activities, the energy companies standardized some operations that are conducted on both off-shore and in-land platforms. The so-called "drilling reports" are required to be filled out to maintain a log from the whole drilling process, thus generating tons of data from every new well that is going to be drilled. Another problem concerns the subjectivity when filling the reports out since some parts of the form are free-text, i.e., the users can type in using an informal vocabulary. Such a thing turns out to be a critical issue for the application of text mining approaches, which makes the problem of learning text patterns in drilling reports worth researching.

Rassenfoss (RASSENFOSS, 2015) highlighted the importance of learning patterns from drilling logs and also reported the benefits obtained by two well-known private energy companies to discover and predict faults during the drilling process. Dawson and Verkuil (DAWSON; VERKUIL, 2014) proposed a concept of a data management system oriented to the problem of drilling monitoring. The authors spotted the need for a pattern that must be followed when filling the forms out. Antoniak et al. (ANTONIAK et al., 2016) presented a system that makes use of natural language processing (NLP) techniques to mine textual data from drilling reports, thus generating a repository to retrieve the data further. The main idea is to use data from the repository to evaluate risks during the drilling operation and to query for old reports. With such information beforehand, the drilling engineer can predict similar events that may occur.

Sidahmed et al. (SIDAHMED; COLEY; SHIRZADI, 2015) proposed an approach for concept extraction together with their frequency from drilling reports. Further, the identified concepts are then used to point out the cause and main factors that lead to some specific events occur. The authors used data from well logs provided by a private oil-and-gas company. Later on, Priyadarshy et al. (PRIYADARSHY et al., 2017) presented a methodology to extract concepts from text data that are strongly related to some abnormalities, as well as their respective symptoms, events, and further actions. The data extracted from the reports are used to train Support Vector Machines and Naïve Bayes classifiers.

Esmael et al. (ESMAEL et al., 2012) proposed an approach to organize and extract concepts from daily reports. An experiment was conducted using Support Vector Machines and Neural Networks to identify automatically usual and non-usual events in the context of drilling activities. Guilherme et al. (GUILHERME et al., 2016) proposed an ontology-based system for text mining in drilling reports. Given the knowledge of some typical problems that may occur during drilling, the authors created a series of ontologies that can describe them and further used together with some automated text analysis tool. Hoffimann et al. (HOFFIMANN et al., 2017) presented an approach based on NLP and deep learning for text mining in drilling reports. The proposed work used Convolutional Neural Networks and Long-Short Term Memory-based architectures to fulfill its central purposes, as well as standard neural nets were also used for comparison purposes. The authors claimed that Long-Short Term Memory nets were able to achieve the best results for the classification of sentences with nearly 83% of recognition rates.

Apart from text-based mining techniques, some works have also considered using computer vision to monitor the drilling process. Guilherme et al. (GUILHERME et al., 2011), for instance, proposed to track possible problems during the drilling process by taking into account the volume of cuttings coming up at the vibrating shale shaker. With such information at hands, the system could warn the staff whenever some considerable abnormality related to the volume of cuttings has occurred (e.g., if the driller machine is working at full power, but the volume of cutting is far below the expected, some problem may be happening). Guilherme et al. (GUILHERME et al., 2010) also introduced a relatively new pattern recognition technique called Optimum-Path Forest (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012) to the aforementioned context, with results pretty much competitive to some state-of-the-art techniques.

The OPF classifier has obtained competitive results and outperformed some state-of-the-art techniques in some applications, but it has never been applied to the context of text mining and event classification in drilling reports so far, which turns out to be the main contribution of this paper. In this work, an "event" stands for the different actions that may occur during drilling, such as cementing, choke drill, and the drilling operation itself, among others.

We showed results that are competitive to some of the most used pattern recognition techniques up to date, and we provided future directions towards the area of text mining in drilling reports. The remainder of this paper is organized as follows. Section 3.2 presents the proposed approach and the methodology employed in this work. Section 3.3 discusses the experiments, and Section 3.4 states conclusions and future works.

3.2 Proposed Approach and Methodology

In this section, we present the proposed approach for automatic text classification in drilling reports using the Optimum-Path Forest classifier. In this approach, the input is a set of drilling reports containing a sequence of events that are described in free-text fields. The first step in the workflow aims at looking for "sentences", which are defined as a sequence of letters/digits bounded by a period character. However, such characters may correspond to other meanings than ending a phrase, e.g., abbreviations or representing decimal numbers. In this situation, we analyze the neighborhood of such characters to decide whether they stand for the end of a sentence or not.

Further, the sentences must be pre-processed as follows: firstly, numbers are identified and replaced by a special token (e.g., "speed 0.5m/s" is replaced by "speed [[number 0.5]]m/s"). Later on, each word belonging to a sentence (i.e., a term) is segmented using a regular expression that identifies terms that are composed of alphanumeric characters, followed by special tokens (e.g., '[' or ']'), and then a non-blank space. Next, stopwords are removed according to a pre-defined list that includes punctuation, articles, and prepositions of the Portuguese language (we are dealing with drilling reports from a Brazilian oil and gas company). Finally, it is applied the RSLP (*Removedor de Sufixos da Língua Portuguesa*) technique (ORENGO; BURIOL; COELHO, 2007), which is a rule-based suffix-stripping algorithm that is used to stem the remaining terms.

Last but not least, the pre-processed free-text data are described by different feature extractors. This work evaluated four distinct representation models: Term Frequency - Inverse Document Frequency (TF-IDF) (SALTON; MCGILL, 1986) , Textual Language Model Jelinek-Mercer (JELINEK; MERCER, 1980), Best Matching (BM)-25 (JONES; WALKER; ROBERTSON, 2000) , and fusion of TF-IDF and BM-25. Figure 3.1 depicts the workflow of the proposed approach.

Additionally, we evaluated the robustness of the OPF classifier under higher-dimensional spaces. To fulfill that purpose, we also employed TF-IDF to extract features from either terms and bigrams, as well as a combination of both. Since these new scenarios pose a more significant challenge considering higher dimensional spaces, we examined the influence of dimensionality reduction techniques concerning sentence identification using Latent Semantic Analysis (LSA) (WIEMER-HASTINGS; WIEMER-HASTINGS; GRAESSER, 2004) and Non-negative Matrix Factorization (NMF) (PAATERO; TAPPER, 1994) .

The experiments used a dataset containing 12 drilling reports covering a total of $6,667$ operations (sentences) labeled by specialists in the field of petroleum engineering, involving 43 classes (i.e., actions). For comparison purposes, OPF is compared against the well-known Naïve Bayes (BC) and k-NN classifiers using a k-fold-like cross-validation with 12 folds (i.e., each fold representing the collection of sentences extracted from a drilling report).

Regarding the source-codes, we used LibOPF (PAPA; SUZUKI; FALCÃO, 2014) concerning OPF classifier, and our implementation for both k-NN and Naïve Bayes classifiers. Concerning k-NN, we chose the value of $k \in \{1, 100\}$ that maximized the accuracy over the training set. Notice that range was set up empirically.

3.3 Experimental Section

In this section, we present the experimental results concerning the proposed approach for automatic event classification in drilling reports[1]. As aforementioned, we divided the experiment into two rounds: (i) the first one aims at evaluating different representation models (i.e.,

[1]All experiments were performed in a computer equipped with a Xeon® Bronze 3106 processor of 1.70GHz and 64GB of RAM.

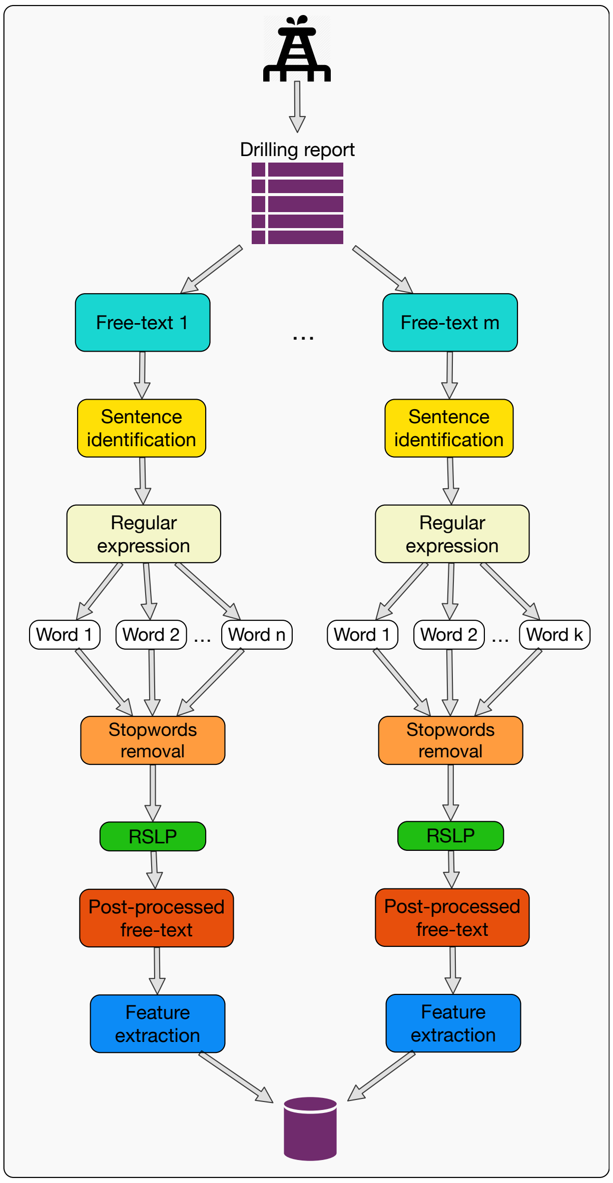


Figure 3.1: Workflow of the proposed approach for automatic event classification in drilling reports.

feature extractors), and (ii) the second assessed the robustness of the classifiers under reduced feature spaces. To provide a statistical evaluation, we employed the Wilcoxon signed-rank test (WILCOXON, 1945) with significance as of 5%.

3.3.1 Evaluating the representation models

Tables 3.1- 3.3 present the recognition rates concerning TF-IDF, Jelinek-Mercer, and BM-25, respectively. The best techniques according to the Wilcoxon test are highlighted in bold. As mentioned earlier, we performed a 12-fold cross-validation approach, where each drilling report extracted from a certain well is used for testing purposes, and the remaining 11 reports are employed for training the model.

Table 3.1: Experimental results using TF-IDF.

Well	BC	*k*-NN	OPF
W1	79.46%	80.32%	80.33%
W2	80.11%	80.07%	79.43%
W3	80.89%	81.11%	80.79%
W4	80.69%	80.89%	80.93%
W5	79.31%	80.92%	79.10%
W6	83.08%	83.06%	82.65%
W7	79.89%	79.95%	79.74%
W8	75.90%	77.06%	75.49%
W9	80.88%	81.62%	80.57%
W10	85.25%	87.66%	87.22%
W11	86.49%	88.40%	88.38%
W12	76.89%	77.03%	77.01%
Average	**81.57%±2.97**	**81.51%±3.50**	**80.97%±3.70**

In a nutshell, one can draw three main conclusions: (i) all techniques obtained similar recognition rates across all representation models, (ii) TF-IDF played the major role in the results, with the highest recognition rates, and (iii) the proposed approach generalize reasonably among the wells since one has small standard deviation values.

Table 3.2: Experimental results using Textual Language Model Jelinek-Mercer.

Well	BC	*k*-NN	OPF
W1	61.39%	63.51%	63.13%
W2	64.92%	66.36%	63.74%
W3	66.47%	67.52%	67.52%
W4	60.13%	62.95%	62.95%
W5	59.71%	59.83%	59.24%
W6	59.25%	59.58%	59.58%
W7	70.04%	69.85%	68.54%
W8	66.55%	66.55%	67.08%
W9	68.27%	69.33%	67.20%
W10	75.98%	78.35%	76.77%
W11	75.08%	77.00%	74.76%
W12	58.85%	58.25%	55.47%
Average	65.55%±5.99	**65.59%±6.42**	65.50%±6.19

TF-IDF technique has provided the best results among all representation models. Although *k*-NN obtained the best results concerning BM-25 and Jelinek-Mercer, these approaches did not achieve reasonable results. Additionally, the statistical test pointed out that BM-25 outperformed Jelinek-Mercer by a small margin. In short, TF-IDF works by measuring how much information a word provides but considering those who appear less as the most informative ones. In the context of data mining in drilling reports, one may face two challenging situations: (i) free-texts with more than one event (usually separated in sentences), and, very much often, (ii) free-text fields with a few sentences only. On the other hand, some events are readily characterized by some specific verbs, thus helping TF-IDF obtaining good results.

Table 3.3: Experimental results using BM-25.

Well	**BC**	***k*-NN**	**OPF**
W1	61.97%	63.90%	63.32%
W2	67.02%	68.06%	65.58%
W3	66.02%	67.22%	67.22%
W4	61.48%	64.43%	62.82%
W5	60.67%	60.79%	58.76%
W6	61.04%	61.36%	60.88%
W7	71.54%	71.16%	68.73%
W8	66.55%	66.73%	63.75%
W9	71.20%	72.27%	69.60%
W10	75.59%	77.95%	76.38%
W11	75.08%	77.00%	74.44%
W12	60.64%	60.24%	59.24%
Average	66.57%±5.63	**67.59%±5.99**	65.89%±5.62

Figures 3.2 and 3.3 depict the training and classification times (logarithmic scale) concerning TF-IDF representation model, respectively[2]. A significant advantage concerning OPF is related to its training phase, which is parameterless. Although *k*-NN does not figure a training step either, we considered the fine-tuning parameter phase in the training time computation.

[2]We considered only TF-IDF since it has obtained the best results, and a similar behavior can be observed for Jelinek-Mercer and BM-25 models.

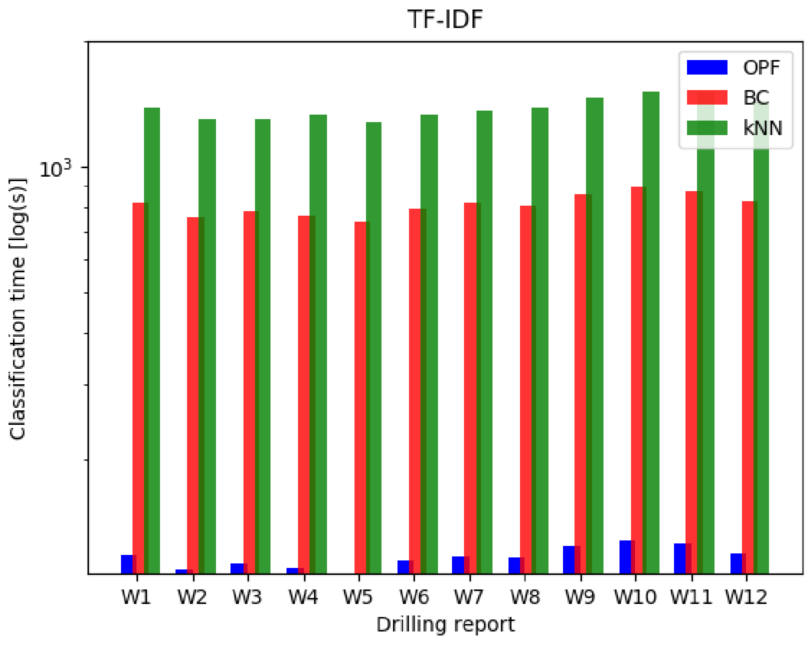


Figure 3.2: Computational load concerning the training step.

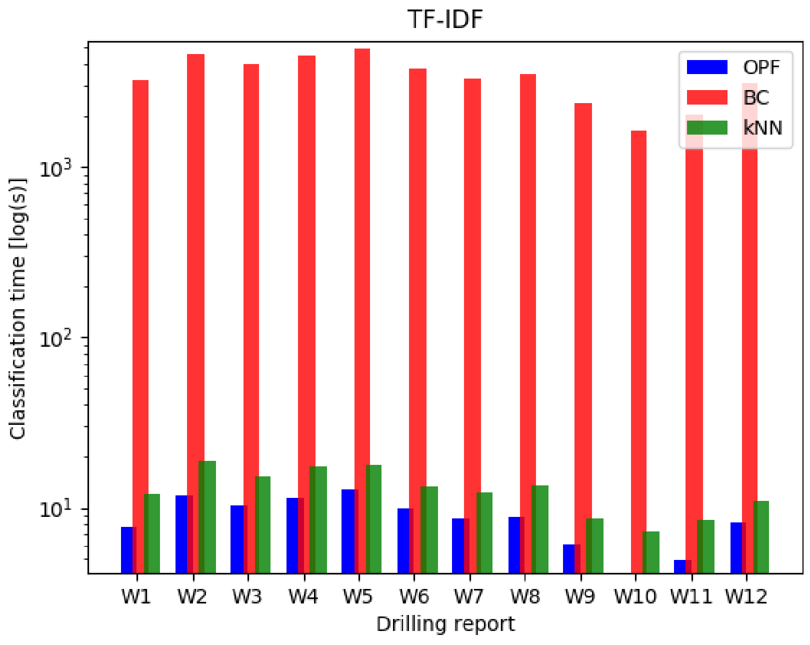


Figure 3.3: Computational load concerning the classification time.

Concerning the classification step, one can realize that OPF is also faster than *k*-NN for classification since the latter needs to compute the distance from the testing sample to the entire training set to find out the *k*-nearest neighbors. An optimization approach proposed by Papa et al. (PAPA et al., 2012) makes use of a list of training samples ordered by their costs, which turned out to be helpful when computing the distances from the testing sample to the training ones. In this approach, there is no need to run over all training samples to perform the competition process.

We performed an additional experiment to compare the usage of bigrams against the terms, as well as a combination of both using concatenations. This further investigation was accomplished over the TF-IDF representation model only since it has obtained the best results. Table 3.4 presents the average results concerning the bigrams, which were slightly below the ones stated in Table 3.1. Since we have a few small sentences per free-text, bigrams contribute to diminishing even more such amount of information.

Table 3.4: Experimental results using TF-IDF over bigrams.

Well	**BC**	***k*NN**	**OPF**
W1	64.73%	78.90%	64.84%
W2	58.12%	77.90%	59.75%
W3	60.88%	75.44%	60.83%
W4	60.21%	81.74%	59.90%
W5	75.76%	77.41%	75.73%
W6	57.31%	76.26%	57.31%
W7	61.03%	80.36%	60.75%
W8	56.93%	73.18%	56.95%
W9	60.21%	78.28%	60.85%
W10	61.51%	90.69%	63.91%
W11	64.39%	88.47%	66.07%
W12	56.67%	77.24%	56.73%
Average	61.48%±5.22	**80.04%±5.22**	61.97%±5.27

Table 3.5 presents the average results concerning the combination of both bigrams and terms, with results slightly better than the ones obtained over bigrams only (Table 3.4). The benefits of the concatenation are clearer noticed concerning the OPF and BC classifiers. Since *k*-NN is purely based on distance computations, increasing the feature space may neglect subtle information encoded in the feature vectors, thus not contributing (or sometimes making it worse) to better results.

Table 3.5: Experimental results using TF-IDF and a combination of terms and bigrams.

Well	**BC**	***k*NN**	**OPF**
W1	79.18%	80.75%	80.46%
W2	78.77%	79.67%	79.06%
W3	78.27%	78.48%	77.95%
W4	80.96%	81.57%	81.26%
W5	78.53%	80.18%	78.37%
W6	78.25%	78.20%	77.74%
W7	82.21%	82.31%	81.97%
W8	78.76%	79.96%	78.35%
W9	77.93%	78.78%	78.62%
W10	85.62%	88.20%	86.41%
W11	84.23%	86.23%	83.81%
W12	73.82%	74.45%	74.33%
Average	79.71%±3.14	**80.73%±3.64**	79.86%±3.17

3.3.2 Dimensionality reduction

The second round of experiments aimed at assessing the impact of lower-dimensional feature spaces on the accuracy rate of the classification techniques. Usually, text-oriented data mining makes use of large vocabularies, increasing the dimensionality of the feature vectors used to represent each sentence.

In this round of experiments, we compared two approaches for dimensionality reduction in the context of event classification in drilling reports. The effect of reduced dimensionality was evaluated in the distinct scenarios: (i) terms (i.e., the same used in the previous experiment), (ii) bigrams, and (iii) a combination (i.e., concatenation) of both. Since TF-IDF obtained the best results in the previous experiments, we focused on this approach only. Notice that the dimensionality was empirically reduced to 100 for all cases.

Table 3.6 presents the dimensionality reduction results using LSA over the terms. The results were somehow similar among the techniques, being BC and *k*-NN the most accurate techniques, closely followed by OPF. One can observe that the classifiers did not benefit from the dimensionality reduction, which indicates the good choice for the vocabulary's size employed in the previous experiment.

Table 3.6: Dimensionality reduction using Latent Semantic Analysis with terms.

Well	BC	*k*NN	OPF
W1	65.25%	66.22%	64.48%
W2	62.17%	61.91%	61.65%
W3	67.52%	67.67%	66.92%
W4	67.11%	66.98%	65.64%
W5	62.34%	62.34%	61.86%
W6	62.98%	62.99%	62.01%
W7	67.60%	67.42%	66.67%
W8	70.40%	70.23%	69.88%
W9	68.00%	68.80%	67.73%
W10	71.26%	73.62%	72.44%
W11	67.73%	68.05%	67.09%
W12	60.34%	60.64%	60.44%
Average	**66.06%±3.44**	**66.41%±3.81**	65.57%±3.63

Figures 3.4 and 3.5 depict the computational load concerning dimensionality reduction using LSA and TF-IDF considering the training and testing phases, respectively. Once again, OPF has been the fastest technique for training, and with similar efficiency when compared to *k*-NN since now we have reduced dimensional feature spaces, which means *k*-NN takes lower for distance computation.

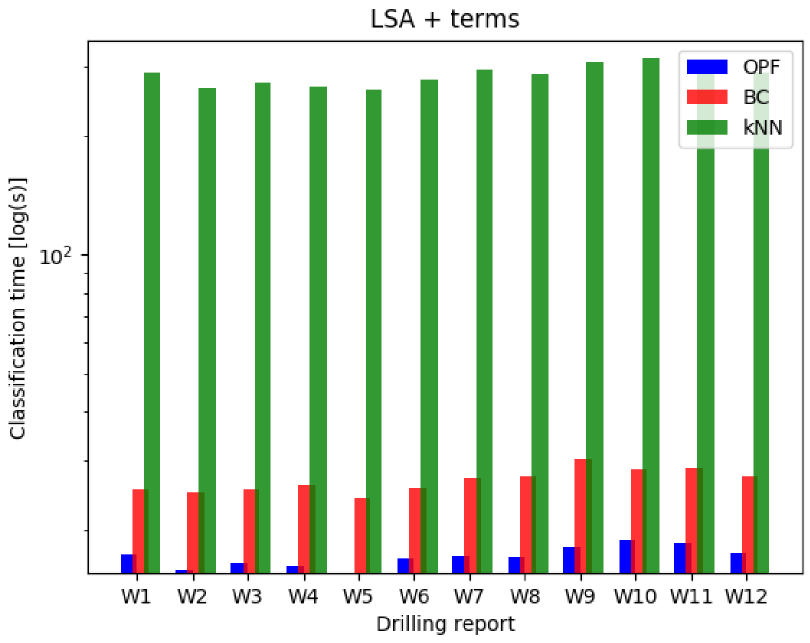


Figure 3.4: Computational load concerning the training step.

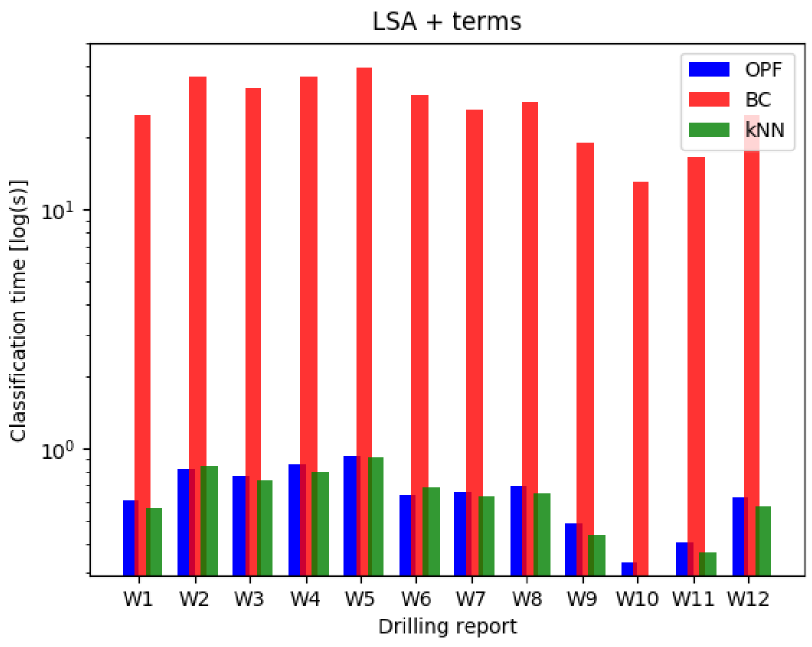


Figure 3.5: Computational load concerning the testing step.

Table 3.7 presents the dimensionality reduction experiment using LSA over bigrams. Once again, reducing the size of the dictionary did not play an interesting role since the recognition rates were considerably reduced. Also, as mentioned earlier, the free-text samples do not comprise too many words, and clustering them as bigrams help to have even less information from the sentences.

Table 3.7: Dimensionality reduction using Latent Semantic Analysis with bigrams.

Well	**BC**	***k*NN**	**OPF**
W1	57.53%	58.69%	56.95%
W2	54.97%	54.84%	53.80%
W3	58.05%	58.20%	57.14%
W4	59.19%	58.79%	58.52%
W5	52.21%	52.21%	52.21%
W6	52.11%	52.44%	51.62%
W7	62.36%	61.99%	61.24%
W8	56.39%	56.39%	56.04%
W9	48.80%	49.07%	48.00%
W10	65.75%	67.72%	65.75%
W11	60.70%	61.02%	60.38%
W12	54.87%	54.67%	54.87%
Average	**56.91%±4.75**	**57.17%±5.05**	56.38%±4.79

Table 3.8 presents the results with dimensionality reduction using LSA over the combination of terms and bigrams. The results outperformed the ones with either terms or bigrams solely but did not achieve better results than the ones presented in the previous section. All classifiers obtained similar results for all situations, thus showing that different wells share similar properties, even with descriptors that do not get good recognition rates.

Table 3.8: Dimensionality reduction using Latent Semantic Analysis with terms and bigrams.

Well	**BC**	***k*NN**	**OPF**
W1	66.02%	67.18%	66.60%
W2	62.70%	62.43%	62.04%
W3	69.47%	69.92%	68.87%
W4	67.11%	66.98%	66.58%
W5	62.57%	62.57%	61.03%
W6	60.71%	60.71%	59.58%
W7	67.42%	67.04%	66.85%
W8	68.13%	67.95%	66.90%
W9	67.47%	68.00%	67.73%
W10	75.20%	76.77%	76.38%
W11	74.76%	75.40%	73.48%
W12	61.43%	61.43%	60.44%
Average	**66.92%±4.72**	**67.20%±5.11**	66.37%±5.12

Table 3.9 presents the results concerning dimensionality reduction using non-negative matrix factorization applied over the terms only. Once again, all classifiers achieved pretty much close results but with results inferior to the ones obtained with LSA and without dimensionality reduction, as stated in the previous section. Also, the results were less stable since the standard deviation values are higher than the ones obtained with LSA (Table 3.6).

Table 3.9: Dimensionality reduction using Non-negative Matrix Factorization with terms.

Well	**BC**	*k***NN**	**OPF**
W1	61.78%	62.74%	61.78%
W2	58.12%	57.72%	57.85%
W3	64.21%	64.51%	64.51%
W4	59.72%	59.73%	58.26%
W5	57.45%	57.45%	56.85%
W6	54.22%	54.06%	51.95%
W7	63.86%	63.67%	62.55%
W8	69.00%	69.00%	68.65%
W9	58.93%	59.73%	58.13%
W10	66.14%	67.72%	66.14%
W11	67.73%	68.37%	67.09%
W12	58.65%	58.65%	57.26%
Average	**61.65%±4.55**	**61.95%±4.82**	60.92%±5.01

The bigrams did not allow satisfying recognition rates once more, as one can observe in Table 3.10, which presents the recognition rates using non-negative matrix factorization over the bigrams. In this case, the results were even lower than the ones obtained with LSA. Although a combination of both bigrams and terms allowed better recognition rates than the sole approaches (Table 3.11), they are far below the results obtained without dimensionality reduction.

Table 3.10: Dimensionality reduction using Non-negative Matrix Factorization with bigrams.

Well	**BC**	*k***NN**	**OPF**
W1	51.74%	48.26%	47.68%
W2	48.30%	44.50%	47.38%
W3	47.22%	44.66%	47.37%
W4	48.59%	46.44%	48.05%
W5	46.60%	43.86%	46.25%
W6	45.13%	43.83%	44.81%
W7	55.62%	51.50%	55.06%
W8	47.29%	43.96%	47.11%
W9	40.27%	37.60%	40.53%
W10	57.87%	56.69%	57.09%
W11	60.38%	57.51%	60.70%
W12	50.70%	48.51%	51.09%
Average	**49.98%±5.68**	47.28%±5.70	**49.43%±5.64**

Table 3.11: Dimensionality reduction using Non-negative Matrix Factorization with terms and bigrams.

Well	**BC**	***k*NN**	**OPF**
W1	60.62%	61.78%	59.85%
W2	56.41%	56.15%	55.76%
W3	63.31%	63.61%	62.71%
W4	59.19%	59.19%	58.39%
W5	56.62%	56.62%	55.66%
W6	53.90%	53.90%	53.57%
W7	60.30%	60.11%	59.74%
W8	56.39%	56.39%	56.04%
W9	56.80%	57.07%	55.73%
W10	67.72%	69.29%	67.32%
W11	70.29%	70.61%	70.29%
W12	55.27%	55.86%	55.86%
Average	59.74%±5.08	**60.02%±5.40**	59.24%±5.15

3.4 Conclusions and Future Works

In this paper, we dealt with the problem of event classification in drilling report using supervised classifiers. Our primary focus was in the evaluation of the Optimum-Path Forest classifier in the context of text mining in the oil and gas industry, since it has never been applied to this application up to date.

The proposed approach was evaluated in drilling reports obtained from a Brazilian oil and gas company (Petrobras), which were previously pre-processed for the further extraction of features using three distinct representation models. Additionally, we conducted an additional experiment to evaluate whether the techniques would benefit from dimensionality or not.

We have observed that TF-IDF provided the best results so far, with BC, *k*-NN, and OPF being similar according to statistical evaluation. Also, the proposed approach seemed to generalize reasonably for different wells, with results nearly to 81% of recognition rates. Such effectiveness is quite promising given the fact we have 43 various events to be recognized, and some drilling reports may have more than one event described in it.

The OPF classifier figured as the best choice among all since it obtained results similar (or close to it) to the best ones, but with much less computational effort and with no parameters either. Concerning future works, we intend to research a different manner to estimate the weights in the standard TF-IDF approach, which does not consider the classifier's accuracy on it, but rather the frequency of terms only. We believe that designing an integrated approach that considers TF-IDF and classifiers would estimate better weights, thus favoring the effectiveness at the end.

Chapter 4

A Kernel-based Optimum-Path Forest Classifier

This chapter presents a variation of the supervised OPF classifier by incorporating kernel functions in both training and classification steps. This work was proposed by Afonso et al. (AFONSO; PEREIRA; PAPA, 2018) and was presented at the 22nd Iberoamerican Congress on Pattern Recognition 2017 (Qualis-CC A4).

4.1 Introduction

The nature of real-world problems has driven their modeling to structured objects where samples are usually represented by nodes and their relationship are represented by connections (edges), similarly to a graph (KONDOR; LAFFERTY, 2002; VISHWANATHAN et al., 2010). Among these problems, one can mention studies in bioinformatics, social networks, and data mining in documents, just to name a few, in which it is necessary to identify structures or elements that share similar properties.

Another very common characteristic present in real-world problems is the non-linear distribution, which requires complex models to identify the different existing patterns in the dataset (HOFMANN; SCHöLKOPF; SMOLA, 2008). In this matter, kernels have been proposed as a tool to overcome this issue by providing a way to map the data into a space of higher dimension, where the input data may be linearly separable.

Support Vector Machines is perhaps the most well-known technique that makes use of kernels for data embedding into higher dimensional feature spaces (BURGES, 1998). There are also the kernel-PCA (Principal Component Analysis) (SCHöLKOPF; SMOLA; MüLLER, 1998) , and the kernel-Fisher discriminant analysis (MIKA et al., 1999), just to mention a few. Although the mapping can take a feature space to a very high dimension that increases the computing cost, both training and classification steps can be performed by means of a dot product, the so-called *kernel trick*. Such procedure consists in computing the dot product without the need for mapping the samples to a higher dimensional space. The combination of these two worlds, i.e., representations based on graphs and non-linear distributions, has motivated studies in the application of kernels in structured data (KONDOR; LAFFERTY, 2002; GäRTNER, 2003). In this research area, it is typical to find two different approaches, i.e., applications that make use of kernels to identify similarity among graphs, while others focus on the similarity among the graph nodes.

One of the first works on kernels among graphs was proposed by Gärtner et al. (GÄRTNER; FLACH; WROBEL, 2003) and Borgwardt et al. (BORGWARDT et al., 2005) using the random walk, and marginalized kernels by Tsuda et al. (TSUDA; KIN; ASAI, 2002), Kashima et al. (KASHIMA; TSUDA; INOKUCHI, 2003, 2004) and Mahé et al. (MAHÉ et al., 2004). The basic idea of random-walk graph kernels is to perform a random walk in a pair of graphs and sum up the number of matching walks (VISHWANATHAN et al., 2010) on both graphs. The marginalized kernel is defined as the inner product of the count vectors averaged over all possible label paths (KASHIMA; TSUDA; INOKUCHI, 2003). Regarding kernels in graphs, interesting works were proposed by Kondor and Lafferty (KONDOR; LAFFERTY, 2002) and Smola and Kondor (SMOLA; KONDOR, 2003). In (KONDOR; LAFFERTY, 2002), the authors proposed the diffusion kernels, which is a special class of exponential kernels based on the heat equation.

Graph-based machine learning techniques can be noticed as well. The current OPF algorithm implementation works naturally with non-linear situations, but it does not map samples from one space to another. In this paper, we take one step further by modifying the OPF algorithm to work with kernels on graphs to improve its training and classification results, since such approach has not been applied so far. The performance of the proposed approach is assessed under three different kernels, and it is compared against the original OPF algorithm and the well-known SVM in 11 different datasets. The remainder of this paper is organized as follows. Section 4.2 presents the theoretical background concerning the OPF kernel-based variant. Section 4.3 discusses the methodology and experiments, and Section 4.4 states the conclusions.

4.2 Proposed Approach

The proposed kernel-based OPF, hereinafter called *k*OPF , works similarly to SVM algorithm, in which samples are mapped into a feature space of higher dimension. In the SVM context, such mapping is performed as an attempt to make the data linearly separable. The OPF, on the other hand, naturally works with non-linear data. Therefore, the main idea of this work is to evaluate OPF's behavior under such assumption of samples' separability in higher dimensional spaces.

Let $\Phi(\cdot,\cdot)$ be a kernel function that generates a new dataset $\mathscr{M} = \Phi(\mathscr{Z})$. Given a sample $\mathbf{z} \in \mathscr{Z}$, such that $\mathbf{z} \in \Re^n$, its new representation $\hat{\mathbf{z}} \in \mathscr{M}$ is defined as follows.

$$\hat{\mathbf{z}} = (\Phi_1, \Phi_2, \ldots, \Phi_{|\mathscr{Z}_1|}), \tag{4.1}$$

where $\Phi_i = (\mathbf{z}, \mathbf{s}_i)$, $\mathbf{s}_i \in \mathscr{Z}_1$. Notice that $\hat{\mathbf{z}} \in \Re^{|\mathscr{Z}_1|}$, which means the new sample $\mathbf{z}$ contains as many dimensions as the number of training samples.

In short, $\Phi(\mathbf{z}, \mathbf{s}_i)$ makes use of a distance function (i.e., Euclidean, Mahalanobis, among others) to compute a term that replaces the norm in kernel functions, such as Radial Basis Function (RBF) and Sigmoid, for instance. The aforementioned term corresponds to the distance between a sample to be mapped $\mathbf{z}$ and a training sample $\mathbf{s}$. Basically, the mapping performed by *k*OPF is carried out by computing a feature vector, where each component has the distance value from the sample to be mapped (either training or testing sample) to a different training sample applied to a kernel function. It is important to highlight that large training sets may cause a significant increase on the size of the new feature vector, since the size is dependent on the number of training samples $|\mathscr{Z}_1|$.

4.3 Methodology and Experiments

The *k*OPF classifier has both its performance and accuracy assessed by means of 11 public benchmarking datasets[1] that provide different classification scenarios. The implementation of our proposed approach is developed over the LibOPF (PAPA; SUZUKI; FALCÃO, 2014), being standard OPF and SVM used as baselines for the experiments. With respect to SVM, we used the well-known LibSVM[2]. Table 4.1 presents detailed information from the datasets.

Since the kernel function can influence the final accuracy, we evaluated its impact by ap-

[1]`https://github.com/jppbsi/LibOPF`

[2]`https://www.csie.ntu.edu.tw/~cjlin/libsvm`

Table 4.1: Information about the datasets used in the experiments.

dataset	No. samples	No. features	No. classes
boat (bt)	100	2	3
cone-torus (ct)	400	2	3
data1 (d1)	1,423	2	2
data2 (d2)	283	2	2
data3 (d3)	340	2	5
data4 (d4)	698	2	3
data5 (d5)	1,850	2	2
mpeg7-BAS (m-B)	1,400	180	70
mpeg7-Fourier (m-F)	1,400	126	70
petals (ps)	100	2	4
saturn (sn)	200	2	2

plying three different kernel functions for *k*OPF as follows:

- Identity: $\Phi(\mathbf{z},\mathbf{s}) = ||\mathbf{z},\mathbf{s}||$
- RBF: $\Phi(\mathbf{z},\mathbf{s}) = e^{-(\gamma||\mathbf{z},\mathbf{s}||^2)}$
- Sigmoid: $\Phi(\mathbf{z},\mathbf{s}) = \tanh(\gamma||\mathbf{z},\mathbf{s}|| + C)$

where $||\mathbf{z},\mathbf{s}||$ denotes the Euclidean distance. Notice the Identity kernel is parameterless. The SVM was also evaluated using three different kernel functions: linear, RBF and Sigmoid. The situations in which parameters C and γ are required, it is performed their optimization using the intervals $C = [-32, 32]$ and $\gamma = [0, 32]$ with steps equals to 2 for both of them.

The classification experiments were conducted by means of a hold-out process using 15 runs, in which both training and testing sets were randomly generated in each run and always having a number of samples equals to 50% of the dataset size. The experiments also evaluated the impact on the accuracy rate when features are normalized. Tables 4.2 and 4.3 present the mean accuracy results considering non-normalized and normalized datasets, respectively. The accuracy rates were computed using the accuracy measure proposed by Papa et al. (PAPA; FALCÃO; SUZUKI, 2009), which considers unbalanced data. The best results according to the Wilcoxon signed-rank test with significance 0.05 are shown in bold.

In the non-normalized feature scenario, SVM-RBF achieved the best results (or similar) in 9 out of 11 datasets, followed by the *k*OPF (*k*OPF-Identity and *k*OPF-RBF) with 7 out of 11 (being *k*OPF-Identity the best in 6 out of 11). The standard OPF obtained the best results in 5 out of 11 datasets. Considering normalized features, the *k*OPF (*k*OPF-Identity, *k*OPF-RBF and *k*OPF-Sigmoid) obtained the best results (or similar) in 8 out of 11 datasets (being

Table 4.2: Mean classification rates for non-normalized features.

dataset	OPF	*k*OPF			SVM		
		Identity	RBF	Sigmoid	Linear	RBF	Sigmoid
bt	98.5±0.0	96.8±3.0	**100.0±0.0**	99.1±1.2	76.9±1.8	**100.0±0.0**	76.9±1.8
ct	86.2±0.0	84.3±2.6	84.5±2.7	84.2±2.4	74.9±0.2	**86.5±1.1**	74.4±0.4
d1	99.5±0.0	**99.4±0.2**	67.2±8.6	68.5±14.5	94.8±0.8	**99.4±0.3**	94.0±0.8
d2	**99.3±0.0**	98.1±1.1	57.0±2.6	62.0±6.1	97.1±0.5	98.2±0.2	81.3±3.6
d3	99.6±0.0	**99.7±0.4**	64.0±3.6	71.1±5.4	98.4±0.8	**99.7±0.4**	97.9±0.7
d4	**100.0±0.0**	**100.0±0.0**	60.8±2.6	67.8±6.4	**100.0±0.0**	**100.0±0.0**	50.9±1.3
d5	**100.0±0.0**	**100.0±0.0**	62.5±3.2	67.5±9.3	50.0±0.0	**100.0±0.0**	50.0±0.0
m-B	89.4±0.0	89.0±0.30	50.1±0.1	50.1±0.1	87.9±0.2	**90.5±0.2**	50.0±0.0
m-F	**73.0±0.0**	**73.0±0.5**	64.4±0.6	66.2±0.5	69.0±0.8	**73.0±0.5**	68.1±0.5
ps	98.7±0.0	**100.0±0.0**	99.2±0.6	98.9±1.0	99.6±0.6	**100.0±0.0**	**100.0±0.0**
sn	**93.0±0.0**	90.2±2.0	80.8±2.6	81.8±4.0	42.7±3.7	91.3±1.9	49.0±3.3

Table 4.3: Mean classification rates for normalized features.

dataset	OPF	*k*OPF			SVM		
		Identity	RBF	Sigmoid	Linear	RBF	Sigmoid
bt	97.1±0.0	99.7±0.6	99.4±1.2	**100.0±0.0**	76.9±1.8	99.5±0.7	76.5±1.2
ct	89.3±0.0	86.7±1.7	88.0±1.2	**89.9±0.5**	75.9±1.6	**89.4±0.5**	76.1±1.6
d1	**99.3±0.0**	99.0±0.3	99.0±0.3	99.0±0.3	94.7±0.4	**99.3±0.2**	94.7±0.4
d2	97.3±0.0	96.9±1.5	97.1±1.2	97.0±1.3	**98.8±0.9**	**98.6±0.6**	**98.6±0.6**
d3	**100.0±0.0**	98.4±1.4	**100.0±0.0**	98.7±0.8	99.5±0.4	99.3±0.7	99.5±0.4
d4	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**
d5	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	50.0±0.0	**100.0±0.0**	50.7±0.9
m-B	88.8±0.0	**90.8±0.3**	53.2±0.5	56.3±0.4	87.7±0.9	89.9±0.6	87.6±0.9
m-F	62.9±0.0	62.5±0.4	59.2±0.7	61.1±0.8	64.7±0.7	**65.4±0.5**	64.3±0.9
ps	98.7±0.0	**100.0±0.0**	99.5±0.6	99.5±1.0	**100.0±0.0**	99.6±0.6	**100.0±0.0**
sn	91.0±0.0	**91.8±0.2**	79.2±4.8	84.6±0.8	52.3±1.7	90.7±4.5	50.7±3.4

*k*OPF-Identity the best in 5 out of 11), followed by SVM (SVM-Linear, SVM-RBF and SVM-Sigmoid) with 7 out of 11 (being SVM-RBF the best in 6 out of 11). The OPF obtained the best results in only 4 out of 11 datasets.

In both (non-normalized and normalized) scenarios, the proposed *k*OPF outperformed the traditional OPF in most datasets, and for normalized features, *k*OPF outperformed the SVM in some datasets as well. The results are quite interesting since *k*OPF was able to improve OPF and outperforming SVM in some datasets. Considering some other datasets, although *k*OPF did not outperform both OPF and SVM, their results were considerably close.

The experiments also comprised the analysis of computational load required by each technique in each dataset. The results showed OPF and *k*OPF require a considerably small computational load in the training phases when compared against SVM. The high training time consumption turns the SVM prohibitive in real-time learning systems, especially if the training set is very dynamic over time. In this situation, both OPF and *k*OPF seems to be the most suitable approach. Tables 4.4 and 4.5 present the mean training and testing computational loads,

Table 4.4: Mean training time using 50% of the samples for training.

dataset	OPF	*k*OPF			SVM		
		Identity	RBF	Sigmoid	Linear	RBF	Sigmoid
bt	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	15.8±0.9	0.4±0.0	0.4±0.0
ct	0.0±0.0	0.02±0.0	0.0±0.0	0.0±0.0	628.6±295.9	2.24±0.2	1.3±0.0
d1	0.0±0.0	0.7±0.0	1.4±0.0	1.1±0.0	45.2±21.7	5.96±0.2	6.1±0.2
d2	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.6±0.2	0.67±0.0	0.6±0.0
d3	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.6±0.0	0.97±0.0	0.9±0.0
d4	0.0±0.0	0.1±0.0	0.2±0.0	0.1±0.0	0.6±0.0	1.74±0.0	1.7±0.0
d5	0.1±0.0	1.4±0.0	2.8±0.1	2.4±0.0	2906.1±490.7	3.7±0.1	11.6±1.2
m-B	0.2±0.0	0.3±0.0	0.6±0.0	0.6±0.0	224.7±2.8	295.2±1.0	226.8±1.1
m-F	0.1±0.0	0.7±0.0	1.0±0.0	0.9±0.0	221.9±0.7	232.9±0.5	203.5±0.5
ps	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.3±0.0	0.4±0.0	0.4±0.0
sn	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	34.8±10.3	0.9±0.1	0.7±0.01

Table 4.5: Mean testing time.

dataset	OPF	*k*OPF			SVM		
		Identity	RBF	Sigmoid	Linear	RBF	Sigmoid
bt	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
ct	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
d1	0.0±0.0	0.5±0.0	0.6±0.0	0.5±0.0	0.0±0.0	0.0±0.0	0.0±0.0
d2	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
d3	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
d4	0.0±0.0	0.1±0.0	0.1±0.0	0.1±0.0	0.0±0.0	0.0±0.0	0.0±0.0
d5	0.0±0.0	1.2±0.0	1.4±0.1	1.2±0.0	0.0±0.0	0.0±0.0	0.0±0.0
m-B	0.2±0.0	0.8±0.0	1.0±0.0	0.9±0.0	1.0±0.0	1.2±0.0	1.0±0.0
m-F	0.2±0.0	0.9±0.0	1.5±0.0	0.9±0.0	0.8±0.0	0.9±0.0	0.8±0.0
ps	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
sn	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0

respectively. Notice that the training time takes into account the fine-tuning when performed.

4.4 Conclusions

This paper introduced a kernel-based OPF, which is a modification made over the standard OPF classifier that allows the usage of different kernel functions for both learning and classification. In our proposed approach, the mapping makes use of distance metric whose results are applied to kernel functions, such as RBF and Sigmoid. The main goal of such modification is to improve the accuracy rate.

The evaluation using 11 benchmark datasets and three different kernels showed the proposed approach achieved very interesting results, in which the application of kernel functions improved the accuracy rate of the traditional OPF, and even outperformed the well-known SVM when features were normalized. In summary, *k*OPF achieved satisfactory results and is an interesting option for classification, especially when training sets are very dynamic due to its low computational load for training purposes.

Chapter 5

Improving Optimum-Path Forest Classification Using Unsupervised Manifold Learning

This chapter presents the work accepted for presentation in the 24th International Conference on Pattern Recognition (ICPR) 2018 (AFONSO; PEDRONETTE; PAPA, 2018) (Qualis-CC A2), which proposes the combination of unsupervised manifold learning with two supervised OPF classifiers aiming the accuracy rate improvement.

5.1 Introduction

General classification performed by automated methods (e.g., machine learning algorithms) may not meet the users' expectation. For example, in Content-Based Image Retrieval (CBIR) applications, the unsatisfactory performance can be linked to the fact that low-level features (e.g., color, texture, and shape) are usually not able to capture the similarity observed by humans (LEE; JIN; JAIN, 2008). Besides the low-level feature issue, the fact that different data distribution may require a distinct distance metric for obtaining better results is also an issue that must be considered.

To overcome those problems, a few works were proposed attempting to reduce the semantic gap by developing new visual features (PENATTI; VALLE; TORRES, 2012) and similarity functions using low-level features (MÜLLER; PUN; SQUIRE, 2004). Visual features are not completely capable of capturing relevant semantic information, and traditional pairwise metrics often fail to provide reasonable results primarily because of the heterogeneity of the input space in image retrieval tasks (LEE; JIN; JAIN, 2008).

Finding the appropriate distance metric became paramount for a better classification/retrieval performance, also posing a challenging task (BELLET; HABRARD; SEBBAN, 2013). Studies shifted to the design of approaches that are capable of learning appropriate metrics for a given input data. In a nutshell, metric learning can be defined as the transformation of data samples from their original space to another feature space in such way the intra-class variation is reduced, meanwhile the inter-class variation is increased (LIONG; LU; GE, 2015; PEDRONETTE; GONÇALVES; GUILHERME, 2018).

Works available in the literature explore the three different learning methods: supervised, semi-supervised, and unsupervised. The majority of works follow the supervised (BIE; MOMMA; CRISTIANINI, 2003) and semi-supervised (HOI; LIU; CHANG, 2010) fashion by usually using samples with side-information (i.e., relevance judgments). Those works add constraints on pairwise distance metrics, which can be information provided by users.

On the other hand, unsupervised methods explore the dataset manifold by using more global affinity measures that consider the context of the database objects (ZHOU et al., 2003; PEDRONETTE; TORRES, 2013; PEDRONETTE; GONÇALVES; GUILHERME, 2018). In this scenario, contextual information is essential for finding an appropriate metric since it gives more details of the relation among the objects of a dataset. Hence, manifold learning showed to be a promising tool and has been applied in many different learning scenarios (LU; TAN; WANG, 2013; THEODORAKOPOULOS et al., 2016).

Pedronette et al. (PEDRONETTE; GONÇALVES; GUILHERME, 2018) proposed an unsupervised manifold learning algorithm for image retrieval that exploits the Reciprocal *k*-NN Graph and Connected Components (CCs) . Their work analyzes ranking information to learn the dataset structure, in which the reciprocal references encoded in the ranking information are modeled as a graph. Then, CCs are used to identify the geometry of the dataset manifold.

Although metric learning approaches have been used in a number of different applications, they have been poorly studied in the context of OPF-based classifiers. Therefore, the main contribution of this work is to propose a metric learning approach based on the work by Pedronette et al. (PEDRONETTE; GONÇALVES; GUILHERME, 2018) to be validated in the context of two supervised OPF classifiers: (i) one that makes use of a complete graph (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012), and (ii) the other that employs a *k*-nearest neighbors graph as an adjacency relation (PAPA; FERNANDES; FALCÃO, 2017). We have shown that OPF can benefit from manifold learning approaches in several public datasets.

The remainder of this paper is organized as follows. The proposed approach is presented in Section 5.2, while Section 5.3 details the experimental setup and results. Finally, the conclusions are stated in Section 5.5.

5.2 Proposed Approach

Effectively measuring the similarity among data samples remains a challenging problem in classification and retrieval tasks. The classical pairwise distance measures, as the Euclidean distance, often fail in taking into account the dataset structure. In this scenario, we propose to compute a more effective distance function $d(\mathbf{z},\mathbf{v})$ required by the OPF classification using an Unsupervised Manifold Learning Algorithm based on the Reciprocal kNN Graph and its Connected Components (PEDRONETTE; GONÇALVES; GUILHERME, 2018).

5.2.1 Reciprocal kNN Graph

The *Reciprocal kNN Graph* can be defined as an undirected graph $G_r = (\mathcal{V},\mathcal{E})$, where the set of vertices $\mathcal{V}$ is given by the data collection, such that $\mathcal{V} = \mathcal{Z}$. Each data sample in the dataset is represented by a node in the graph. Notice that, since the algorithm is unsupervised, the graph represents the whole collection including training and test sets, although no labeled information is used.

The edge set $\mathcal{E}$ is computed based on the k-reciprocal neighborhood considering different thresholds of k. Let $\mathcal{N}(\mathbf{z},k)$ denotes a neighborhood set which contains the k most similar samples to a data sample $\mathbf{z}$. Once the nearest neighbor relationships are not symmetric (QIN et al., 2011), the set of k-reciprocal nearest neighbors of sample $\mathbf{z}$ can be defined (QIN et al., 2011) as:

$$\mathcal{N}_r(\mathbf{z},k) = \{\mathbf{u} \in \mathcal{N}(\mathbf{z},k) \wedge \mathbf{z} \in \mathcal{N}(\mathbf{u},k)\}. \tag{5.1}$$

The Reciprocal kNN Graph is constructed at different neighborhood depths, allowing a multi-level analysis. Let t_k denotes a threshold which defines the value of k at a given moment of algorithm execution, the edge set E can be formally defined as:

$$E = \{(\mathbf{z},\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_r(\mathbf{z},t_k)\}. \tag{5.2}$$

Therefore, an edge between data samples $\mathbf{z}$ and $\mathbf{v}$ is constructed if they are reciprocal neighbors at a t_k depth.

5.2.2 Connected Components

The reciprocal neighborhood provides a strong indication of similarity (QIN et al., 2011). However, only a small number of edges is created, deriving a sparse and disconnected graph. In this way, the Connected Components of the Reciprocal kNN Graph are used for expanding the similarity neighborhood and analyzing the geometry of the dataset manifold. Data samples in the same connected component have their similarity increased.

Formally, a connected component $\mathscr{C}_i$ is defined as a set of data samples (PEDRONETTE; GONÇALVES; GUILHERME, 2018). Therefore, the CCs computed for the entire dataset is given by a set $\mathscr{S} = \{\mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_m\}$, such that $\{\mathscr{C}_1 \cap \mathscr{C}_2 \cap \cdots \cap \mathscr{C}_m\} = \emptyset$.

5.2.3 Reciprocal kNN Graph CCs Distance

The edges of the Reciprocal kNN Graph G_r and the set of Connected Components $\mathscr{S}$ are exploited for computing a novel and more effective distance among data samples, capable of considering the dataset manifold (PEDRONETTE; GONÇALVES; GUILHERME, 2018). Different depths of reciprocal neighborhood (represented by t_k) are considered. First, a similarity score $w_e(\mathbf{u}, \mathbf{v})$ between data samples $\mathbf{u}$, $\mathbf{v}$ is defined based on the graph connectivity:

$$w_e(\mathbf{u}, \mathbf{v}) = \sum_{t_k=1}^{k} \sum_{\mathbf{q} \in \mathscr{X} \wedge \mathbf{u}, \mathbf{v} \in \mathscr{E}(q)} (k - t_k + 1), \tag{5.3}$$

where $\mathscr{E}(\mathbf{q})$ denotes the set of nodes to which sample $\mathbf{q}$ has edges at a given depth t_k. Analogously, a similarity score $w_c(\mathbf{u}, \mathbf{v})$ is defined aiming at exploiting infomration encoded in the connected components:

$$w_c(\mathbf{u}, \mathbf{v}) = \sum_{t_k=1}^{k} \sum_{\mathbf{u}, \mathbf{v} \in C_l} (k - t_k + 1). \tag{5.4}$$

Finally, a *Reciprocal kNN Graph CCs Distance* d_r is defined considering information from both similarity scores as:

$$d_r(\mathbf{u}, \mathbf{v}) = \frac{1}{1 + w_e(\mathbf{u}, \mathbf{v}) + w_c(\mathbf{u}, \mathbf{v})}. \tag{5.5}$$

The distance d_r impacts the neighborhood sets which are used to update the graph G_r. Therefore, the algorithm can be iteratively repeated. Let the superscript $^{(t)}$ denotes the iteration, an iterative distance can be defined n terms of the current similarity score:

$$d_r^{(t+1)}(\mathbf{u},\mathbf{v}) = \frac{1}{1+w_e{}^{(t)}(\mathbf{u},\mathbf{v})+w_c{}^{(t)}(\mathbf{u},\mathbf{v})}. \tag{5.6}$$

After T iterations, a final distance $d(\mathbf{z},\mathbf{v}) = d_r^{(T)}(\mathbf{z},\mathbf{v})$ is computed and used by the OPF classification.

5.3 Methodology

The robustness of the proposed approach is assessed using four different image datasets, as described in Table 5.1.

Table 5.1: Description of the datasets.

dataset	Type	# samples	# classes
Brodatz	Texture	1,776	111
Corel 5k	Objects/Scenes	5,000	50
MPEG-7	Shape	1,400	70

Each dataset has a different number of descriptors that are computed according to their main applications (e.g., a dataset may be texture- or color-oriented), as listed below:

- Brodatz[1]: Color Co-Occurrence Matrix (CCOM) , Local Activity Spectrum (LAS) , and Local Binary Patterns (LBP) ;
- Corel 5k[2]: Convolutional Neural Network by Caffe using the full-connected layer 7 (CNN-Caffe) , Auto Color Correlograms Spatial Pyramid (ACC-SPy) , Color and Edge Directivity Descriptor Spatial Pyramid (CEDD-SPy) , Fuzzy Color and Texture Histogram Spatial Pyramid (FCTH-SPy) , Joint Composite Descriptor Spatial Pyramid (JCD-SPy) , and Local Binary Patterns Spatial Pyramid (LBP-SPy) ;
- MPEG7[3]: Articulation-Invariant Representation (AIR) , Aspect Shape Context (ASC) , Beam Angle Statistics (BAS) , Contour Features Descriptor (CFD) , Inner Distance Shape Context (IDSC) , and Segment Saliences (SS) ;

[1]`http://www.ci.gxnu.edu.cn/cbir/Dataset.aspx`
[2]`http://www.ci.gxnu.edu.cn/cbir/Dataset.aspx`
[3]`http://www.dabi.temple.edu/~shape/MPEG7/dataset.html`

Therefore, the proposed approach has an initial set of 15 different scenarios (i.e., number of pairs [dataset, descriptor]) to be assessed. In each scenario, a comparison is conducted considering the OPF-based classification based on the original distance function and the distance computed by the unsupervised manifold learning algorithm.

The experiments aimed at comparing the distances that were recommended by the authors of the datasets from now on called "original", against the distances computed by the proposed approach, from now on called "manifold". The parameter settings for the unsupervised manifold learning algorithm used only one iteration ($T = 1$) and the neighborhood size suggested by (PEDRONETTE; GONÇALVES; GUILHERME, 2018) with $k = 20$.

The proposed approach was evaluated considering both CG-OPF or kNN-OPF classifiers. There were used three different configurations of training/testing set sizes: $25\%/75\%$, $50\%/50\%$ and $75\%/25\%$. The main idea behind this variation is to evaluate the behavior of the manifold learning approach with different training set sizes, especially the small ones. In such scenarios, obtaining high-accuracy classification rates is more challenger and the use of the manifold learning presents a greater potential due to its capacity of considering the underlying dataset structure.

Moreover, the experiments were carried out by means of a hold-out process with 20 runs, being the best results of each tuple [dataset, descriptor, training set size] defined according to the Wilcoxon signed-rank test with variance as of 0.05. Notice that the accuracy rates were computed using the accuracy measure proposed by Papa et al. (PAPA; FALCÃO; SUZUKI, 2009).

It is important to highlight that the kNN-OPF has an additional step that is a pre-training required to search for the best value of k, say that k^*. That process is performed by means of a pre-training ($\mathscr{Z}_3$) and an evaluating set ($\mathscr{Z}_4$), such that $\mathscr{Z}_1 = \mathscr{Z}_3 \cup \mathscr{Z}_4$. Once k^* is found, $\mathscr{Z}_3$ and $\mathscr{Z}_4$ are merged, and the proper training is performed once more using k^* over $\mathscr{Z}_1$. The k search range was set empirically as $[1, 50]$ for all situations.

5.4 Experimental Results

In this section, we present the experimental results concerning the unsupervised manifold learning approach applied to OPF classifiers. Figures 5.1 – 5.6 depict the average accuracies organized by dataset and OPF variant. Although we did not display the standard deviation bars, the proposed approach outperformed the original manifolds in all situations, except for 1 out of a total of 90 situations (i.e., 1.11%) as follows: Brodatz dataset using the $75\%/25\%$ configuration and LBP descriptor with classification performed by CG-OPF. In the remaining scenarios, the proposed approach based on manifold learning either outperformed (85% - 94.45%) or it was statistically similar (4% - 4.44%) to the original distance functions.

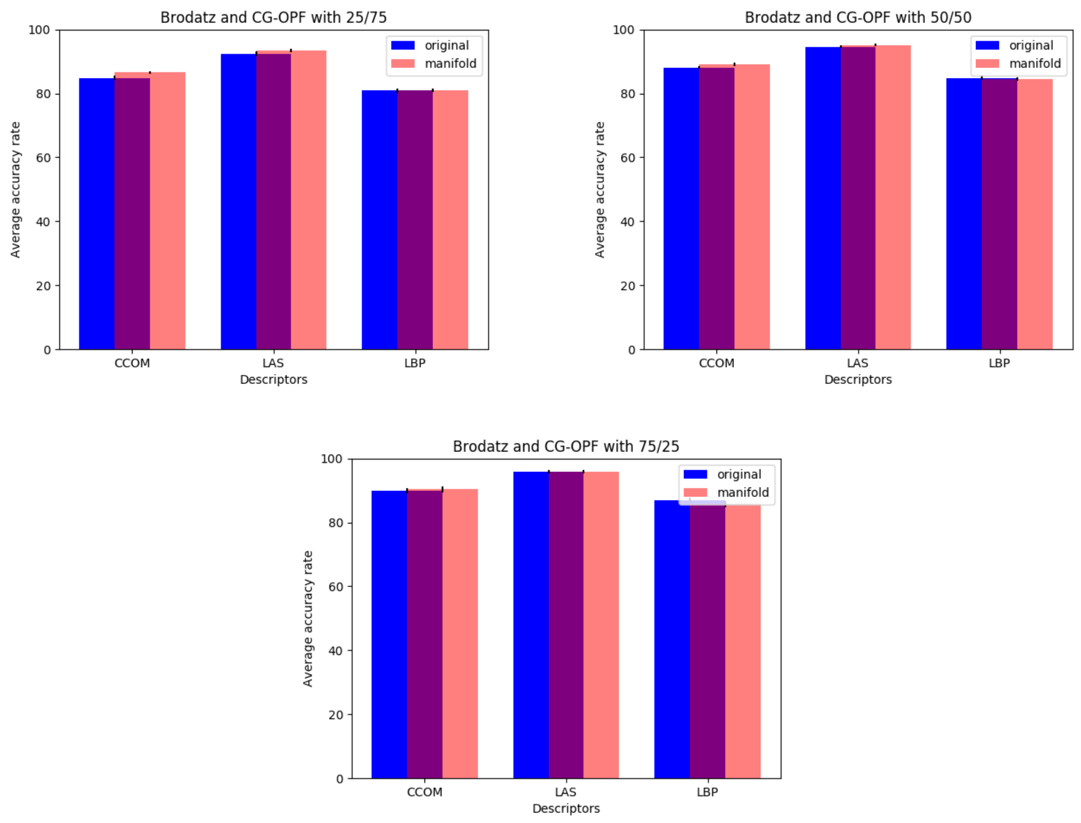


Figure 5.1: Average accuracy rates for Brodatz dataset and CG-OPF concerning the configurations 25%/75%, 50%/50%, and 75%/25%.

One of the main advantages of using manifold learning is to benefit in situations that picture small training sets. One can observe, for instance, that we can still improve the results over the smaller datasets (i.e., Brodatz and MPEG-7), which usually lack informative samples. Additionally, a comparison regarding the accuracies of both CG-OPF e *k*NN-OPF evidenced the former has been slightly more accurate in all situations. Actually, some previous works showed that both variants are somehow complementary to each other, which means we can benefit from ensembles of OPF-based classifiers (FERNANDES; PAPA, 2017).

The complementarity concerning both CG-OPF and *k*NN-OPF is related to the prototype estimation methodology mainly. The former estimates prototypes at the boundary among classes, which are known to be more susceptible to classification errors, and *k*NN-OPF estimates prototypes at the regions of highest concentration of samples, which are likely to be the center of the classes. Therefore, different sets of prototypes generate distinct optimum-path trees, thus partitioning the training set into varying configurations, which affect the final classification as a whole.

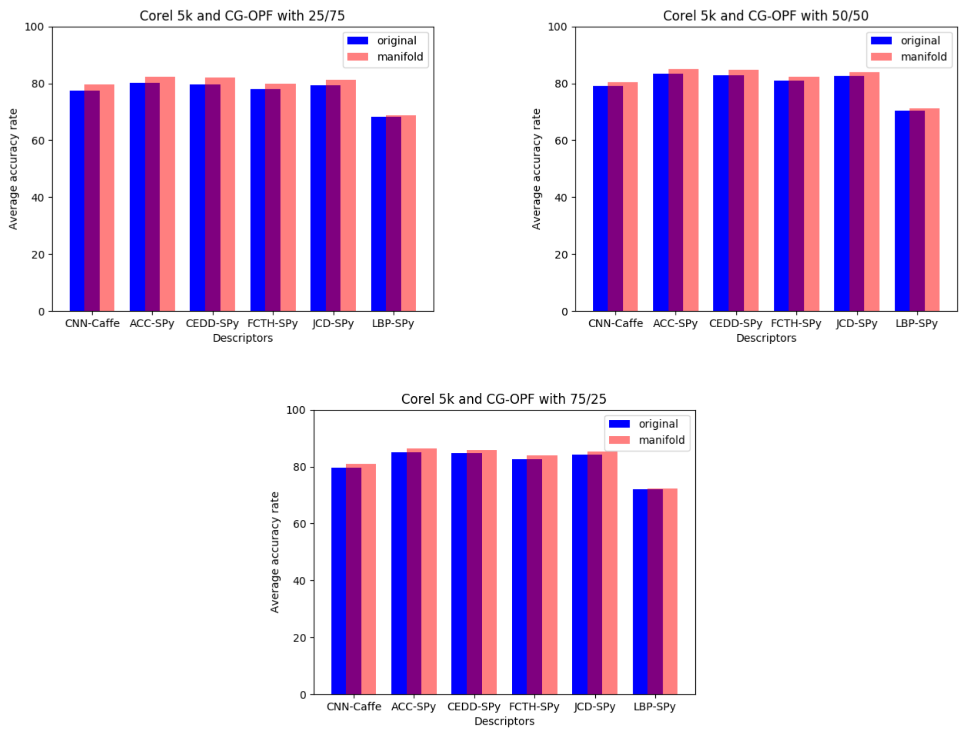


Figure 5.2: Average accuracy rates for Corel 5k dataset and CG-OPF concerning the configurations 25%/75%, 50%/50%, and 75%/25%.

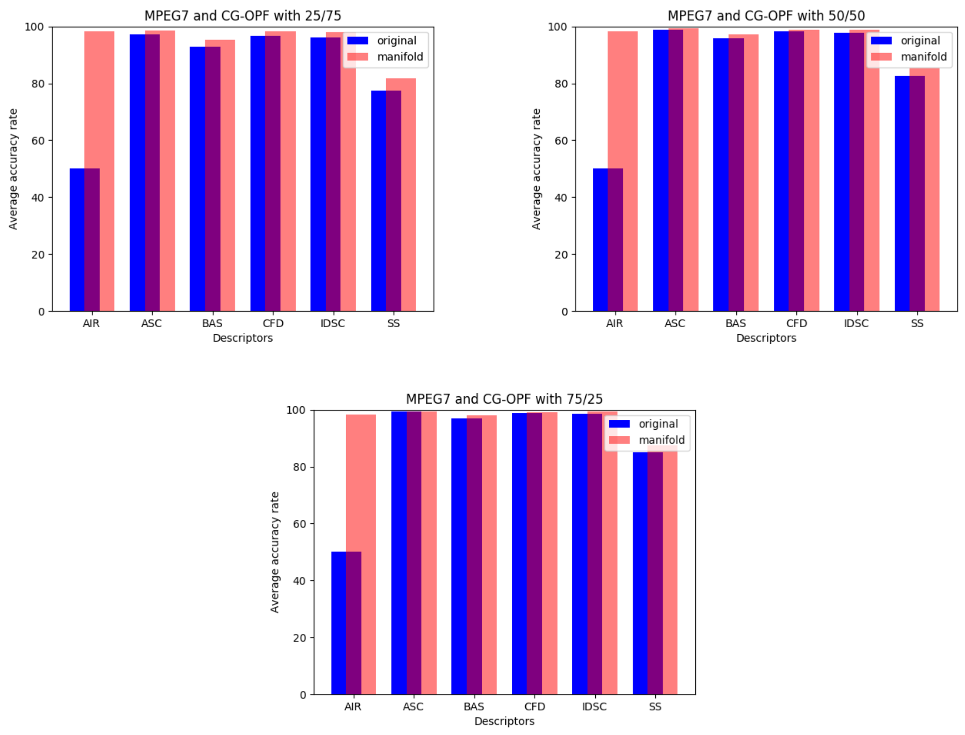


Figure 5.3: Average accuracy rates for MPEG7 dataset and CG-OPF concerning the configurations 25%/75%, 50%/50%, and 75%/25%.

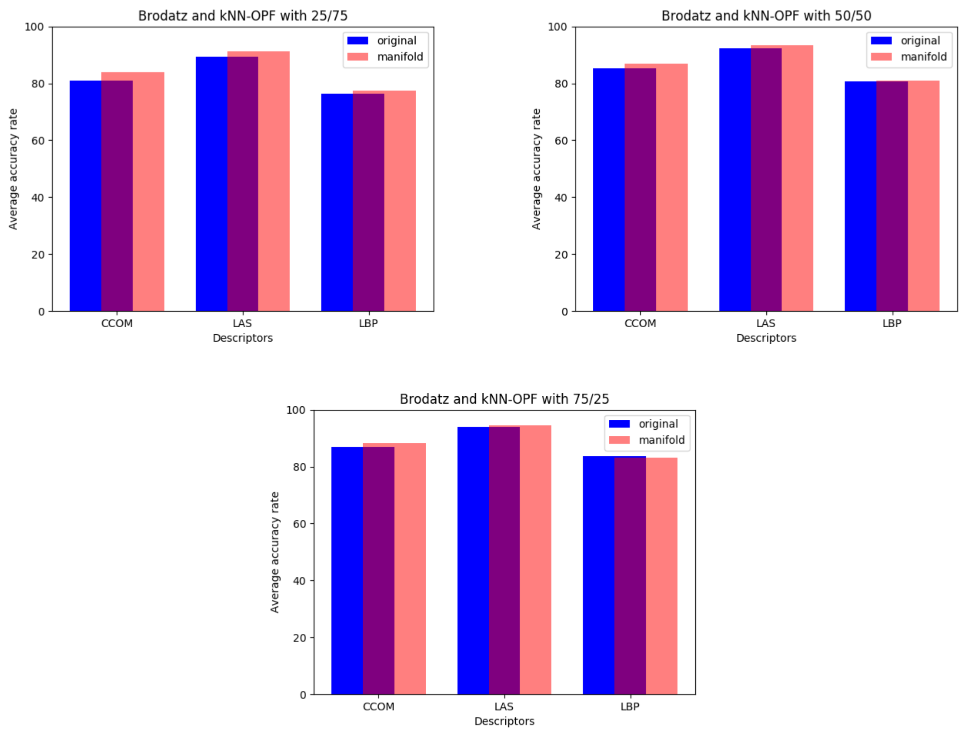


Figure 5.4: Average accuracy rates for Brodatz dataset and kNN-OPF concerning the configurations 25%/75%, 50%/50%, and %75/25%.

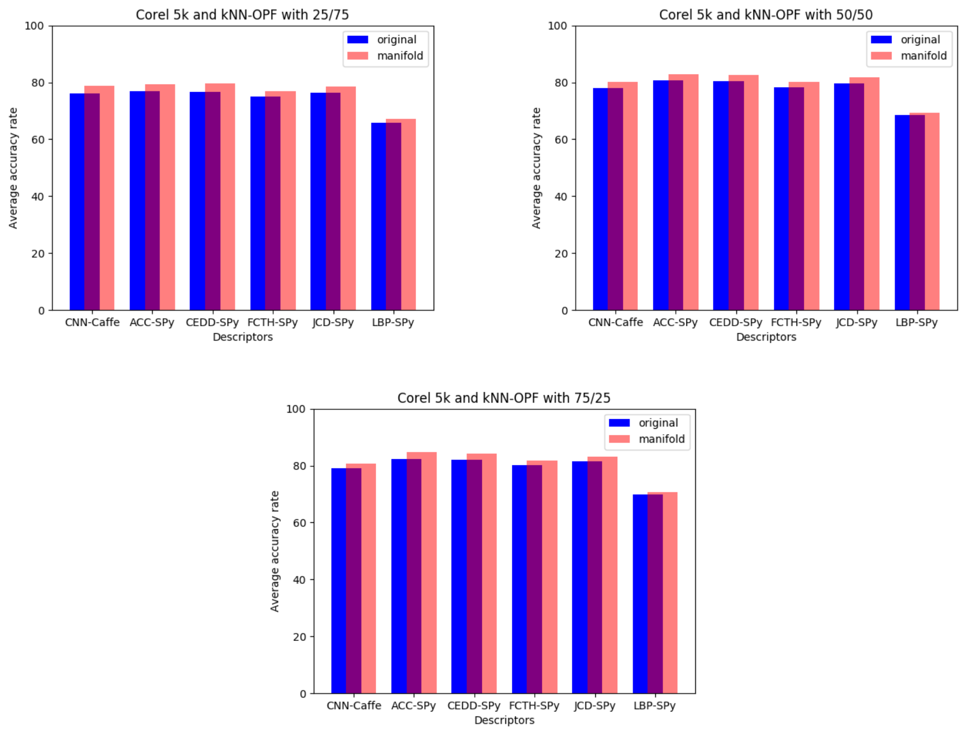


Figure 5.5: Average accuracy rates for Corel 5k dataset and kNN-OPF in the configurations 25%/75%, 50%/50%, and 75%/25%.

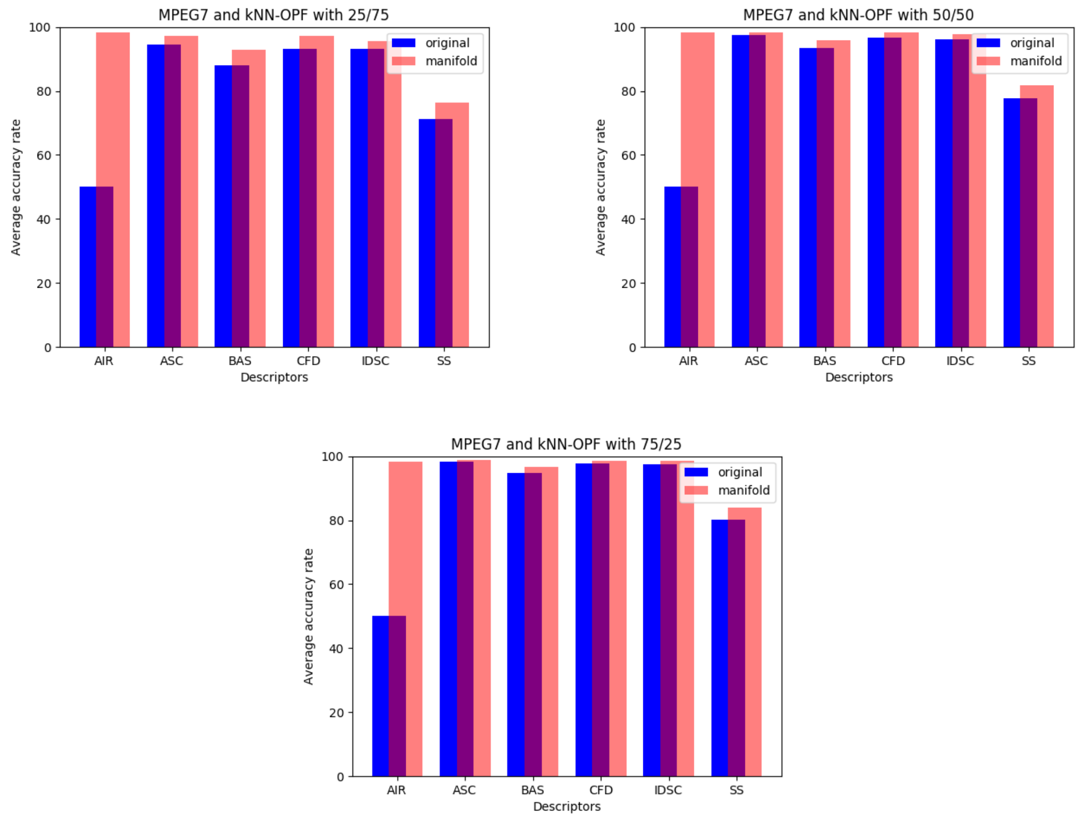


Figure 5.6: Average accuracy rates for MPEG7 dataset and *k*NN-OPF concerning the configurations 25%/75%, 50%/50%, and 75%/25%.

Regarding the training/testing set size configurations, larger proportions of training samples achieved better results, as expected. The improvement varies according to the dataset and descriptor used. One can notice small gains concerning the MPEG-7 dataset with AIR and IDSC descriptors, as well as more significant ones in the Corel 5k dataset with CEDD-Spy and JCD-Spy descriptors.

5.5 Conclusions

This paper proposed a novel approach that combines an unsupervised manifold learning algorithm with two versions of the OPF classifier. Experiments showed we can learn better manifolds that capture the contextual information encoded in the feature space. The main advantage was observed in situations where small training sets are available (e.g., Brodatz and MPEG-7 datasets), and with accuracy rates even higher than those achieved in a larger dataset such as Corel 5k. Therefore, unsupervised manifold learning showed to be a promising tool for improving accuracy, and especially when few training samples are available.

Chapter 6

EVOLVING OPTIMUM-PATH FOREST

This chapter studied and evaluated an evolutionary-based approach to estimate the prototypes for the supervised version of the OPF. The main question to be answered by this work is, *can an evolutionary-based approach estimate better prototypes than using the minimum spanning tree?* The work was submitted to the Natural Computing journal (Qualis-CC B1) and is currently under review.

6.1 Introduction

Nowadays, machine learning plays an essential role in our society. Recommendation and video surveillance systems, handwriting recognition, natural language processing, and autonomous vehicles are some examples of machine learning technologies that are already part of the everyday life of big companies. In fact, the growing amount of data increased the need for building efficient and effective models capable of analyzing such complex patterns.

A common problem found in machine learning concerns the pattern classification, where the fundamental idea is to discriminate samples and classify them correctly within categories or classes. The classification algorithms are divided according to the learning process, which can be supervised or unsupervised mainly. The former approaches assume the training set has been fully labeled with the correct outputs. Besides, unsupervised learning concerns clustering unlabeled samples that share similar properties. Last but not least, we shall mention semi-supervised learning (CHAPELLE; SCHLKOPF; ZIEN, 2010), where both labeled and unlabeled samples are used to guide the classification process.

Among the well-known classifiers in the literature, one shall mention Support Vector Machines (SVM) (CRISTIANINI; SHAWE-TAYLOR, 2000; CORTES; VAPNIK, 1995), Decision

Trees (ROKACH; MAIMON, 2005; QUINLAN, 1986), Artificial Neural Networks (HAYKIN, 2007; BISHOP, 1995), and Optimum-Path Forest (OPF) (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012; PAPA; FERNANDES; FALCÃO, 2017), among others. The OPF is a fast and straightforward graph-based framework that can handle some degree of overlap among classes. It is possible to build different classifiers by changing the adjacency relation, the path-cost function, and the prototype estimation methodology. (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012) proposed to place prototypes nearby decision boundaries since such regions are more prone to misclassification. The idea is to compute a minimum-spanning tree (MST) over the training graph and select connected samples from different classes as the prototypes. Later on, (IWASHITA et al., 2014) proposed a faster OPF that exploits the MST for further propagating labels and optimum-path costs, and (PONTI; RIVA, 2017) proposed an incremental OPF that allows faster training procedures.

Therefore, prototypes play an important role in the learning process since they are in charge of propagating the labels to the remaining (i.e., non-prototype) samples. As a matter of fact, (SOUZA; RITTNER; LOTUFO, 2014) showed that OPF and the well-known k-nearest neighbors classifier are similar when all training samples become prototypes. Although the OPF classifier proposed by. (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012) guarantees the lowest error boundary in the training set, such a property may not be interesting when the test set does not share similar behavior, thus leading to the so-called data overfitting.

In this paper, we propose a new approach for estimating prototypes using meta-heuristic optimization algorithms. The main idea is to minimize the classification error by choosing the most representative prototypes, which may or may not be located at the boundaries of the classes. In this case, we aim at avoiding overfitting by providing a data-driven approach for selecting prototypes, thus not relying solely on the MST methodology. We showed the proposed approach can outperform naïve OPF in several public datasets. Although one can use any meta-heuristic optimization technique, we observed that evolutionary ones are able to obtain the best results, thus leading us to coin the term "Evolutionary Optimum-Path Forest".

The remainder of this paper is organized as follows: Section 6.2 introduces the proposed approach. Sections 6.3 and 6.4 discuss the methodology and experiments, respectively. Finally, Section 6.5 states conclusions and future works.

6.2 Proposed Approach

As aforementioned, traditional OPF defines the set of prototypes by computing an MST over the training set and selecting the samples located at the decision boundaries. Samples at such locations are closer to others from different classes. Since the goal in the training phase is to minimize the classification error, border samples are more likely to become prototypes because they are more prone to be misclassified.

The proposed approach, hereinafter named as OPF$_{mh}$, aims at exploring a different prototype selection method other than via MST. The approach models such task as an optimization problem, which can be addressed by meta-heuristic algorithms. Such techniques are inspired by many different natural events and are mostly categorized as swarm-based ones. The swarm is comprised of agents that perform walks on the solution space (i.e, search space) to find the optimum/near-optimum solution. The approach used to perform walks varies from one technique to another, and the final solution is obtained by either a certain number of iterations or until some predefined error is reached. Each position in the search space defines a possible solution, which is evaluated by the so-called *fitness function*.

Let $\mathscr{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ be the set of possible solutions (i.e., the set of candidate samples to become prototypes), such that $\boldsymbol{x}_i \in \Re^n$. The number m of prototypes to be computed by OPF$_{mh}$ is a user-defined parameter that concerns the percentage of samples that must be promoted as prototypes. Notice that the traditional OPF does not have any concern about keeping a certain proportion of prototypes for all classes, which is now considered in the proposed approach. The rationale behind that idea is to keep a balanced proportion of prototypes considering all classes. Figure 6.1 illustrates the composition of a solution $\boldsymbol{x}_i \in \mathscr{X}$, in which each decision variable stands for a coordinate of a prototype.

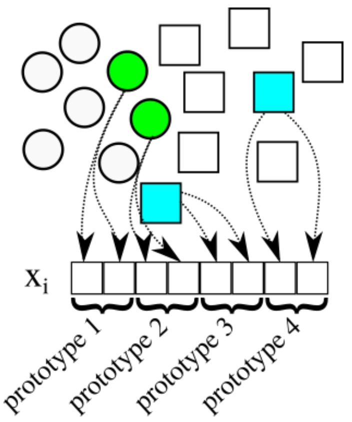


Figure 6.1: Modeling prototype selection as an optimization task.

For each possible solution, the working mechanism can be divided into two steps: (i) to

Table 6.1: Parameter configuration.

Technique	**Parameters**
Artificial Bee Colony (ABC) (KARABOGA; AKAY; OZTURK, 2007)	$limit = 10$
Adaptive Inertia Weight Particle Swarm Optimization (AIWPSO) (NICKABADI; EBADZADEH; SAFABAKHSH, 2011)	$w = 0.7$, $w_{min} = 0.5$, $w_{max} = 1.5$, $c_1 = 1.7$, $c_2 = 1.7$
Bat Algorithm (BA) (RODRIGUES et al., 2014)	$f_{min} = 0$, $f_{max} = 0$, $r = 0.5$, $A = 1.5$
Black Hole algorithm (BHA) (HATAMLOU, 2013)	parameterless
Backtracking Search Optimization Algorithm (BSA) (CIVICIOGLU, 2013)	$mix\_rate = 1.0$, $F = 3$
Brainstorm Optimization (BSO) (SHI, 2011b)	$p_{one\_cluster} = 0.8$, $p_{one\_center} = 0.4$, $p_{two\_centers} = 0.5$
Cuckoo Search (CS) (YANG; DEB, 2010)	$\beta = 1.5$, $p = 0.25$, $\alpha = 0.8$
Firefly Algorithm (FA) (YANG; XINGSHI, 2013)	$\alpha = 0.2$, $\beta = 1$, $gamma = 1$
Flower Polinization Algorithm (FPA) (YANG; KARAMANOGLU; HE, 2014)	$\beta = 1.5$, $p = 0.8$
Genetic Algorithm (GA) (HOLLAND, 1992)	$p_{Mutation} = 0.2$
Harmony Search (HS) (GEEM, 2009)	$HMCR = 0.7$, $PAR = 0.7$, $PAR_{min} = 0$, $PAR_{max} = 1$, $bw = 10$;, $bw_{min} = 0$, $bw_{max} = 20$

train a classifier over the training set and (ii) further evaluate it over the validating set. In a nutshell, the former step uses the prototypes' position encoded in each solution $\boldsymbol{x}_i \in \mathscr{X}$ to start the competition process that ends up in the optimum-path forest. Further, the classifier is evaluated over the validating set, and its accuracy is used as the fitness function. The two-step process is performed for each solution and at all iterations. The best solution is the one with the best accuracy in the evaluation step, and it is always updated when a better solution is found. Finally, the prototypes encoded by the best solution are used to design the learner that is going to be used to classify the testing set. Figure 6.2 depicts the dynamics of the prototype estimation methodology over the iterations in a meta-heuristic technique. A different set is selected at each iteration and further evaluated.

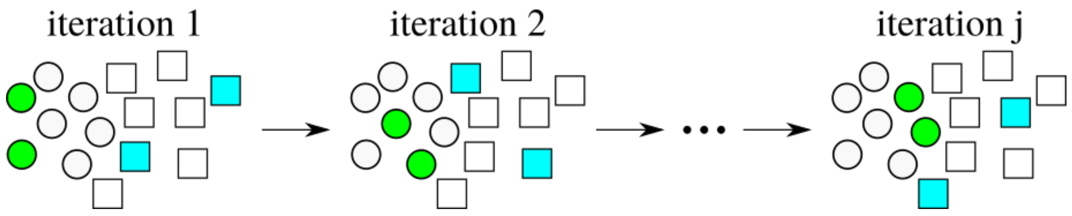


Figure 6.2: Prototype selection dynamics over the iterations.

6.3 Methodology

The OPF$_{mh}$ was evaluated using 11 meta-heuristic techniques with source-codes provided by the open-source library LibOPT (PAPA et al., 2017)[1] and additional tools from the library LibDEV[2]. Regarding the OPF classifier, we used the open-source library LibOPF (PAPA; SUZUKI; FALCÃO, 2014)[3]. The optimization techniques were selected based on their promising results in many other applications, despite being from different nature-inspired heuristics. Table 6.1 presents the parameter setting up applied for each technique.

[1] `https://github.com/jppbsi/LibOPT`
[2] `https://github.com/jppbsi/LibDEV`
[3] `https://github.com/jppbsi/LibOPF`

The experiments were carried out using 20 agents and 5 iterations. Such values were empirically chosen since the experiments showed the convergence for any pair [technique, dataset] occurs in a very few iterations. The number of decision variables n (i.e., the dimensionality of the search space) varies according to the dataset and number of prototypes, is defined as follows:

$$n = A \times \left(\sum_{i=1}^{c} |D_i| \times p \right), \tag{6.1}$$

where $|D_i|$ stands for the number of samples that belong to class i, A denotes the number of features, c stands for the number of classes in the dataset, and p is the percentage of training samples to be used as prototypes.

As aforementioned, OPF$_{mh}$ keeps a proportional amount of prototypes for all classes, which is encoded by the parameter p, which was empirically set as 0.15 (15%). Each possible solution is evaluated according to the accuracy it provides over the validating set, and the best overall solution is used to classify the testing set. The accuracy is computed using the approach proposed by (PAPA; FALCÃO; SUZUKI, 2009), which takes into account unbalanced datasets.

The robustness of OPF$_{mh}$ is assessed on 15 public datasets[4] with a varying number of samples, dimensions, and number of classes. By building such an experimental environment, it is possible to explore the approach under different scenarios, especially for a low/high number of samples per class. Table 6.2 provides an overview of each dataset. The experiments were carried out in a hold-out fashion using 15 runs for each pair [technique, dataset], in which datasets were randomly partitioned into three subsets: training, validating, and testing sets with 50%, 25%, and 25% of the samples, respectively. Additionally, all techniques were compared using statistical validation.

6.4 Experimental Results

This section reports the experiments as well as a discussion concerning the results obtained by each technique. Firstly, it is presented a comparison of the number of prototypes computed by traditional OPF and the proposed approach (OPF$_{mh}$) in Table 7.1. One can observe that OPF$_{mh}$ computed a significantly lower number of prototypes than OPF to most datasets. The most significant difference occurred in the Breast Tissue dataset (i.e., 7 times lesser). The situations in which OPF$_{mh}$ selected more prototypes than OPF are mostly characterized by

[4]https://archive.ics.uci.edu/ml/datasets.html

Table 6.2: General information about the datasets.

Datasets	# samples	# features	# classes
Abalone	4,177	8	3
Australian	690	14	2
Banknote	1,372	4	2
Breast Cancer	683	10	2
Breast Tissue	106	9	6
Diabetes	768	8	2
Fourclass	862	8	2
German Numer	1,000	24	2
Glass	214	9	6
Ionosphere	351	34	2
Iris	150	4	3
Liver Disorders	345	6	2
MPEG	1,400	180	70
Seeds	210	7	3
Wine	178	13	3

datasets with a higher ratio of samples per class, such as the Fourclass dataset.

Table 6.3: Number of prototypes selected.

Datasets	# training samples	# prototypes	
		OPF$_{mh}$	OPF
Abalone	2,087	311	2,061
Australian	344	50	153
Banknote	684	102	12
Breast Cancer	341	51	40
Breast Tissue	49	6	42
Diabetes	384	57	263
Fourclass	430	63	13
German Numer	500	74	362
Glass	104	15	77
Ionosphere	174	25	68
Iris	72	9	14
Liver Disorders	172	25	154
MPEG	700	105	697
Seeds	102	15	14
Wine	88	12	17

Figure 6.3 depicts the distribution of the training samples concerning the Abalone, Fourclass, Ionosphere, and Seeds datasets, as well as an overview of the distribution of the selected prototypes in a 2D representation obtained using the well-known Principal Component Analysis technique (S., 1901). These datasets were chosen to illustrate a few situations, such as one of the highest differences (i.e., Abalone) and almost the same number of computed prototypes

(i.e., Seeds). One can notice that many of the prototypes selected by OPF are located in the overlap area among classes.

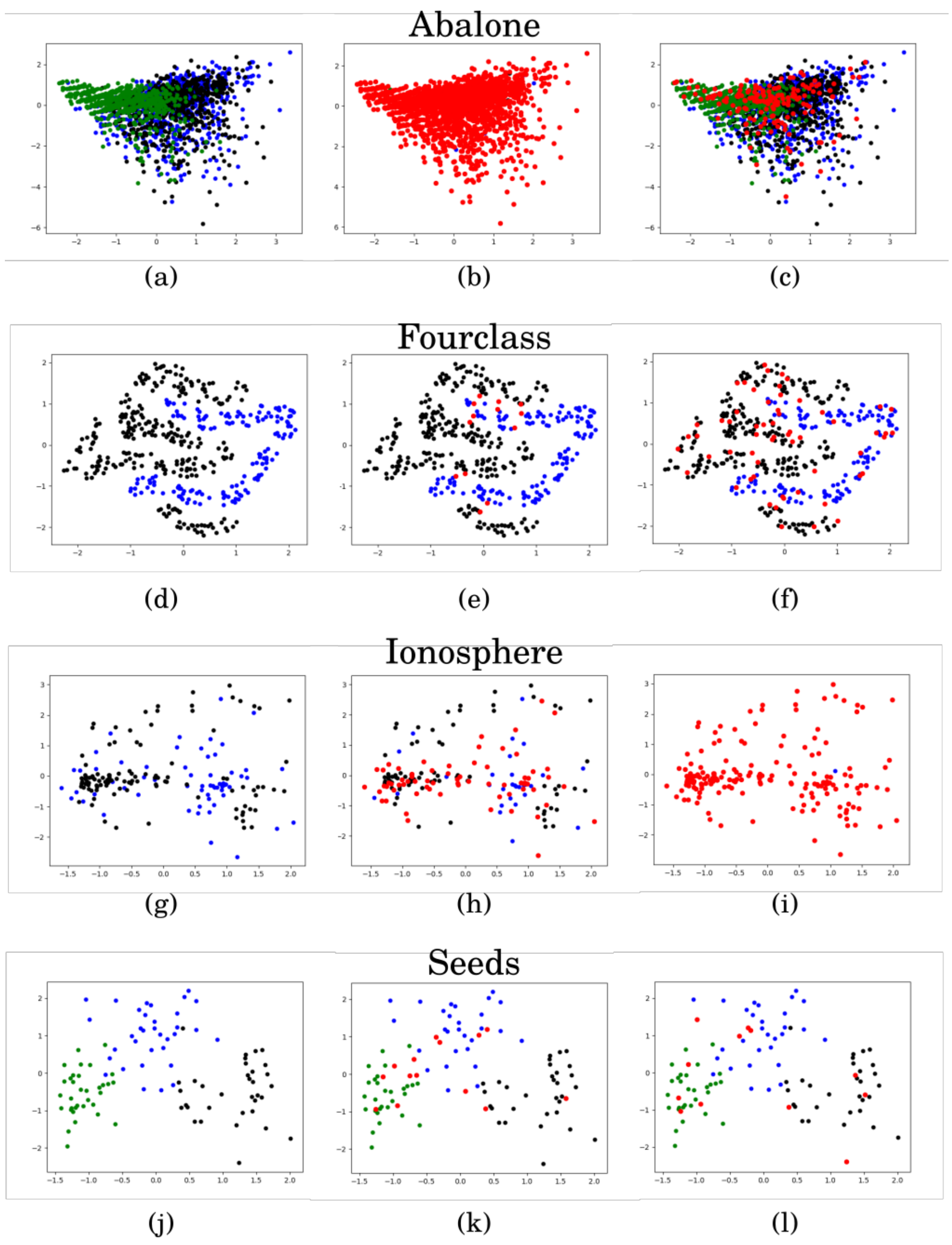


Figure 6.3: Distribution of the training samples in the feature space concerning the Abalone, Fourclass, Ionosphere, and Seeds datasets. The first column stands for the training set previous to any training. The second and third columns represent the training set labeled after the training procedure by OPF and OPF$_{mh}$, respectively. The prototypes are represented by the red dots.

Concerning the results, the experiments aimed at evaluating the accuracy over both validation and testing sets. Tables 6.4 and 6.5 present the average accuracy over 15 runs for each pair [technique, dataset] over the validating and testing sets, respectively. The Wilcoxon signed-rank test (WILCOXON, 1945) with a significance of 5% assesses the best accuracies, which are reported in bold.

Table 6.4 presents the accuracy rate concerning the validation set. One can observe a predominance of OPF in the best results. The Breast Cancer, German Numer, and Liver Disorders were the most challenging datasets to OPF$_{mh}$, whose average accuracy reached values below 60%. One of the reasons for such low results might be a considerable overlapping among classes. Hence, the OPF might have taken advantage of its method of selection of border samples to achieve 100% of accuracy in the validation set. Except for the Breast Cancer dataset, the number of prototypes computed by OPF were considerably higher than those of OPF$_{mh}$ (i.e., 4.9 times for German Numer and around 6 times for Liver Disorders dataset).

Table 6.4: Accuracy rate over the validation set.

Datasets	OPF$_{mh}$											OPF
	ABC	AIWPSO	BA	BHA	BSA	BSO	CS	FA	FPA	GA	HS	
Abalone	52.828	61.042	63.214	61.166	61.525	62.637	57.521	60.799	60.991	53.777	61.080	**99.850**
Australian	55.621	75.956	86.153	81.323	81.960	85.651	71.051	76.476	74.290	55.211	79.538	**100.000**
Banknote	93.455	99.075	99.459	99.180	99.511	99.616	97.958	99.197	99.162	93.455	99.058	**100.000**
Breast Cancer	49.902	51.135	59.198	53.623	55.348	56.730	51.135	52.317	50.219	49.811	52.071	**100.000**
Breast Tissue	65.957	61.547	68.357	62.821	61.286	68.357	65.957	67.152	64.768	65.957	60.597	**100.000**
Diabetes	50.346	60.445	69.772	63.209	63.906	67.643	59.347	64.134	61.113	51.128	59.914	**100.000**
Fourclass	62.230	97.274	99.111	93.413	97.633	98.894	84.278	97.647	97.523	66.070	97.331	**100.000**
German Numer	49.429	50.610	56.965	52.171	52.730	56.032	51.530	51.860	51.600	49.143	51.257	**100.000**
Glass	61.466	60.315	65.919	65.821	61.288	66.606	63.078	59.739	58.428	62.962	58.104	**100.000**
Ionosphere	50.000	60.699	83.363	74.568	73.125	81.607	62.247	69.673	65.789	52.068	70.015	**100.000**
Iris	78.846	88.974	94.487	92.179	89.744	94.615	85.000	88.077	92.692	80.385	90.897	**100.000**
Liver Disorders	45.674	50.000	60.874	54.570	53.163	61.356	47.330	47.426	48.667	45.193	48.789	**100.000**
MPEG	72.754	74.696	75.507	74.783	74.773	75.556	73.101	74.116	73.749	72.754	73.884	**89.790**
Seeds	81.944	85.185	92.963	88.148	87.130	92.500	82.500	87.963	86.852	81.944	84.815	**100.000**
Wine	91.313	89.914	93.653	91.095	88.633	88.037	91.609	89.815	86.895	91.313	85.677	**100.000**

OPF$_{mh}$ highest accuracies were achieved in the Banknote, Fourclass, Iris, Seeds, and Wine datasets (i.e., accuracies over 90%). The amount of samples per class appears to have a lower impact on the learning process than the distribution of the training samples in the feature space. One example is the MPEG dataset, which has 700 training samples distributed in 70 classes (i.e., 10 samples per class), and with accuracies between 72% and 76%. Concerning the meta-heuristic techniques, they provided similar accuracies over the validation set.

Table 6.5 presents the average accuracy over the testing set. As in the validation set, OPF$_{mh}$ achieved the lowest accuracies in the Breast Cancer, German Numer, and Liver Disorders datasets. Although the significant difference in accuracy between the baseline and the proposed approach in many scenarios concerning the validation set, OPF$_{mh}$ achieved competitive results and now outperformed standard OPF in many datasets.

It is worth noting that OPF showed a significant decrease in accuracy in many datasets, such as Abalone, Diabetes, German Numer, and Liver Disorders. This phenomenon can be related to the overfitting during the training process, as argued earlier. By taking into account the information from Table 7.1, one shall notice that OPF selected almost all training samples of the

Table 6.5: Accuracy rate over the testing set.

Datasets	OPF$_{mh}$											OPF
	ABC	AIWPSO	BA	BHA	BSA	BSO	CS	FA	FPA	GA	HS	
Abalone	52.159	61.053	**62.685**	**61.666**	**62.047**	**61.834**	**59.228**	**61.800**	60.692	52.856	61.866	59.980
Australian	58.732	73.688	**80.926**	**77.315**	76.984	**80.377**	70.387	73.830	71.582	58.081	76.414	75.040
Banknote	92.147	98.674	99.162	98.883	99.546	99.651	97.051	99.005	99.075	92.147	98.691	**99.740**
Breast Cancer	54.617	50.620	49.637	49.282	51.559	50.336	53.932	51.411	50.887	55.197	49.934	**93.270**
Breast Tissue	59.248	59.509	59.205	59.261	58.445	59.205	59.248	60.078	60.247	59.248	56.234	**74.960**
Diabetes	50.293	55.726	60.009	57.120	60.399	59.789	54.267	59.566	58.802	50.393	56.617	**63.770**
Fourclass	62.950	96.168	**99.520**	93.582	97.271	**99.520**	81.943	97.122	96.748	67.045	96.220	**100.000**
German Numer	49.810	50.914	54.806	52.571	53.403	54.476	52.495	53.670	53.270	49.562	52.997	**64.100**
Glass	66.945	65.010	68.173	67.379	62.627	67.749	66.983	61.703	61.288	66.760	59.090	**84.030**
Ionosphere	50.000	61.663	**82.012**	73.222	73.560	**79.974**	61.436	69.444	67.204	52.058	69.764	81.060
Iris	76.923	90.128	**94.615**	**91.923**	89.231	**94.487**	85.128	87.051	**92.692**	78.974	**92.051**	94.230
Liver Disorders	50.575	52.323	50.085	50.148	52.119	49.625	50.159	53.310	53.541	49.905	50.885	**65.380**
MPEG	73.478	73.807	73.246	73.604	**74.908**	73.082	**74.097**	**74.512**	**74.444**	73.478	**74.097**	**74.350**
Seeds	84.722	86.204	**90.370**	87.500	88.148	**89.722**	84.352	**89.167**	86.574	84.722	85.278	**90.280**
Wine	94.074	92.922	94.957	92.883	90.802	90.350	95.062	92.099	91.835	94.074	90.831	**95.610**

Abalone dataset as prototypes (i.e., approximately 98.8%). The datasets mentioned above figure among the ones with the highest proportion of prototypes per training samples considering the OPF classifier.

Concerning the meta-heuristic techniques, all but ABC obtained similar performance in the Abalone dataset. Both BA and BSO techniques provided the best results in 6 out of 15 datasets, followed by BHA and FA with the best results in 3 situations. Figure 6.4 illustrates the relation between the number of training samples selected as prototypes and the accuracy in the testing set concerning standard OPF.

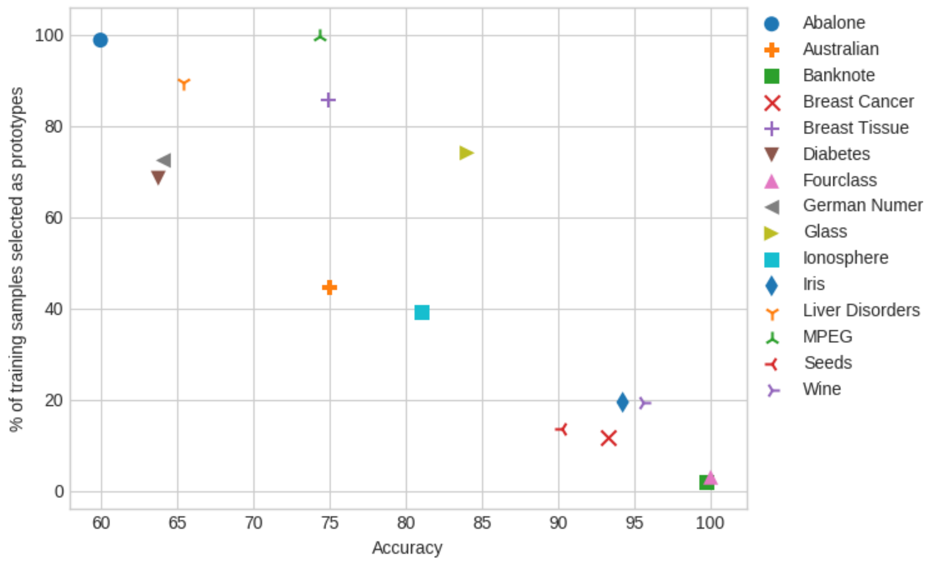


Figure 6.4: Relation between the accuracy over the testing set and the percentage of training samples selected as prototypes.

Situations with a few dozens of samples per class are not favorable for the use of optimization to select prototypes (e.g., MPEG). Due to the low number of samples, most of them may be located at the border among classes and should be used as prototypes by standard OPF to avoid classification errors. Since the optimization shall be limited to a low percentage of samples to avoid small-sized OPTs (i.e., overclustering), many samples will be prone to be misclassified. Taking MPEG dataset as an example, one can notice that nearly all samples were elected prototypes (i.e., only 3 are not prototypes).

Table 6.6 presents the average optimization time for each pair [technique/dataset], with the lowest times shown bolded. The Harmony Search showed to be the fastest technique to all datasets since it is not swarm-based, which means only one solution is evaluated per iteration.

Table 6.6: Optimization time [seconds].

Datasets	ABC	AIWPSO	BA	BHA	BSA	BSO	CS	FA	FPA	GA	HS
Abalone	49.798	26.975	26.818	49.226	26.676	27.120	11.104	22.553	27.083	26.170	**5.703**
Australian	1.531	0.834	0.827	1.508	0.825	0.831	0.347	0.702	0.841	0.815	**0.174**
Banknote	4.695	2.568	2.567	4.700	2.576	2.576	1.058	2.165	2.557	2.415	**0.531**
Breast Cancer	1.343	0.741	0.733	1.337	0.733	0.745	0.306	0.621	0.735	0.709	**0.157**
Breast Tissue	0.034	0.017	0.016	0.031	0.017	0.018	0.007	0.014	0.017	0.016	**0.004**
Diabetes	1.605	0.869	0.878	1.605	0.879	0.865	0.360	0.728	0.866	0.848	**0.182**
Fourclass	1.784	0.975	0.974	1.788	0.988	0.988	0.409	0.818	0.989	0.946	**0.208**
German Numer	3.437	1.866	1.857	3.406	1.881	1.873	0.778	1.588	1.875	1.843	**0.395**
Glass	0.126	0.069	0.068	0.125	0.069	0.069	0.029	0.058	0.068	0.068	**0.015**
Ionosphere	0.533	0.293	0.290	0.536	0.292	0.294	0.121	0.245	0.292	0.282	**0.061**
Iris	0.056	0.030	0.030	0.056	0.031	0.031	0.013	0.026	0.031	0.030	**0.007**
Liver Disorders	0.330	0.179	0.179	0.325	0.179	0.182	0.075	0.150	0.182	0.151	**0.038**
MPEG	24.619	13.376	13.243	24.398	13.289	13.296	5.564	11.095	13.280	13.381	**2.788**
Seeds	0.129	0.069	0.070	0.127	0.071	0.072	0.029	0.060	0.071	0.069	**0.015**
Wine	0.103	0.056	0.055	0.102	0.056	0.057	0.024	0.047	0.056	0.055	**0.012**

6.5 Conclusions

This paper introduced a meta-heuristic-based approach for selecting prototypes concerning the Optimum-Path Forest classifier. The prototypes are key samples and crucial to the effectiveness of the classifier and shall be learned from the training set. The outcome of the training procedure is a collection of optimum-path trees, each rooted at a prototype node. The traditional approach selects those samples located at the border among classes since these samples are more likely to be misclassified because they are closely located to samples from other classes.

The proposed approach, i.e., OPF$_{mh}$, was evaluated using 11 meta-heuristic techniques over 15 datasets with experimental results compared against the standard OPF. The preliminary results showed that OPF could achieve better generalization than OPF$_{mh}$ during training, with an accuracy of 100% in the validation set in most datasets.

However, this might come at a high cost. It has been observed that OPF employed a significantly higher number of prototypes, which might have overfitted the model, thus leading to decreased accuracies in the testing set. Almost all training samples were used as prototypes in the Abalone and MPEG datasets, for instance. On the other hand, OPF_{mh} selected a lower amount of prototypes and achieved competitive accuracy rates in the testing set and even outperforming standard OPF in some situations. We also conclude that a meta-heuristic-based approach can provide a better trade-off between data generalization and accuracy in the testing set, despite being a promising option to improve OPF classifier.

Concerning the meta-heuristic techniques, all approaches obtained similar results, with BA and BSO the most accurate ones. Also, HS showed to be the fastest one to all datasets. Concerning future works, we intend to evaluate the robustness of OPF_{mh} with respect to different concentrations of prototypes, as well as to consider other supervised (PAPA; FERNANDES; FALCÃO, 2017) and unsupervised (ROCHA; CAPPABIANCO; FALCÃO, 2009) variants of the OPF classifier.

Chapter 7

MULTIPLE-INSTANCE LEARNING THROUGH OPTIMUM-PATH FOREST

This chapter presents the work proposed by Afonso et al. (AFONSO et al., 2019b), which introduces the supervised OPF in the context of multiple-instance learning. The work was presented in the International Joint Conference in Neural Networks, 2019 (Qualis-CC A2).

7.1 Introduction

Machine learning techniques have been widely employed to address several problems, which are usually categorized into three distinct types: (i) supervised, (ii) semi-supervised, and (iii) unsupervised learning. The main difference among them relies on the amount of knowledge one possesses of the training set.

In their standard formulation, machine learning techniques consider that each dataset sample (e.g., a feature vector describing an image, signal, or a video) has been *individually* labeled. On the other hand, some problems require multiple instances of a given sample to define to what class it belongs to. Such situations are addressed using the so-called *multiple-instance* (MI) learning paradigm, where the learner receives a bag of samples instead of a single one. The most straightforward way to cope with MI problems is the *binary* case, which assumes a bag is considered *positive* if it contains (at least) a single sample labeled as positive. Also, a bag is considered *negative* when all samples are also labeled as negative ones (FOULDS; FRANK, 2010).

Keeler et al. (KEELER; RUMELHART; LEOW, 1981) and Dietterich et al. (DIETTERICH; LATHROP; LOZANO-PÉREZ, 1997) are acknowledged to be the first ones to explore the concept of multiple-instance learning. The latter work considered the problem of predicting drug activity, i.e., whether a collection of molecules could be used for making new drugs or not. Dietterich et al. (DIETTERICH; LATHROP; LOZANO-PÉREZ, 1997) proposed the three Axis-Parallel Rectangle algorithm to address such a problem, which constructs axis-parallel rectangles based on the conjunction of the features. Their work was validated in the Musk dataset (DHEERU; KARRA, 2017), which was one of the most popular benchmark datasets used in the multiple-instance learning research community for years.

Quellec et al. (QUELLEC et al., 2017) evaluated the MI paradigm in the context of the medical image and video analysis. The authors argued that MI-based techniques could be more suitable than standard approaches for some applications. Yu et al. (YU et al., 2018) employed Bi-directional Long-short Term and Convolutional Neural Networks for feature extraction in the context of topic categorization in documents. The authors also designed a framework based on the multiple-instance learning paradigm for the further classification step.

The main contribution of this paper is to introduce the OPF classifier in the context of multiple-instance learning. We considered two distinct versions of the Optimum-Path Forest classifier, and we showed it can outperform or obtain very much competitive results when compared to some well-known approaches for MI learning. As far as we are concerned, OPF- based classifiers have never been used in the MI paradigm.

Another contribution of this work is to model the problem of action recognition in well drilling reports as a multiple-instance learning task. During the drilling process in the petroleum off-shore platforms, workers keep a log of the whole procedure for further analysis and to improve safety. Sousa et al. (SOUSA et al., 2018) evaluated the OPF classifier for action recognition in well drilling reports with very much promising results, but the authors did not consider the problem as an MI classification process. In this paper, we mapped the above question to the context of multiple-instance learning, where a bag is composed of several instances of a given action for further recognition as an anomaly, i.e., an event that shall not be considered as a normal situation during operation times.

The remainder of this paper is organized as follows. Section 7.2 briefly revisits the theoretical background concerning the MI learning paradigm and OPF classifiers. Section 7.3 presents the proposed approach and the methodology, while Section 11.4 discusses the experiments. Finally, Section 7.5 states conclusions and future works.

7.2 Multiple-Instance Learning

Let $\mathscr{I} = \{(\mathbf{z}_1,y_1),(\mathbf{z}_2,y_2),\ldots,(\mathbf{z}_m,y_m)\}$ be a set of labeled instances (i.e., samples) such that $\mathbf{z}_i \in \Re^n$ and $y_i \in \{-1,1\}$ denote a sample and its label, respectively. Standard machine learning techniques, hereinafter called "single-instance" approaches, usually partition $\mathscr{I}$ into training and testing sets for learning purposes. Such an approach is widely employed by the research community, but it considers the label of each sample individually when designing the model.

As mentioned above, multiple-instance approaches take into account a collection of samples for label assignment. Let $\mathscr{B} = \{\mathscr{B}_1,\mathscr{B}_2,\ldots,\mathscr{B}_p\}$ a set of bags derived from $\mathscr{I}$ such that $\mathscr{B}_i$ contains a set of samples. Additionally, assume that $\mathscr{B}_i \cap \mathscr{B}_j = \emptyset$, $i \neq j$, and $\mathscr{B} = \mathscr{B}_1 \cup \mathscr{B}_2 \cup \ldots \cup \mathscr{B}_p$. In a nutshell, MI-based techniques aim at learning a function $f : \mathscr{B} \rightarrow \{-1,1\}$.

In binary-driven MI problems, the label of $\mathscr{B}_i$ is considered positive if there exists, at least, a single positive sample that belongs to it. On the other hand, the label of $\mathscr{B}_i$ is considered negative when all its samples are assigned to the negative class. Such an approach is also regarded as *presence-based* (WEIDMANN; FRANK; PFAHRINGER, 2003). Other approaches assume that a certain number of positive samples must be reached to label a bag as positive, also known as *threshold-based*. Finally, the *count-based* approaches establish lower and upper boundaries concerning the number of positive samples to classify an entire bag as positive.

7.3 Proposed Approach and Methodology

In this section, we introduce the proposed approach for multiple-instance learning using OPF-based classifiers, i.e., MI-CG-OPF and MI-*k*NN-OPF, which aim at classifying each bag as either positive or negative. The difference between MI-CG-OPF and MI-*k*NN-OPF relies on the adjacency relation, how prototypes are defined, and the path-cost function as described in the previous section.

The bags are represented by a single element, which is defined as the average of the feature vectors that fall in that bag, and modeled as the nodes of a graph. Figure 7.1 depicts an example of a complete graph approach used by CG-OPF, where each node (i.e., bag) is composed of positive (green) and negative (red) samples. Additionally, the dashed line surrounding each bag is colored with its corresponding label.

As described in Section 2.2.1, the next step concerning CG-OPF stands for the prototype estimation. To fulfill that purpose, we compute an MST on the complete graph and select the

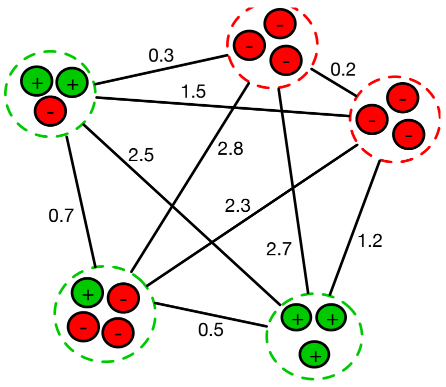


Figure 7.1: Complete graph where each node encodes a bag of instances.

connected bags with different classes as the prototypes. Figure 7.2 displays such a procedure, where the prototypes are highlighted. Notice that edges are weighted by the distance between bags, that are represented by a single instance that is computed as the average feature vector concerning all instances that fall in the bag. In this paper, we considered two different approaches for weighting edges: (i) the Euclidean distance and the (ii) cosine similarity.

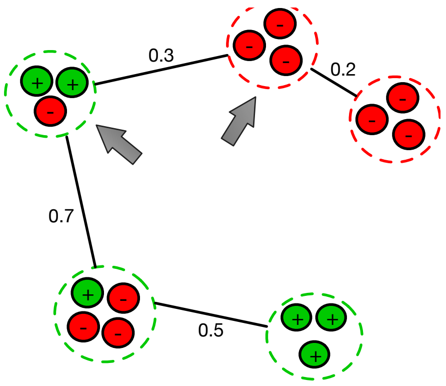


Figure 7.2: Minimum Spanning Tree and the prototypes indicated by the gray arrows.

After computing the prototypes, the competition process described in Section 2.2.1 takes place. As aforementioned, the main idea of CG-OPF is to minimize the cost of each node based on the f_{max} path-cost function (Equation 2.11). The prototypes are assigned a zero cost, meanwhile a large cost (i.e., ∞) is assigned to all remaining nodes, as depicted in Figure 7.3.

The final step of the training phase consists of the competition process itself, where the prototypes compete among themselves in order to conquer the remaining samples. This process ends up partitioning the training set into optimum-path trees, which are rooted in each prototype bag, as displayed in Figure 7.4. Notice that the optimum-path forest generated during the training phase (Figure 7.4) has a close similarity to the shape of the minimum spanning tree (Figure 7.2) over that same training set. As a matter of fact, such characteristic was considered in the work of Iwashita et al. (IWASHITA et al., 2014), which proposed a modification of the OPF

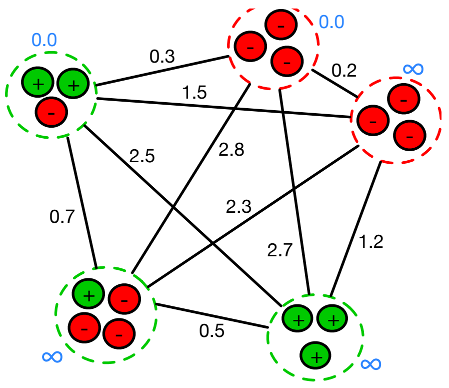


Figure 7.3: Complete graph with costs in blue assigned to all bags.

training algorithm that runs faster than its naïve version.

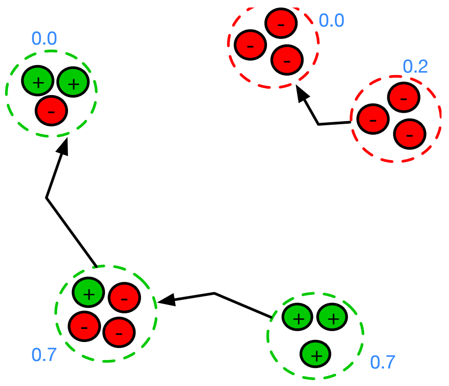


Figure 7.4: Optimum-path forest generated during the training phase.

7.3.1 Datasets

The robustness of the proposed approach was evaluated over 13 datasets that can be broadly categorized as image categorization (3), molecule description (2), text categorization (7), and anomaly detection (1). Below, we provide more details regarding each category as well as how the datasets were generated[1].

Image Categorization The automatic image categorization is comprised of three MI datasets derived from the Corel dataset. Andrews et al. (ANDREWS; TSOCHANTARIDIS; HOFMANN, 2002) preprocessed and segmented the images through the Blobworld system (CARSON et al., 1999). The outcome of the segmentation is a set of blobs characterized by color, texture, and shape descriptors that represent each image. The experiments used three classes of images (i.e., elephant, fox, and tiger), being each comprised of 100 positive and 100 negative bags. The negative

[1]The datasets, except the one related to anomaly detection, are available at `http://www.cs.columbia.edu/~andrews/mil/datasets.html`

samples represent blobs randomly generated from pictures of other animals. Andrews and colleagues argue that the limited accuracy of the image segmentation, the relatively small number of region descriptors and the small training set size makes this category quite a hard classification problem.

Molecule Description This category is represented by MUSK1 and MUSK2 datasets, which comprise a set of 92 and 102 molecule data, respectively. Humans experts manually labeled the molecules as either "musk" or "non-musk". A single molecule can assume many different shapes (conformation) due to the rotation. Hence, there were generated multiple low-energy conformations of the molecules. Each conformation is described by a 166-dimensional feature vector computed from surface properties with the highly similar ones being discarded. The molecules have an average of 6 conformations in MUSK1, and more than 60 conformations in each bag in the MUSK2 dataset.

Text Categorization The text categorization datasets were derived from the TREC9 dataset, also known as OHSUMED. This dataset is a collection of articles from MEDLINE of five years (1987-1991). The articles were labeled according to the MeSH terms (Medical Subject Headings). The total number of MeSH terms in TREC9 is $4,903$. Andrews and colleagues used approximately $54,000$ documents from the year of 1987, which were split through overlapping windows of a maximal of 50 words each. The experiments used the first seven categories (i.e., datasets TST1 – TST4, TST7, TST9, and TST10) of the pre-test portion with 200 positive and 200 negative bags in each category.

Anomally Detection This category is represented by a dataset comprised of textual descriptions of events during petroleum well drilling. The descriptions report actions and the parameters of the equipment used at that moment. A few descriptions may report problems during the operation, which are considered as anomalies. The dataset in question was built from daily well drilling reports (DWDR) provided by the Brazilian oil and gas company, Petrobras. The descriptions are represented by a 50-dimensional feature vector computed using the FastText (JOULIN et al., 2016). The bags are comprised of 10 instances labeled as either "normal" or "abnormal" activity, being the latter one the positive label. Hence, those bags containing at least one abnormal sample are labeled as an anomaly. The positive bags have an average of 3.2 positive instances.

Table 7.1 provides an overview of each dataset concerning the characteristics of the samples. One can notice that most of the datasets have a high level of sparsity. The number of non-zero samples takes into account the maximum amount among all bags.

Table 7.1: Dataset information.

Category	Datasets	# features	Bags		Instances	
		(non-zero)	positive	negative	positive	negative
Anomally	DWDR	50(50)	300	300	997	5,003
Image	Elephant	230(143)	100	100	762	629
	Fox	230(143)	100	100	647	673
	Tiger	230(143)	100	100	544	676
Molecule	MUSK1	166(166)	47	45	207	269
	MUSK2	166(166)	39	63	1,017	5,581
Text	TST1	66,552(31)	200	200	1,580	1,644
	TST2	66,153(31)	200	200	1,715	1,629
	TST3	66,144(31)	200	200	1,626	1,620
	TST4	67,085(32)	200	200	1,754	1,637
	TST7	66,823(31)	200	200	1,746	1,621
	TST9	66,627(33)	200	200	1,684	1,616
	TST10	66,082(32)	200	200	1,818	1,635

7.3.2 Experimental Setup

The experimental setup of this work follows the same protocol performed by Andrews et al. (ANDREWS; TSOCHANTARIDIS; HOFMANN, 2002). The option for such a protocol is to perform a fair comparison of accuracy performance with their proposed technique (i.e., *mi-SVM*), whose results are the baseline of this work. The validation is accomplished through a 10-fold cross-validation to all datasets. In the first round of experiments, the CG-OPF and *k*NN-OPF techniques were evaluated using the original feature space (i.e., indicated as *original* in tables) with Euclidean distance and the cosine similarity (i.e., shown as *cosine* in result tables). Notice that the mi-SVM technique using the linear, polynomial and radial basis function (RBF) kernels were applied to the original feature space.

Due to the high level of data sparsity, we performed a second round of experiments considering only the CG-OPF and *k*NN-OPF techniques. The original feature vectors had their dimensionality reduced through the well-known Principal Component Analysis (PCA). There were computed representations of three distinct dimensions for each dataset: 15 (PCA-15), 25 (PCA-25), and 50 (PCA-50).

Concerning the parameters used in the experiments, we set the parameter $k_{\max}$ of *k*NN-OPF as 20. The *k*NN-OPF has an additional step that is a pre-training required to search for the best value of k, say that k^*. That process is performed by means of a pre-training ($\mathscr{T}^{pre}$) and an evaluating sets ($\mathscr{I}^{eval}$), such that $\mathscr{I}^{tr} = \mathscr{I}^{pre} \cup \mathscr{I}^{eval}$. Once k^* is found, $\mathscr{I}^{pre}$ and $\mathscr{I}^{eval}$ are merged and a final proper training is performed once more using k^* over $\mathscr{I}^{tr}$. The CG-OPF is parameterless. Regarding the source-codes, we used the OPF implementations from the LibOPF (PAPA; SUZUKI; FALCÃO, 2014).

7.4 Experimental Results

As aforementioned, the proposed approach was evaluated over 13 datasets divided into four categories: image categorization, molecule description, text categorization, and anomaly detection in well drilling activities. Each category figures different levels of sparsity, which allows an investigation of the OPF's behavior in such situations in the context of MI-based problems. The average accuracies of both rounds of experiments are synthesized in Tables 7.2–7.5 with the best results shown underlined. The results of a few other MI-based techniques found in the literature are also reported for comparison purposes.

The average results achieved in the image categorization datasets are presented in Table 7.2. The analysis of accuracy in the original feature space shows that both MI-CG-OPF and MI-*k*NN-OPF achieved competitive results when compared to the baseline technique, and outperforming in the Fox dataset. Concerning the compressed representations, they provided better results in the Elephant dataset using Euclidean distance, where the most significative gains in accuracy can be observed. The best overall results were obtained by MI-CG-OPF (i.e., Fox) and MI-*k*NN-OPF (i.e., Elephant and Tiger).

The densest feature vectors characterize the MUSK1 and MUSK2 among all datasets. Once again, OPF-based classifiers achieved competitive results, especially when applying the cosine similarity. The results in Table 7.3 also show that compressed representations can be helpful in the classification task. The highest gains are reported by MI-CG-OPF using the Euclidean distance (i.e., 9% in MUSK1, and 8.5% in MUSK2).

The text categorization datasets have the highest sparsity level. The results reported in Table 7.4 show that highly sparse representations posed as a very challenging classification task to OPF-based classifiers. The TST1 dataset was the most difficult one where there were observed the lowest accuracy rates among all results using the original feature space. However, changing the similarity metric from Euclidean distance to the cosine similarity showed to be a better approach for such a situation, except for TST1 dataset. By generating denser and more compressed representations, it is possible to achieve higher accuracies, especially when the Euclidean distance is applied.

Table 7.2: Accuracy results on the Image Categorization datasets.

		Elephant	Fox	Tiger
EMDD (ZHANG; GOLDMAN, 2002)	-	78.3	56.1	72.1
mi-SVM (ANDREWS; TSOCHANTARIDIS; HOFMANN, 2002)	Linear	82.2	58.2	78.4
	Polynomial	78.1	55.2	78.1
	RBF	80.0	57.9	78.9
MI-CG-OPF	Original	74.5	58.0	78.5
	PCA-50	79.0	58.0	73.5
	PCA-25	79.0	64.5	74.0
	PCA-15	82.5	59.5	66.5
MI-*k*NN-OPF	Original	72.0	61.0	76.0
	PCA-50	81.5	57.5	78.0
	PCA-25	80.0	60.5	81.5
	PCA-15	84.0	58.0	68.5
MI-CG-OPF cosine	Original	79.5	53.5	75.0
	PCA-50	80.0	55.5	74.5
	PCA-25	81.5	60.0	73.5
	PCA-15	81.5	57.0	68.0
MI-*k*NN-OPF cosine	Original	80.0	56.0	76.5
	PCA-50	81.0	58.5	76.5
	PCA-25	84.0	59.0	76.5
	PCA-15	82.5	56.0	67.5

Table 7.4: Accuracy results on the TST datasets.

		TST1	TST2	TST3	TST4	TST7	TST9	TST10
EMDD (ZHANG; GOLDMAN, 2002)	-	85.8	**84.0**	69.0	80.5	75.4	65.5	78.5
mi-SVM (ANDREWS; TSOCHANTARIDIS; HOFMANN, 2002)	Linear	93.6	78.2	87.0	82.8	81.3	67.5	79.6
	Polynomial	92.5	75.9	83.3	80.0	78.7	65.6	78.3
	RBF	90.4	74.3	69.0	69.6	81.3	55.2	52.6
MI-CG-OPF	Original	49.8	47.5	49.8	55.0	51.0	45.8	51.3
	PCA-50	87.5	66.0	72.0	73.8	67.7	59.3	68.8
	PCA-25	90.8	70.5	75.3	78.5	72.0	56.8	69.5
	PCA-15	90.8	70.8	74.8	75.8	74.0	53.8	68.8
MI-*k*NN-OPF	Original	50.3	50.3	50.3	54.0	50.8	48.8	51.5
	PCA-50	85.3	65.3	71.3	74.8	68.3	59.8	69.3
	PCA-25	90.0	68.8	73.3	76.5	73.0	57.5	69.5
	PCA-15	91.3	71.0	72.3	79.3	74.3	56.5	71.3
MI-CG-OPF cosine	Original	49.8	60.8	65.5	71.8	63.5	57.0	69.0
	PCA-50	88.8	64.0	73.8	74.0	70.0	60.5	73.5
	PCA-25	91.8	70.0	80.3	79.0	74.8	62.5	72.0
	PCA-15	92.0	69.0	76.0	76.3	74.3	58.8	72.8
MI-*k*NN-OPF cosine	Original	49.8	60.3	65.5	72.0	64.3	56.5	66.8
	PCA-50	88.5	66.3	75.0	75.8	70.0	61.8	73.0
	PCA-25	92.3	70.0	80.3	79.0	71.3	60.5	71.8
	PCA-15	91.8	68.5	76.3	78.0	73.0	56.8	71.0

The DWDR dataset also has a compact representation but with a lower dimension if compared to the other datasets. This dataset can be considered a challenging one because the de-

Table 7.3: Accuracy results on the MUSK datasets.

		MUSK1	**MUSK2**
EMDD (ZHANG; GOLDMAN, 2002)	-	84.8	84.9
DD (MARON; RATAN, 1998)	-	88.0	84.0
MI-NN (RAMON; RAEDT, 2000)	-	88.9	82.5
IAPR (DIETTERICH; LATHROP; LOZANO-PÉREZ, 1997)	-	<u>92.4</u>	<u>89.2</u>
mi-SVM (ANDREWS; TSOCHANTARIDIS; HOFMANN, 2002)	RBF	87.4	83.6
MI-CG-OPF	Original	76.3	70.5
	PCA-50	85.3	76.4
	PCA-25	85.3	76.5
	PCA-15	81.5	79.0
MI-*k*NN-OPF	Original	79.8	68.3
	PCA-50	82.3	78.6
	PCA-25	83.0	68.1
	PCA-15	82.5	71.4
MI-CG-OPF cosine	Original	84.8	77.8
	PCA-50	89.0	79.7
	PCA-25	88.3	79.2
	PCA-15	84.8	80.3
MI-*k*NN-OPF cosine	Original	84.0	78.3
	PCA-50	86.5	79.7
	PCA-25	82.3	77.9
	PCA-15	83.8	72.1

scriptions are stored in a free-text format, i.e., the users can type in using an informal vocabulary. However, both techniques achieved interesting results with very high accuracy. Nonetheless, MI-CG-OPF and MI-*k*NN-OPF were nearly perfect in their classification results by reaching over 98% of accuracy in all situations. In this case, representations generated by PCA did not provide significant gain.

Table 7.6 presents the average training time of both CG-OPF and *k*NN-OPF in each scenario. It is worth noting the *k*NN-OPF training times include the time required for the optimization process responsible for finding k^*. One can observe that OPF-based classifiers are reasonably fast for training, even in the case of MI-*k*NN-OPF that features a fine-tuning parameter step.

7.5 Conclusions

This paper introduced a graph-based classifier for the multiple-instance learning problem. The proposed approach evaluated the Optimum-Path Forest classifier using the complete graph (CG-OPF) and *k*-nn (*k*NN-OPF) adjacency relations under different scenarios. The experiments

Table 7.5: Accuracy results on the Anomaly Detection dataset.

		DWDR
MI-CG-OPF	Original	99.0
	PCA-25	99.0
	PCA-15	99.0
MI-*k*NN-OPF	Original	98.3
	PCA-25	98.2
	PCA-15	98.0
MI-CG-OPF cosine	Original	99.2
	PCA-25	99.0
	PCA-15	99.2
MI-*k*NN-OPF cosine	Original	99.3
	PCA-25	<u>99.5</u>
	PCA-15	98.0

Table 7.6: Average training time [seconds]. The symbol '-' denotes that PCA-50 has not been employed to that dataset since it contains 50 **dimensions already.**

Techniques		DWDR	Elephant	Fox	Tiger	MUSK1	MUSK2	TST1	TST2	TST3	TST4	TST7	TST9	TST10
MI-CG-OPF	Original	0.068	0.036	0.043	0.036	0.006	0.006	20.077	20.308	20.371	21.016	20.500	20.090	20.321
	PCA-50	–	0.009	0.009	0.007	0.002	0.003	0.030	0.024	0.024	0.026	0.040	0.039	0.026
	PCA-25	0.40	0.004	0.004	0.005	0.003	0.007	0.020	0.019	0.017	0.019	0.018	0.018	0.021
	PCA-15	0.029	0.007	0.010	0.004	0.001	0.003	0.012	0.014	0.011	0.030	0.018	0.022	0.032
MI-*k*NN-OPF	Original	1.025	0.354	0.358	0.357	0.089	0.097	356.960	347.154	409.456	414.679	376.571	378.938	394.303
	PCA-50	–	0.106	0.106	0.105	0.047	0.036	0.372	0.397	0.385	0.377	0.374	0.388	0.384
	PCA-25	0.497	0.068	0.069	0.066	0.051	0.032	0.230	0.248	0.259	0241	0.251	0.253	0.240
	PCA-15	0.486	0.056	0.057	0.058	0.018	0.026	0.185	0.180	0.183	0.183	0.182	0.191	0.181
MI-CG-OPF cosine	Original	0.069	0.054	0.037	0.047	0.008	0.009	19.450	19.130	19.790	20.605	19.672	19.148	28.915
	PCA-50	–	0.016	0.013	0.022	0.005	0.002	0.031	0.052	0.057	0.025	0.040	0.020	0.032
	PCA-25	0.046	0.005	0.010	0.011	0.001	0.001	0.023	0.044	0.041	0.024	0.037	0.038	0.014
	PCA-15	0.030	0.003	0.008	0.003	0.001	0.001	0.015	0.023	0.034	0.011	0.027	0.015	0.037
MI-*k*NN-OPF cosine	Original	0.808	0.340	0.342	0.342	0.071	0.087	351.577	349.390	345.774	350.532	349.146	348.089	345.300
	PCA-50	–	0.106	0.107	0.112	0.041	0.045	0.391	0.360	0.351	0.355	0.355	0.357	0.355
	PCA-25	0.549	0.072	0.074	0.075	0.032	0.044	0.241	0.228	0.220	0.222	0.222	0.222	0.222
	PCA-15	0.428	0.059	0.059	0.059	0.026	0.036	0.192	0.186	0.167	0.168	0.168	0.178	0.173

were performed using a variety of datasets that included text, image, and molecule data.

The experimental results were compared against a baseline work, where there was proposed an MI-based version of the well-known Support Vector Machine classifier. Moreover, the proposed approach was evaluated in the second round of experiments using a compressed representation of the original feature space through the Principal Component Analysis, as well as distinct similarity metrics to extend the study.

The MI-CG-OPF and MI-*k*NN-OPF achieved competitive results in the image categorization and MUSK datasets. The sparsity showed to be an issue for the proposed approach as observed in the text categorization datasets. However, a change in the similarity metric allowed a significative gain in accuracy in almost all cases. Moreover, denser representations of the original feature space also come as an approach to the sparsity issue.

The anomaly detection in well drilling reports was also considered since it is of great importance for oil and gas companies. The monitoring of drilling operations allows to prevent faults, save resources, and take care of environmental and eco-planning businesses. The accuracies of MI-CG-OPF and MI-*k*NN-OPF were relatively similar to each other and with a minimal difference to compressed representations.

This work showed the viability of OPF-based classifier for MI-based problems with competitive accuracies and low average training times. Concerning feature works, we intend to evaluate other approaches to compute the instance that is going to represent the bag other than the mean feature vector of its instances.

Chapter 8

Enhancing Brain Storm Optimization Through Optimum-Path Forest

This chapter presents the work of Afonso et al. (AFONSO; JUNIOR; PAPA, 2018), which proposes a modification in the traditional BSO algorithm through the replacement of *k*-means by the OPF algorithm regarding the clustering of solutions. The work was accepted for presentation at the IEEE 12th International Symposium on Applied Computational Intelligence and Informatics, 2018 (Qualis-CC B2).

8.1 Introduction

Several interesting meta-heuristic optimization algorithms have been proposed inspired by many different natural events, being the population- or swarm-based ones the most widely used. They model the interaction and exchange of information within a group of objects or living beings to achieve a common goal (i.e., optimum solution). Among the proposed ones, there are the biologically-inspired ones, such as Particle Swarm Optimization (EBERHART; KENNEDY, 1995), Ant Colony Optimization (DORIGO; CARO, 1999), Artificial Bee Colony (KARABOGA; BASTURK, 2007), physics-based ones such as the Big Bang-Big Crunch (EROL; EKSIN, 2006) and Charged System Search (KAVEH; TALATAHARI, 2010). Also, we must refer to those inspired by the human behavior, such as the Human Behavior-based optimization (AHMADI, 2016) and the Brain Storm Optimization (SHI, 2011a), among others.

The human-inspired ones are considered to outperform those motivated by animals' behavior due to the superior intelligence of humans. The BSO is also a swarm-based algorithm that simulates the creative human brainstorming process to solve problems (SHI, 2011a; CHENG et al., 2016). The primary motivation behind the BSO algorithm is the social behavior of human beings when gathering a group of persons for brainstorming to face problems. Brainstorming eases the search for a good solution, and a problem is more likely to be solved if the group is comprised of people with different expertizes (ZHAN et al., 2012). The model of Shi and colleagues has shown its effectiveness in many optimization applications, such as (JADHAV et al., 2012; JORDEHI, 2015), just to mention a few.

Nonetheless, a few works available in the literature aimed at improving the BSO search process by proposing variations in its search strategies, which can be divided into three main steps: (i) solution clustering, (ii) new solution generation, and (iii) selection of the best solution. Zhan et al. (ZHAN et al., 2012) proposed replacing the *k*-means algorithm by a simple grouping method to reduce the computational burden related to the clustering solution step. Chen et al. (CHEN; XIE; NI, 2014) proposed an approach based on uncertainty information that applies the affinity propagation clustering to analyze the clusters' variations over the iterations. Duan and Li (DUAN; LI, 2015) improved the population diversity by applying a quantum-driven mechanism to prevent individuals getting trapped in local optima, and Cheng et al. (CHENG et al., 2014) studied two kinds of partial re-initialization solutions strategies to enhance population diversity as well.

Our work focuses primarily on the solution clustering strategy. This step aims to converge the solutions into small regions and refine the search area. Depending on the number of clusters, the ability of either exploitation or exploration can be enhanced. However, selecting the number of clusters and the clustering algorithm can be paramount for obtaining better results in any problem, as shown in Afonso et al. (AFONSO et al., 2012). Furthermore, choosing the best number of clusters may require prior knowledge of the problem, which may represent a disadvantage when applying algorithms such as *k*-means.

This work proposes a modification in the traditional BSO algorithm through the replacement of *k*-means by the OPF$_{uns}$ algorithm regarding the clustering of solutions. The technique has a single parameter that requires much less knowledge about the problem than *k*-means. Additionally, OPF$_{uns}$ computes the number of clusters on-the-fly, which is an interesting skill when working with applications where one does not know such information and wants to find it out. Although OPF has been employed in many applications, to best of our knowledge, it has never been applied to this context up to date. The proposed approach is evaluated using six different benchmarking functions whose results are compared against the traditional BSO and a variant that clusters solutions using the Self-Organizing Maps (SOM) (KOHONEN, 2001) . Additionally, we provide a comparison of computational load concerning the three approaches.

The remainder of this paper is organized as follows. Section 8.2 introduces the BSO algorithm. The methodology and experimental setup are described in Section 8.3. The experimental results are presented in Section 8.4, and conclusions and future works are stated in Section 8.5.

8.2 Brainstorm Optimization

The Brainstorm Optimization is a swarm-oriented meta-heuristic technique based on the human brainstorming process (SHI, 2011a), since it is a well-known fact that people are good at solving problems when they get together and share different ideas.

Generally, the original approach comprises three main strategies: (i) clustering, (ii) new individual generation (i.e., new solution), and (iii) selection. The clustering process aims at grouping similar solutions into small/compact regions, thus reducing redundancy and similar individuals. The original BSO employs the well-known k-means in such process, although Cheng et al. (CHENG et al., 2016) mentioned one could use any other clustering technique for such purpose.

Further, one should create a new individual based on some rules. First, a new solution can be created based on one or several individuals, as suggested by the original BSO, which defines a probability p_{gen} that is used to determine whether a new solution will be generated by one or two other individuals. The interesting point is related to the following: if one produces a new individual from one cluster, it can enhance local solutions (exploitation); on the other hand, generating a new solution based on two clusters can place it too far from these clusters, but favoring the exploration ability. The original approach also defines two more variables, i.e., $p_{oneCluster}$ and $p_{twoCluster}$, which stand for the probability of creating a new solution based on only one or two clusters, respectively.

Let $\mathbf{x} \in \Re^n$ be a possible solution in a problem with n features, and $\mathscr{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ be the search space with m possible solutions. After the clustering step, BSO generates a new individual for each possible solution depending on some rules, which may consider just copying the best solution or any other solution from a cluster, or even combining solutions from two different clusters. Such process is ruled out by the probabilities p_{gen}, $p_{oneCluster}$, and $p_{twoCluster}$.

The naïve Brainstorm Optimization technique based on the approach described by El-Abd (EL-ABD, 2016) begins initializing all possible solutions within the range $[L^j, U^j]$, where L^j and U^j stand for the lower and upper bounds concerning decision variable j, respectively. The main iteration is comprised of the following steps:

- Clustering of all possible solutions (ideas) using k-means: the clustering of solutions simulates the process of grouping similar ideas. The best idea of a given cluster is defined as its cluster center. Although the usage of k-means is mentioned, one can use any other clustering technique for such purpose.

- Generating new individuals: the strategy to generate new individuals is defined according to the values of probability p_{gen}, $p_{oneCluster}$ and $p_{twoCluster}$. Firstly, it is generated a random number using a uniform distribution, $r \sim U(0,1)$. If $p_{gen} \geq r$, the new individual $\hat{\mathbf{x}}_i$ is generated from a solution $\mathbf{x}_z$, which is either the best solution (i.e., if $p_{oneCluster} \geq r$) or any other solution (i.e., the otherwise) of a single randomly selected cluster c, $z = 1, 2, \ldots, m$ and $c \neq i$. Otherwise, $\hat{\mathbf{x}}_i$ will be the convex combination of two solutions $\mathbf{x}_{c_1}$ and $\mathbf{x}_{c_2}$ from the randomly selected clusters c_1 and c_2, respectively. If $p_{twoCluster} \geq r$, $\mathbf{x}_{c_1}$ and $\mathbf{x}_{c_2}$ will be the best solutions of the clusters; otherwise, they will be any other solution. Notice that r is always assigned to a new randomly generated number prior to the mentioned verifications. The convex combination is defined as follows:

$$\hat{\mathbf{x}}_i \leftarrow r\mathbf{x}_{c_1} + (1-r)\mathbf{x}_{c_2}. \tag{8.1}$$

- Creating the new solution: the solution of $\hat{\mathbf{x}}_i$ is defined as follows:

$$\hat{\mathbf{x}}_i^j = \hat{\mathbf{x}}_i^j + r_1\phi(t), \tag{8.2}$$

where $\mathbf{x}_i^j$ denotes the j^{th} decision variable of solution $\mathbf{x}_i$, $r_1 \sim U(0,1)$, and t stands for the time step (iteration number). Additionally, $\phi(t)$ can be computed as follows.

$$\phi(t) = r_2\sigma\left(\frac{0.5T - t}{s}\right), \tag{8.3}$$

where $r_2 \sim U(0,1)$ stands for a randomly generated number within a uniform distribution $[0,1]$, σ is the logistic sigmoid function, and T is the total number of iterations.

- Evaluating the new solution: after computing the solution for the new temporary individual $\hat{\mathbf{x}}_i$, its new fitness value $f(\hat{\mathbf{x}}_i)$ is compared against its current solution $f(\mathbf{x}_i)$. If the new individual's fitness value is better than the current one (i.e., lower), $\hat{\mathbf{x}}_i$ is assigned as the new best solution $\mathbf{x}_i$. At the end of all iterations, the outcome of BSO algorihtm will be the best solution among all.

8.3 Methodology

As mentioned, this work introduces the OPF algorithm for clustering ideas in the BSO algorithm. The ideas are modeled as samples of a graph that is partitioned using a competitive process that ends up in a set of trees (forest), in which each tree stands for a different cluster.

8.3.1 Benchmarking Functions

To evaluate both the efficiency and effectiveness of the OPF, we employed a wide variety of functions that include multimodal, separable, non-separable, differentiable, non-convex, or even continuous functions. We selected six different benchmarking functions whose main characteristics are listed below:

- Alpine 1st (f_1) - differentiable, non-separable, non-convex and multimodal;
- Lévy (f_2) - continuous, differentiable and multimodal;
- Pathological (f_3) - multimodal;
- Quintic (f_4) - multimodal;
- Salomon (f_5) - continuous, differentiable, non-separable, multimodal and non-convex; and
- Xin-She Yang #1 (f_6) - separable.

Table 8.1 provides more details of the selected functions. The first column stands for the names of each function, the *formulation* and *bounds* columns stand for their mathematical formulations and the lower and upper bounds of their variables, respectively, and the $f(x^*)$ column for their optimum values.

8.3.2 Experimental Setup

For comparison purposes, the results achieved by the application of OPF were compared against the traditional BSO algorithm and its modification that applies the SOM algorithm. Tables 8.2 and 8.3 depict the parameter configuration of SOM and BSO, respectively. Notice that such values were empirically set up.

Table 8.1: Benchmarking functions.

Identifier	**Formulation**	**Bounds**	$f(x^*)$
Alpine 1st	$f_1(x) = \sum_{i=1}^{n} \lvert x_i sin(x_i) + 0.1x_i \rvert$	$-100 \leq x_i \leq 100$	0
Lévy	$f_2(x) = \sin^2(\pi w_1) + \sum\limits_{i=1}^{D-1} (w_i - 1)^2[1 + 10\sin^2(\pi w_i + 1)] +$ $+(w_d - 1)^2[1 + \sin^2(2\pi w_d)]$, where $w_i = 1 + \frac{x_i - 1}{4}$	$-10 \leq x_i \leq 10$	0
Pathological	$f_3(x) = \sum_{i=1}^{n-1} \frac{\sin^2(\sqrt{100x_{i+1}^2 + x_i^2}) - 0.5}{0.001(x_i - x_{i+1})^4 + 0.5}$	$-100 \leq x_i \leq 100$	0
Quintic	$f_4(x) = \sum_{i=1}^{n} \lvert x_i^5 - 3x_i^4 + 4x_i^3 + 2x_i^2 - 10x_i - 4 \rvert$	$-10 \leq x_i \leq 10$	0
Salomon	$f_5(x) = 1 - \cos(2\pi\sqrt{\sum_{i=1}^{D} x_i^2} + 0.1\sqrt{\sum_{i=1}^{D} x_i^2})$	$-100 \leq x_i \leq 100$	0
Xin-She Yang #1	$f_6(x) = \sum\limits_{i=1}^{D} \varepsilon_i \lvert x_i \rvert^i$	$-5 \leq x_i \leq 5$	0

Table 8.2: Parameters used to set up the SOM algorithm.

Parameter	**Value**
Neuron map size	15×15
Neuron map type	planar
Neighborhood function	Gaussian
Epochs	100
Initial radius	10
Final radius	1
Learning rate cooling strategy	linear
Radius cooling strategy	linear

Table 8.3: Parameters used to set up the BSO algorithm.

Parameter	**Value**
$p_{oneCluster}$	0.8
$p_{twoCluster}$	0.5
p_{gen}	0.4
k	dynamic
# iterations	500

BSO has the characteristic of computing the same number of clusters to all iterations. However, the number of clusters computed by OPF varies even for a fixed value for $k_{\max}$, which was set as 75 in our experiments. Hence, a fair comparison among the approaches is performed by setting *k*-means and SOM to use the same number of clusters found by OPF at each iteration. We also evaluated three different numbers of ideas (agents) for each function ([$1,000$, $1,500$, and $2,000$]) with 15 decision variables.

Regarding the source-codes, we used the implementation of the BSO from LibOPT (PAPA et al., 2017), our own implementation of *k*-means and SOM algorithms, and the OPF implementation from the LibOPF (PAPA; SUZUKI; FALCÃO, 2014).

8.4 Experimental Results

As mentioned earlier, the experiments were carried out over a set of six benchmarking functions. The evaluation was performed over 20 runs to analyze the best fitness value (BF) , the mean of best fitness value (MBF) , and the standard deviation of BF (SDBF) where the lowest value is the best one since we are willing to minimize the loss. The best MBF values for each configuration (i.e., [benchmarking function, number of ideas]) are chosen according to the Wilcoxon signed rank test (WILCOXON, 1945) with a significance of 5% and highlighted in bold. The BF values are underlined as well, but we did not perform any statistical evaluation on them.

Tables 8.4– 8.9 present the results concerning the six benchmarking functions. Regarding the MBF values, OPF was able to provide the best ones in all configurations. The three clustering methods achieved statistically similar results in two out of the three configurations of the Alpine 1st function. The OPF-based approach presented similar results against *k*-means on all configurations of the Pathological function, and on Salomon function against SOM.

Although outperforming *k*-means and SOM in terms of MBF values in all configurations, OPF obtained the best BF in the majority of the configurations (12 out of 18). Figure 8.1 depicts the clustering time of each technique on the optimization of the Alpine 1st function. OPF figured as the slowest approach since we consider the fine-tuning process to find $k \in [1, k_{max}]$. Also, *k*-means was the fastest one due to its simplicity.

Table 8.4: Alpine 1^{st} Benchmarking function

# ideas	*k*-means		SOM		OPF	
	BF	MBF/SDBF	BF	MBF/SDBF	BF	MBF/SDBF
1,000	19.065	**40.362±20.931**	17.494	**37.772±15.014**	<u>10.465</u>	**33.277±16.618**
1,500	<u>11.219</u>	**30.219±13.792**	15.544	**31.226±17.436**	19.580	**29.479±9.779**
2,000	10.643	**22.797±7.966**	14.513	22.875±7.171	<u>10.133</u>	**19.383±5.199**

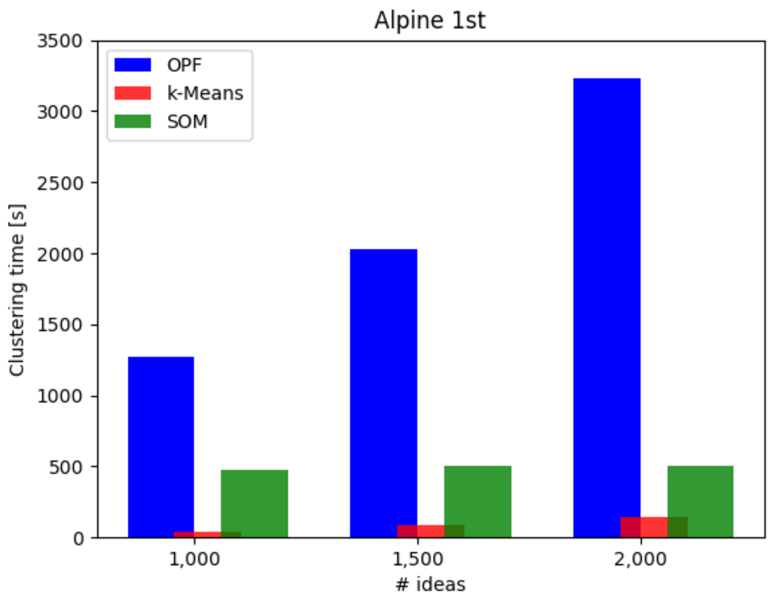


Figure 8.1: Computational load concerning the clustering step.

Table 8.5: Lévy Benchmarking function

# ideas	*k*-means		SOM		OPF	
	BF	MBF/SDBF	BF	MBF/SDBF	BF	MBF/SDBF
1,000	5.102	16.265±8.679	6.880	14.889±4.606	<u>3.692</u>	**10.250±6.041**
1,500	5.559	13.234±6.018	5.097	13.602±6.718	<u>2.530</u>	**9.603±7.278**
2,000	3.700	11.031±5.524	4.961	10.320±4.220	<u>2.560</u>	**6.168±4.400**

Table 8.6: Pathological Benchmarking function

# ideas	*k*-means		SOM		OPF	
	BF	MBF/SDBF	BF	MBF/SDBF	BF	MBF/SDBF
1,000	7.000	**7.000±9.735e-06**	7.000	7.000±9.832e-06	7.000	**7.000±5.967e-06**
1,500	7.000	**7.000±3.872e-06**	7.000	7.000±6.219e-06	7.000	**7.000±5.921e-06**
2,000	7.000	**7.000±3.313e-06**	7.000	7.000±6.463e-06	7.000	**7.000±3.658e-06**

Table 8.7: Quintic Benchmarking function

# ideas	*k*-means		SOM		OPF	
	BF	**MBF/SDBF**	**BF**	**MBF/SDBF**	**BF**	**MBF/SDBF**
1,000	227.563	6,164.135±8,157.007	98.156	6,158.760±7,709.022	68.212	**2,622.902±3,396.080**
1,500	119.835	**3,522.185±4,520.314**	590.721	**3,222.114±2,978.550**	107.271	**2,850.884±2,884.257**
2,000	56.101	3,380.685±4,296.732	49.324	**2,294.584±3,820.453**	72.570	**832.068±975.838**

Table 8.8: Salomon Benchmarking function

# ideas	*k*-means		SOM		OPF	
	BF	**MBF/SDBF**	**BF**	**MBF/SDBF**	**BF**	**MBF/SDBF**
1,000	5.299	9.399±2.255	3.599	**8.195±3.222**	4.399	**7.080±1.632**
1,500	4.099	7.789±2.167	2.599	**6.930±2.806**	3.099	**5.955±2.262**
2,000	3.199	6.555±1.667	2.499	**4.955±1.960**	2.699	**4.675±1.523**

Table 8.9: Xin-She Yang #1 Benchmarking function

# ideas	*k*-means		SOM		OPF	
	BF	**MBF/SDBF**	**BF**	**MBF/SDBF**	**BF**	**MBF/SDBF**
1,000	0.403	26.499±71.960	0.789	**5.017±5.135**	0.049	**3.743±5.345**
1,500	0.194	16.436±39.004	0.176	9.965±26.194	0.026	**1.407±3.004**
2,000	0.027	3.793±4.797	0.073	**2.965±7.744**	0.044	**1.112±1.806**

Figures 8.2, 8.3 and 8.4 depict the convergence plot concerning the Lévy function. It can be observed the BSO convergence is faster at the very first iterations (i.e., between the first and fifth iterations) for any of the clustering algorithms. Notice the application of OPF enabled to reach the lowest fitness values in all three configurations.

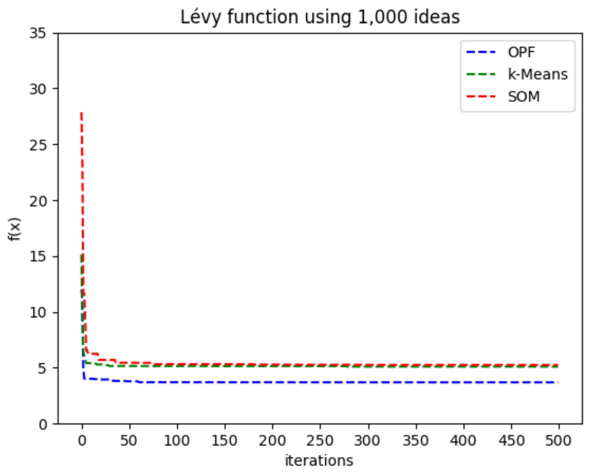


Figure 8.2: Convergence plot concerning the Lévy function for a set of $1,000$ ideas.

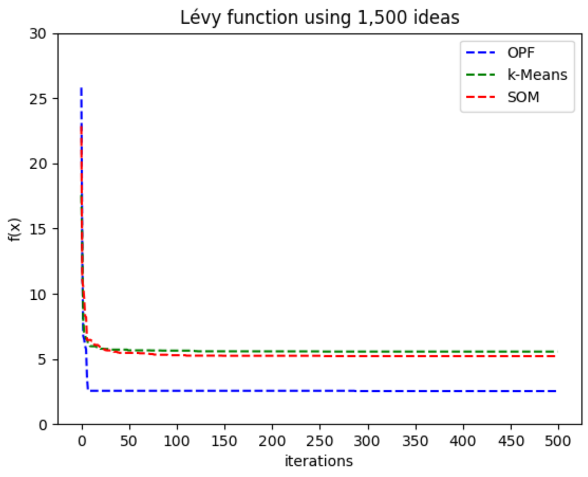


Figure 8.3: Convergence plot concerning the Lévy function for a set of $1,500$ ideas.

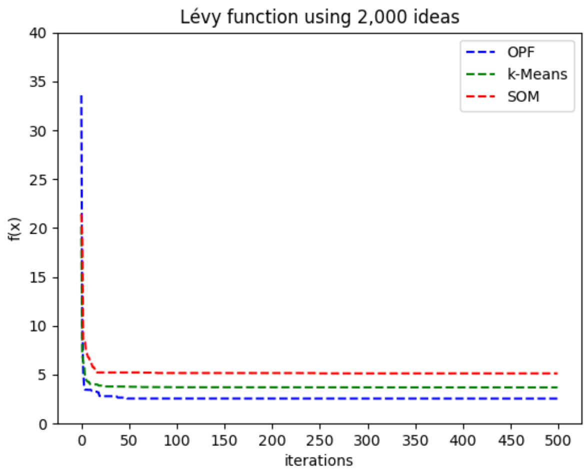


Figure 8.4: Convergence plot concerning the Lévy function for a set of $2,000$ ideas.

The OPF behavior along the 500 iterations in each of the six benchmarking functions is depicted in Figure 8.5. One can observe how the number of clusters varies along time and may present significant changes, such as of the Pathological function between iterations 300 and 350, and of Salomon function between iterations 450 and 500. Such behavior refers when we use $2,000$ ideas, that is when OPF achieved the lowest MBF values.

8.5 Conclusions and Future Works

This paper introduced the Optimum-Path Forest classifier applied to the clustering step of the Brain Storm Optimization algorithm. The main contributions of our work are twofold: (i) to enhance BSO using OPF, and (ii) to vary the number of ideas per iteration concerning BSO. These improvements were observed through experiments over six different benchmarking

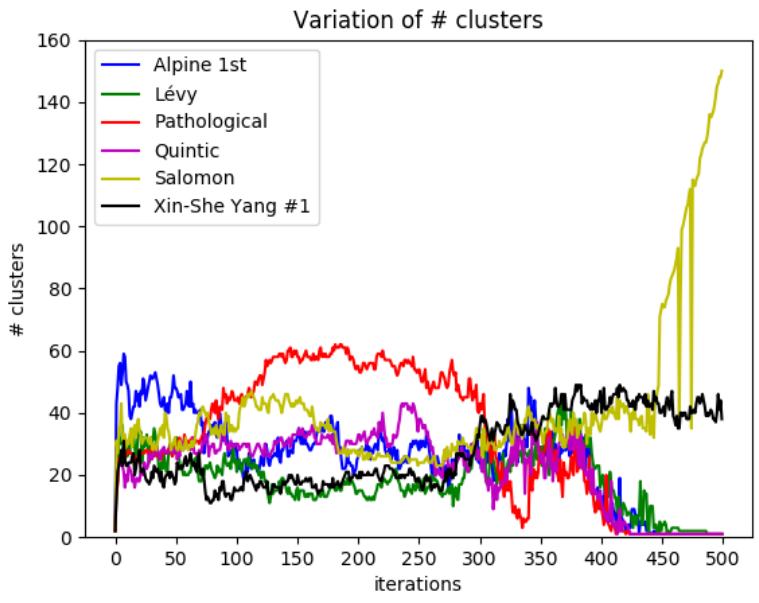


Figure 8.5: Variation of the number of clusters along the iterations considering the benchmarking functions adopted in this work.

functions with results compared against traditional BSO algorithm (i.e., clustering performed by *k*-means) and a variation of BSO that makes use of the well-known Self-Organizing Maps.

Despite its computational cost, OPF showed the best average minimum fitness values over the traditional and SOM-based approaches on all experimental configurations and obtained the best fitness value in the majority of the configurations.

As a future work, we aim at hybridizing BSO with other meta-heuristic techniques and to evaluate other unsupervised techniques to clustering ideas.

Chapter 9

Learning to Classify Seismic Images with Deep Optimum-Path Forest

This chapter presents the work proposed by Afonso et al. (AFONSO et al., 2016) and presented at the 29th Conference on Graphics, Patterns and Images (SIBGRAPI), 2016 (Qualis-CC A3). The proposed work introduces a deep architecture based on the Optimum-Path Forest for unsupervised classification.

9.1 Introduction

Image classification plays an important role in several application domains, which range from medical image analysis to remote sensing-driven tools. However, the lack of labeled data has oriented researchers towards active learning-based techniques, which consider the user feedback to improve the classification process by labeling samples. Although promising results have been obtained in the last years, labeling images is time-consuming and it strongly depends on the human skills, which can be prone to errors as well.

The Big Data era has made available tons of digital content daily, making even more tedious the task of analyzing data by hand. In this context, a foreseeable future can be drawn: we shall not be in lockstep with the amount of data generated, thus paying the price of having important information discarded and/or meaningless. One of the first waves towards the lack of labeled data refers to the so-called deep learning, which basically ends up learning features in an unsupervised fashion (LECUN; BENGIO; HINTON, 2015).

Therefore, unsupervised learning has gained attention in the last years once more, but still posing a tougher challenge than supervised learning, since the notion of a cluster can somehow be doubted and personally-driven. In this scenario, a number of techniques can be highlighted, such as *k*-means (MACQUEEN, 1967), Mean-Shift (COMANICIU, 2003), Self-Organizing Maps (KOHONEN, 1982) and others (JAIN; MURTY; FLYNN, 1999), just to name a few. The so-called *k*-means works surprisingly well in many situations, despite its simplicity.

However, *k*-means has also some well-known shortcomings: (i) first, the user is required to feed the technique with the number of clusters, and (ii) the problem itself is essentially an optimization task, in which the distance of each sample to its nearest center is minimized. For the first statement, although some scientists argue the parameter k can be seen as a meta-parameter, in fact, it requires us to have some knowledge about the "unsupervised" problem, in which by definition we should not have any information so far. The second statement concerns with any non-convex optimization problem, which may get trapped from local optima. Although a number of works have focused on such drawbacks, there is still room for improvements, since there is no "exact solution" to the problem, which means approximations that may cost some computational burden can be derived and thus employed by researchers worldwide.

Graph-based clustering techniques have their appeal as well. Roughly speaking, the idea is to encode each feature vector as a graph node, and then to learn some connectivity function that can group "similar" samples and turn others far apart. Notice the notion of "similarity" also poses an interesting problem, which can be of extreme importance to the success of the technique. Some years ago, Rocha et al. (ROCHA; CAPPABIANCO; FALCÃO, 2009) presented the unsupervised version of the Optimum-Path Forest classifier, which models the problem of clustering data as a competition process, in which some key samples compete among themselves in order to gather others. OPF has gained considerable attention in the last years, since its supervised version has been similarly accurate as Support Vector Machines for some applications, but faster for training (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012).

Unsupervised OPF, hereinafter called OPF, has one parameter only (i.e., k_{max}), which requires much less knowledge than *k*-means with respect to the problem itself. Additionally, OPF computes the number of clusters on-the-fly, which is an interesting skill considering applications where one does not know that information and wants to find it out. Some examples are related to data representation using bag-of-visual-words, in which the size of the dictionary (i.e., the number of visual words) is of extreme importance to the success of the technique. Usually, the user does not have such information but is eager to discover it. Afonso et al. (AFONSO et al., 2012) have successfully employed OPF for such task. Last but not least, OPF can obtain similar clusters' centers when compared to *k*-means, which is an interesting property since the latter is undoubtedly recognized to work well in several problems (ROSA et al., 2014).

However, a weakness of OPF is directly related to its strength: "what if one does know in advance the number of clusters?" In fact, we have no information about any OPF-related paper that deals with that problem. Actually, if one needs to somehow control the number of clusters, we can play with k_{max} until that desired number is reached. Moreover, there is no guarantee of that. Roughly speaking, one can deal with that by experimentally trying different values for k_{max} within the range $[1, k_{max}]$ that can reach or, at least, to be close as to the desired number of clusters. Nonetheless, to try out all possible values within that range, as proposed by Rocha et al. (ROCHA; CAPPABIANCO; FALCÃO, 2009), might be prohibitive. Costa et al. (COSTA et al., 2015) modeled this problem as a meta-heuristic-based optimization task, being the results quite close to the ones obtained by the optimal approach proposed by Rocha et al. (ROCHA; CAPPABIANCO; FALCÃO, 2009), but being faster. However, it is noteworthy to mention the aforementioned works were proposed to find suitable values for $k^* \in [1, k_{max}]$ to create the k^*-neighborhood, and not to establish a proper number of classes[1], although the design of the neighborhood size directly influences the number of clusters.

In this work, we propose to perform OPF clustering at different levels of abstractions (i.e., scales) in a deep-driven approach to be as closest as possible to the number of clusters required by that specific application. By deep we mean we are going to perform unsupervised learning using different "views" of the data until we may reach some desirable result. As aforementioned, OPF makes use of key samples, hereinafter called *prototypes*, which compete among themselves trying to offer the "best" reward (path-cost function) to the remaining samples. By taking into account the prototypes chosen at the very first (initial) step, we can thus use them as the new (and only) samples to the next clustering step. Since the number of prototypes is often much smaller than the number of dataset samples, we have a more compressed representation of the dataset at each level, and thus fewer clusters (in fact, each cluster is represented by one prototype). In this paper, we show it is possible to obtain the desired number of clusters (or at least to be close to that) using few scales of representation. Such methodology is much faster than playing around with the values of k_{max}, as proposed in the works by Rocha et al. (ROCHA; CAPPABIANCO; FALCÃO, 2009) and by Costa et al. (COSTA et al., 2015).

A detailed look at the papers published in the last years has revealed only one work similar to ours (CASTELO-FERNÁNDEZ; CALDERÓN-RUIZ, 2015), but still with a different purpose. This work employed the OPF clustering for automatic video summarization, using the prototypes obtained in the first step as the shots and key frames to represent a reduced version of the video. Soon after, these key frames are clustered again to obtain more refined representations of the

[1]In this context, k^* stands for the neighborhood size that minimizes some fitness function (i.e., the minimum graph cut in that case).

video summary. But clearly, we are interested in working on the problem of restricting OPF to a predefined (desired) number of clusters. Another contribution of this work is to evaluate OPF in the context of seismic-based images concerning the task of hydrocarbon accumulation detection, which is used to detect possible locations for petroleum exploration. As far as we know, OPF has never been used in this context so far. This work also explores the proposed OPF clustering in a wider context by applying it in three large labeled datasets. The remainder of this paper is organized as follows. Section 9.2 presents the proposed deep-based approach to obtain different levels of representations, and thus fewer clusters. The methodology and experiments are discussed in Section 9.3, and Section 9.4 states conclusions and future works.

9.2 Learning Deep Representations

As aforementioned, OPF can find the number of clusters on-the-fly, which means there is no need for such information beforehand. However, to the best of our knowledge, there is no proposed approach that can somehow "force" the number of clusters to a predefined number when one knows that information when dealing with OPF. Although we can play around with k_{max}, it can be prohibitive for large datasets, such as the one addressed in this work (high-resolution seismic images).

To cope with this challenge, we propose to employ different representations (layers) of the dataset samples, being the first layer the original feature space to be clustered. After that, the prototypes at the first layer are then used as the new samples to compose the feature space at the second layer, which is clustered once again. The very same process is repeated until the predefined number of clusters (or at least close to) is reached. Since the OPF clustering prototypes are located in the highest density regions, they are very suitable to represent nearby samples, as argued in the works conducted by Castelo and Calderón-Ruiz (CASTELO-FERNÁNDEZ; CALDERÓN-RUIZ, 2015) and Afonso et al. (AFONSO et al., 2012).

Let $\mathscr{P}_i$ be the set of prototypes at layer B_i, $i = 1, 2, \ldots, l$, where l stands for the number of layers. Since each root will be the maximum of a pdf (Equation 2.4), we have a set of samples that fall in the same optimum-path tree and are encoded by the very same prototype (the root of that tree) in the next layer. In short, the higher the number of layers, the less prototypes (clusters) one shall find, i.e., $|\mathscr{P}_1| < |\mathscr{P}_2| < \ldots < |\mathscr{P}_L| < \ldots \leq 1$. Therefore, at a very coarse layer, one shall find only one cluster. Figure 9.1 displays the proposed OPF-based architecture for deep-driven feature space representation.

At layer B_1, we can observe four clusters (optimum-path trees), where the black nodes

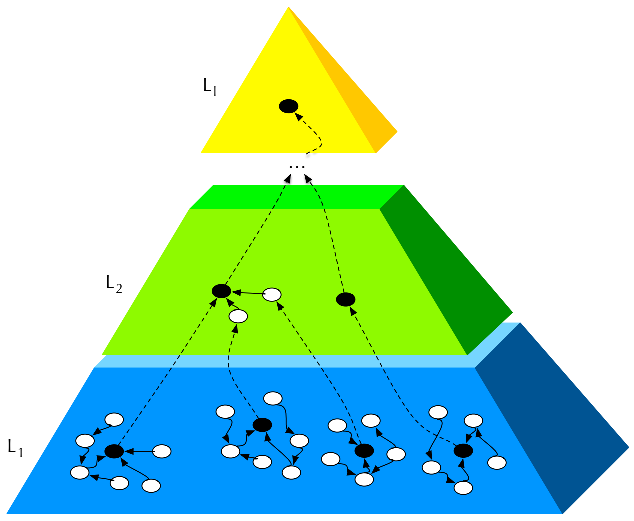


Figure 9.1: Proposed approach based on coarser representations of the feature space.

stand for the set of prototypes at that layer, i.e., $\mathscr{P}_1$. Some of these prototypes will become new prototypes at B_2, and others not (we can observe both black and white nodes at layer 2). This process is carried out up to the last layer specified by the user. Notice at the very coarser scale, i.e., B_l, we shall find only one cluster.

9.3 Methodology and Experimental Results

To provide both qualitative and quantitative insights into the OPF-driven approach, we divided the experiment section in two. In the first part, OPF is applied in a set of seismic images enabling to visualize the clustering result. The second part makes use of large labeled datasets so we can obtain some metrics that indicate the OPF clustering quality.

9.3.1 Seismic Images

To validate the proposed OPF-driven approach to obtain finer representations of the clustered space, we used seismic images from the North Sea at a specific location in the Dutch sector, the so-called "Netherlands Offshore F3 Block Complete" dataset[2]. The dataset has around 466 images (i.e., slices) that are combined together to form a volume that somehow models the geological information of the aforementioned location. In this paper, we used 5 images chosen at random for clustering purposes, say that: 924, 928, 932, 936 e 940. Note these numbers stand

[2]`https://opendtect.org/osr/pmwiki.php/Main`

for the acquisition time in milliseconds of each image. Figure 9.2 depicts image 924, where the colors stand for different layers of rocks or their compactness in the sea floor, the green arrow points North and X1 stands for In-line.

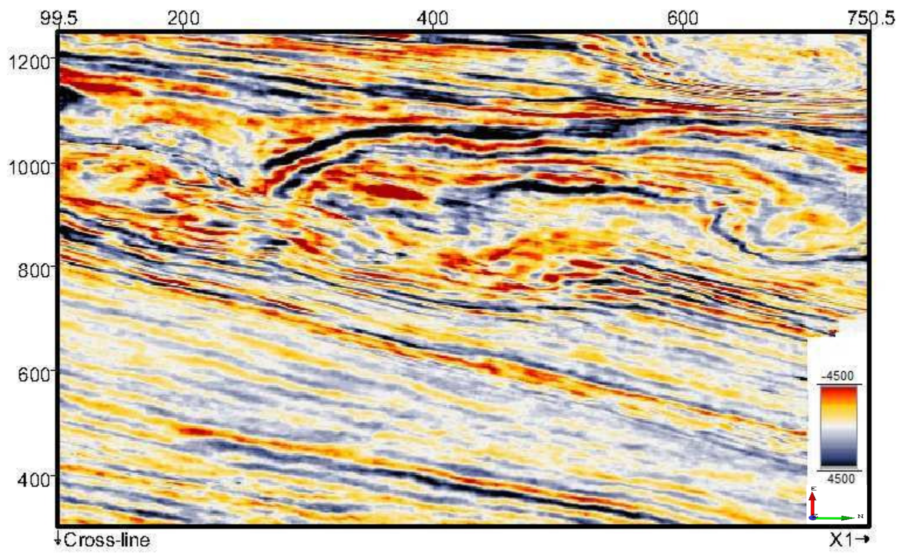


Figure 9.2: Figure 924 of the F3 Block at the Dutch sector.

Each dataset sample is composed of a pixel from the aforementioned images, thus resulting in 5 different datasets, where each pixel is represented by the seismic amplitude. Note the color intensities depicted in Figure 9.2 were used for the sake of visualization purposes only. After that, we then employed OPF with 4 layers against the well-known k-means and SOM for evaluation reasons. Aiming a fair comparison among the techniques, we used the very same number of clusters found by OPF at the last layer as the input to both k-means and SOM[3].

Notice the k_{max} value is strongly related to the number of desired clusters, i.e., the larger k_{max} value, the fewer clusters one shall have. The rationale behind that idea is related to the working mechanism used by OPF to find proper neighborhood sizes (k values), as explained in Section 2.3. Since OPF performs a linear search within the range $k \in [1, k_{max}]$, if one uses larger k_{max} values we also increase the probability of finding larger values of k that minimize the graph cut criterion. Therefore, larger neighborhoods mean fewer clusters. As such, we employed different and decreasing values for k_{max} considering the different layers for all images. Considering the first and last layers, we used $k = 100$ and $k = 2$, respectively. Regarding the inner layers, we used 1% of the new dataset for layer 2, and 10% concerning layer 3. Notice the dataset size decreases in the proposed approach as we move towards the upper layers. Table 9.1 presents the number of clusters found by OPF considering different layers for all images employed.

[3]Recall that we employed k-means to label the SOM map after their learning.

Table 9.1: Number of clusters found by OPF at different layers.

Image	**Layer**			
	1	**2**	**3**	**4**
924	4,102	41	8	3
928	4,135	41	6	2
932	4,074	38	6	2
936	4,144	41	10	2
940	4,193	44	8	2

As aforementioned, one can observe the number of clusters decreases as we move to the upper layers. Clearly, one can obtain much fewer clusters by just using 2 layers. Additionally, it seems, at least for the images employed in this work, that 4 layers are enough to cope with the problem of seismic-driven image classification since we achieved the minimum number of clusters we can work with. Figures 9.3, 9.4 and 9.5 illustrate the classified images at layer 4 with respect to OPF, *k*-means and SOM techniques, respectively, where the different colors represent the different labels. Note these figures refer to the image 924 (Figure 9.2).

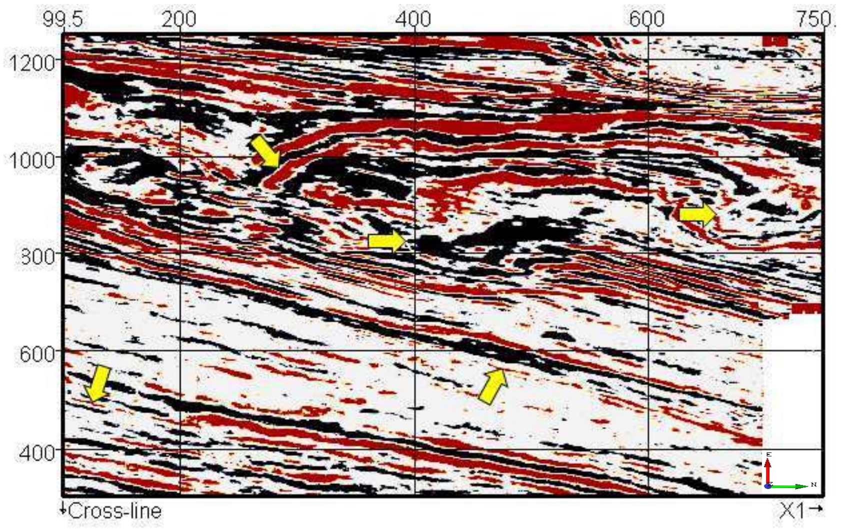


Figure 9.3: Classified image 924 **using OPF at layer** 4**.**

In order to provide a qualitative comparison among the techniques, we asked for a geologist to provide insightful comments about the results. In regard to OPF and SOM results, one can observe the border of the reservoir body (the leftmost arrow) is not visible in the SOM results, which means a negative impact for exploration purposes. Also, SOM was also unable to provide lateral continuity of reflections, i.e., some structures appear as a dashed-like line instead of a continuous-like line. On the other hand, OPF classification overcame such issues by providing some details of the interest region and other main structures, besides lateral continuity. Lateral continuity is important because it allows identifying the limits of a seismic body and any faults

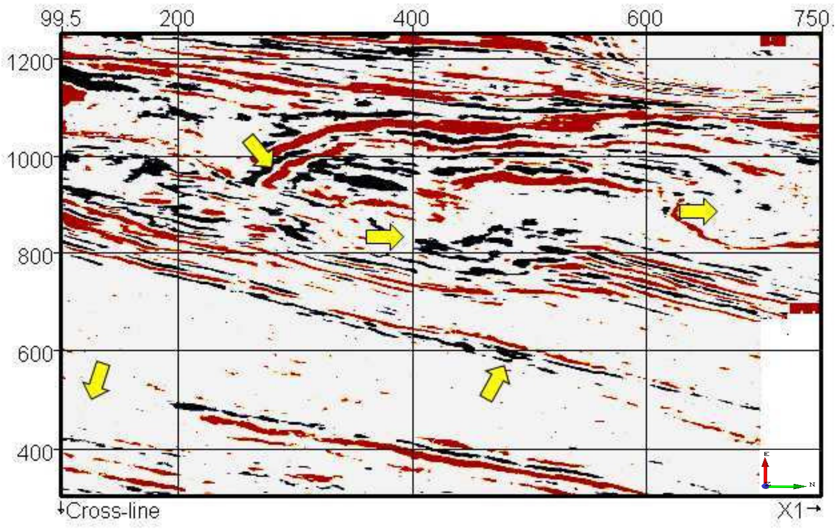


Figure 9.4: Classified image 924 **using** k**-means with** 3 **clusters.**

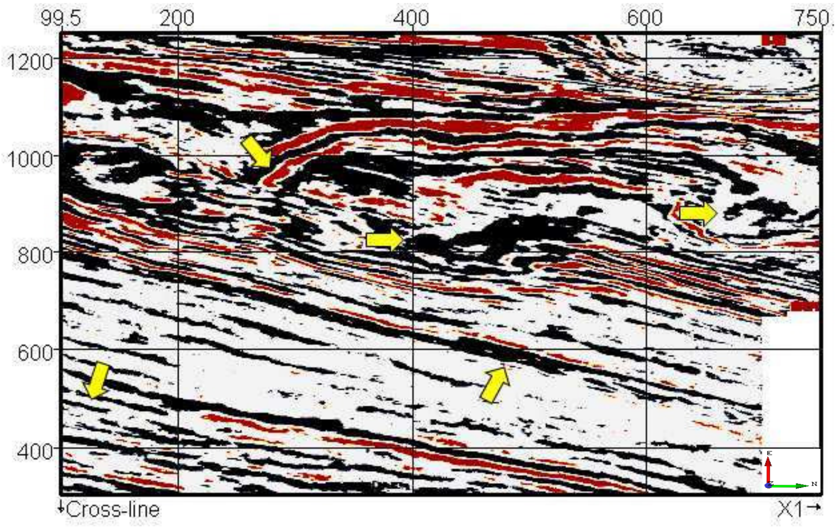


Figure 9.5: Classified image 924 **using SOM with** 3 **clusters.**

that may seal or conduct fluids off a reservoir. Finally, k-means was able to provide suitable levels of details, thus obtaining very good results as well.

9.3.2 General-purpose Images

This section aims to provide some quantitative insight concerning the proposed deep-driven OPF[4] by evaluating it against k-means[5], Mean-Shift[6] and SOM[7] over three large well-known labeled datasets:

[4]In regard to OPF implementation, we used LibOPF (PAPA; SUZUKI; FALCÃO, 2014).
[5]We used our own implementation.
[6]We employed an implementation provided by scikit-learn (PEDREGOSA et al., 2011).
[7]`http://somoclu.readthedocs.io/en/stable`

- CIFAR-10[8]: it consists of 60,000 32×32 images distributed in 10 classes, being $6,000$ images per class. The training set has $50,000$ images and the remaining $10,000$ images are used to compose the testing set. The experiments used all $60,000$ images as a single set.
- CIFAR-100[8]: This dataset is similar to CIFAR-10, and it contains $60,000$ images distributed in 100 classes, being 600 images per each. The 100 classes represent a "finer" label, and are grouped into 20 superclasses as a "coarser" label. The experiments used the $60,000$ images and the coarse label for evaluating the clustering techniques.
- MNIST[9]: this dataset has a total of $70,000$ images of handwritten digits divided in 10 classes (one class for each digit), in which $60,000$ belong to the training set and the remaining images belong to the testing set. The digits are size-normalized and centered in a fixed-size image. All $70,000$ were used for the experiments as single set.

Figures 9.6, 9.7 and 9.8 depict some examples from CIFAR-10, CIFAR-100 and MNIST datasets, respectively.

To describe the images from all aforementioned datasets, we employed the Border/Interior Pixel Classification (BIC) (STEHLING; NASCIMENTO; FALCÃO, 2002) technique, which is a 64-dimensional descriptor. The reason for using such descriptor relies on its compactness and low dimensionality since we are dealing with thousands of images to be clustered. As in Section 9.3.1, OPF is evaluated using a four-layer design, as well as following the same rules for setting the value of parameter k for each layer. Notice k-means and SOM used parameter k equals to the number of clusters found by the OPF on its last layer. The reason for using the very same value is to allow a fair comparison among the techniques[10].

Since Mean-Shift has no parameter k, its number of clusters is different from the other techniques in all datasets, but its clustering metrics are computed though. Mean-Shift found 4, 4 and 3 clusters in CIFAR-10, CIFAR-100, and MNIST datasets, respectively. Table 9.2 presents the number of clusters found by each layer of OPF considering each dataset. The very same number of clusters were computed by k-means and SOM. We considered that Mean-Shift did not obtain interesting results since it has found fewer clusters than desired. Since CIFAR-100 contains 100 classes, it is expected to find out 100 clusters at least. If one considers the results in Table 9.2, OPF has the flexibility to play around with a different number of layers

[8]`https://www.cs.toronto.edu/~kriz/cifar.html`

[9]http://yann.lecun.com/exdb/mnist/

[10]Notice parameter k has distinct meaning for OPF, k-means and SOM. We decided to keep the same notation for the sake of simplicity.

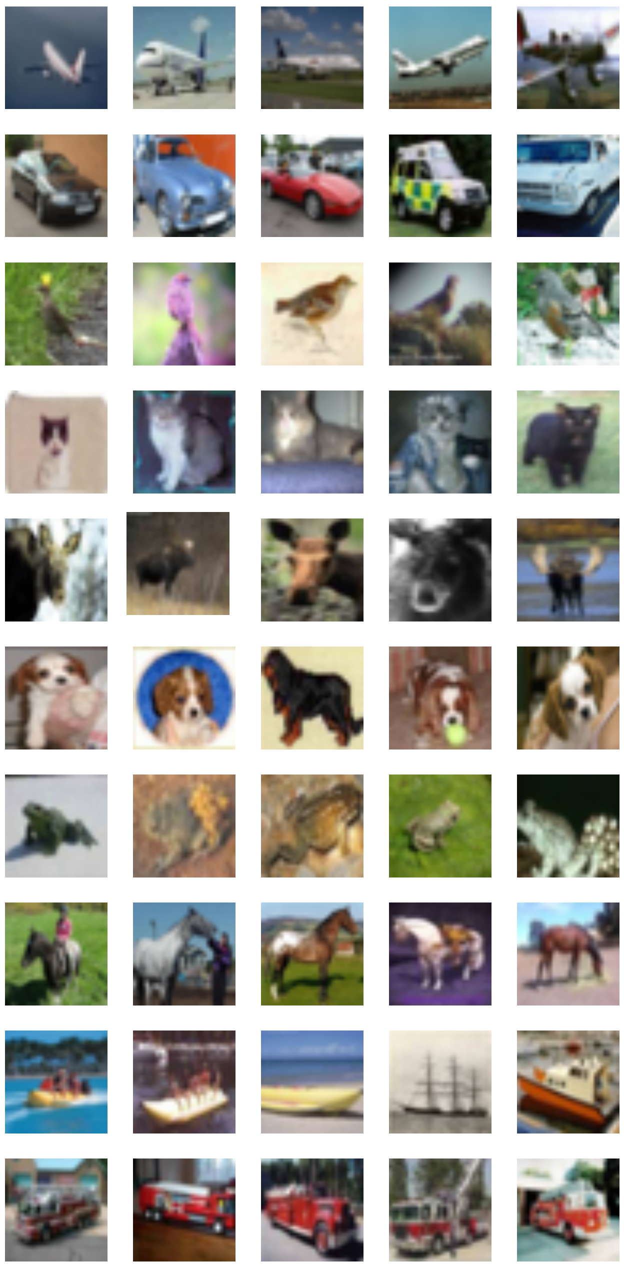


Figure 9.6: A few samples from CIFAR-10 dataset.

until a desired number of clusters has been found.

Table 9.2: Number of clusters found by OPF at different layers considering each dataset.

Dataset	Layer			
	1	**2**	**3**	**4**
CIFAR-10	137	121	17	8
CIFAR-100	216	163	24	15
MNIST	221	145	5	2

Since we have the true labels for each dataset, the overall performance is assessed by five

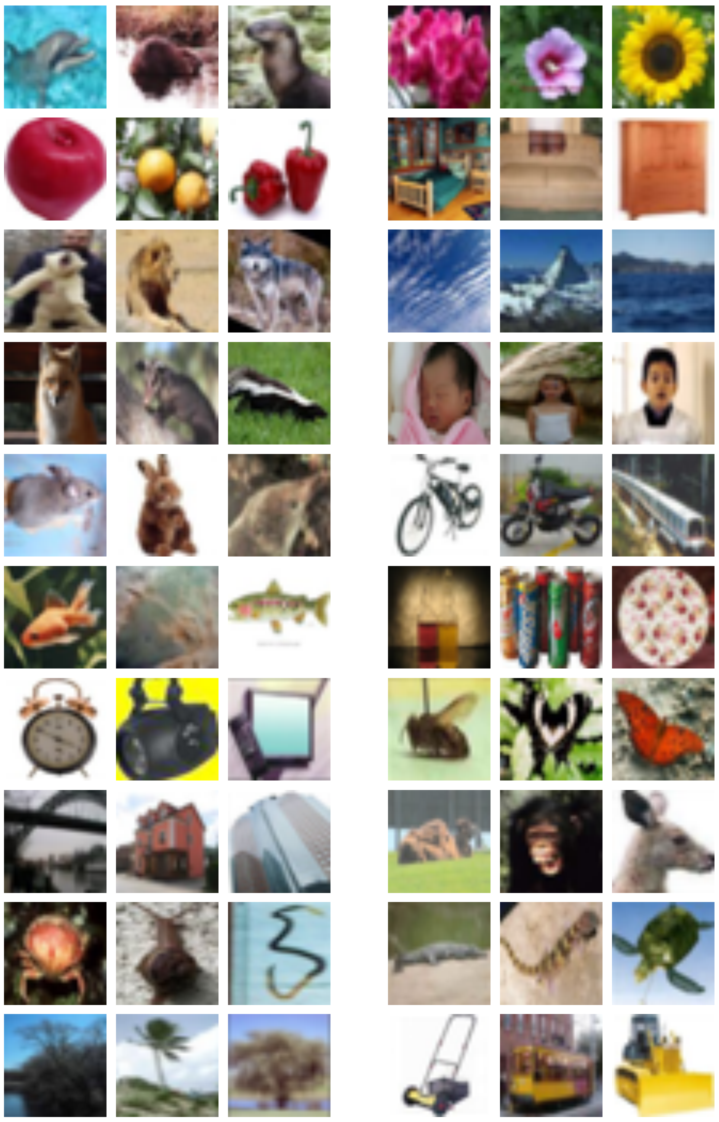


Figure 9.7: A few samples from CIFAR-100 dataset.

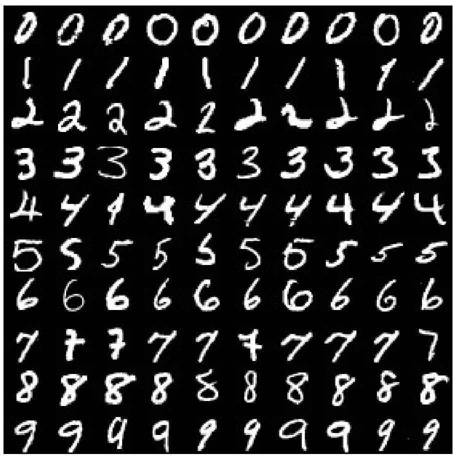


Figure 9.8: A few samples from MNIST dataset.

metrics[11] which use the true and predicted labels for computation purposes, as follows:

- Homogeneity (H) : this metric regards to how pure clusters are, in other words, clusters have a maximum value of homogeneity if they contain only samples that belong to the same class. Notice $H \in [0,1]$, where $H = 1$ denotes the best result.
- Completeness (C) : a clustering result satisfies completeness if all samples that are members of a given class are elements of the same cluster. Notice $C \in [0,1]$, where $C = 1$ denotes the best result.
- V-measure (V) : this metric is the harmonic mean between homogeneity and completeness, as follows:
$$V = 2 * \frac{(H * C)}{(H + C)}. \tag{9.1}$$
Notice $V \in [0,1]$, where $V = 1$ denotes the best result.

The results achieved by each algorithm are shown in Tables 9.3, 9.4 and 9.5. The best results are in bold. Notice these results consider the number of clusters found by OPF at the layer.

Table 9.3: Results for CIFAR-10 dataset.

Metric	Technique			
	OPF	*k*-means	Mean-Shift	SOM
H	0.000	**0.054**	0.001	0.049
C	**0.153**	0.060	0.039	0.056
V	0.000	**0.057**	0.001	0.052

Table 9.4: Results for CIFAR-100 dataset.

Metric	Technique			
	OPF	*k*-means	Mean-Shift	SOM
H	0.010	**0.033**	0.001	0.030
C	0.069	0.038	**0.077**	0.034
V	0.017	**0.035**	0.003	0.032

Since the number of clusters found in all datasets does not match the real number of classes, it is expected the clustering homogeneity of all techniques never reaches the maximum value. The *k*-means was able to find the most homogeneous clusters considering CIFAR-10 and CIFAR-100 datasets, and SOM obtained the best result concerning MNIST dataset. However,

[11] http://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation

Table 9.5: Results for MNIST dataset.

Metric	**Technique**			
	OPF	*k*-means	Mean-Shift	SOM
H	0.000	0.007	0.000	**0.073**
C	**1.000**	0.024	0.005	0.376
V	0.000	0.011	0.001	**0.122**

keep in mind all homogeneity values are pretty close to the minimum in all cases. Regarding the clustering completeness, OPF outperformed the other techniques in CIFAR-10 and MNIST datasets, achieving the maximum value in the latter one, thus meaning all samples for every class were grouped in the very same cluster. Since V-measure tries to balance both homogeneity and completeness, *k*-means obtained the best V values concerning CIFAR-10 and CIFAR-100, and SOM outperformed all techniques in MNIST dataset. With respect to the efficiency, *k*-means was the fastest one, followed by OPF, SOM, and Mean-Shift.

9.4 Conclusions

This paper presented a deep-driven approach that allows OPF to obtain coarser clustered images, being the problem of unsupervised learning decomposed in different layers. The proposed approach was evaluated in the context of seismic image classification, being its results comparable to the ones obtained by *k*-means and SOM techniques. In this specific case, OPF was able to provide visual details that are important to identify certain structures, where *k*-means and SOM were unable to highlight.

Considering general-purpose datasets, we can highlight OPF was able to found a number of clusters close to the real number of classes in CIFAR-10 (8 clusters found out of 10 classes) and CIFAR-100 (15 clusters found out of 20 classes) datasets. Although the numbers of clusters found in MNIST dataset is not close to the number of classes, OPF was able to cluster the whole dataset on either cluster 1 or cluster 2, thus achieving a completeness equals to 1. Regarding other techniques, *k*-means and SOM were able to find more homogeneous clusters in all situations.

The experimental section showed us the proposed deep-driven OPF allows the user a more flexible tool when working with unsupervised clustering-oriented applications where we know the desired number of clusters. Also, experiments over general-purpose datasets shed light over all techniques might be complementary to each other, since they obtained different results over distinct datasets and measures.

Chapter 10

PARKINSON'S DISEASE IDENTIFICATION THROUGH DEEP OPTIMUM-PATH FOREST CLUSTERING

This chapter introduces the Deep-hierarchical OPF clustering algorithm in the context of dictionary learning in a hierarchical way and evaluated in the automatic Parkison's disease application. The work was proposed by Afonso et al. (AFONSO et al., 2017) and published in the 30th Conference on Graphics, Patterns and Images (SIBGRAPI), 2017 (Qualis-CC A3). This work was selected and invited among others to submit an extention to the Special Issue of the Journal of Visual Communication and Image Representation as guest paper (Chapter 11).

10.1 Introduction

The cure for neurodegenerative diseases has been constantly researched by Medicine, mainly concerning Parkinson's disease (PD), which affects nearly 1 million people only in the United States, and around 7 to 10 million people might be living with PD worldwide. Also, the number of new cases diagnosed each year ranges between $50,000$ to $60,000$ individuals according to the National Parkinson's Foundation (FUNDATION, 2017). Parkinson's disease is characterized by motor dysfunctions; it is a chronic, progressive and multi-lesion disease caused by the loss of a neurotransmitter called *Dopamine* (LEES; HARDY; REVESZ, 2009). Such illness is usually diagnosed through a clinical exam by a neurologist with expertise in movement analysis. The PD is considered non-lethal, but people with PD have a shorter life expectancy than the general population.

[*] Both authors contributed equally.

More often in the elderly population, PD produces alterations in gait and posture that may increase the risk of falls and lead to mobility disabilities. As such, it impacts daily activities and reduces the quality of life concerning patients and their families (MAKI; MCILROY, 2005; MARCHETTI; WHITNEY, 2005; ZHAO et al., 2008), especially because it does not have a cure to date. Drugs known as dopaminergic medications and therapy are currently used to treat PD symptoms, being the *Levodopa* (L-dopa) the most widely used for such purpose. Another treatment that has been widely employed is the Deep Brain Stimulation, which is a surgical procedure that delivers electrical pulses to brain cells in order to reduce the effects of the symptoms.

The science does not measure efforts in order to make the quality of life of PD patients better. In computer science, for instance, techniques such as image processing, neural networks, and others have been widely applied in the pursuit of better results in both treatment and diagnosis. Spadotto et al. (SPADOTTO et al., 2010), for instance, introduced the Optimum-Path Forest (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012) classifier to aid the automatic identification of Parkinson's disease. Later on, the same group proposed an evolutionary-based approach to select the most discriminative set of features that helped to improve PD recognition rates (SPADOTTO et al., 2011).

Most works that address automatic PD recognition deal with voice-based data. Procedures to identify voiced and unvoiced (silent) periods have been actively pursued to analyze continuous speech samples since most techniques that quantify periodicity and regularity in voice signals are applied in the voiced regions only (SHAHBAKHI; FAR; TAHAMI, 2014). Das (DAS, 2010) presented a comparison of multiple classification methods for the diagnosis of PD, such as neural networks, regression and decision trees. Several evaluation methods were employed to calculate the performance of the classifiers, being the experiments conducted in a dataset composed of biomedical voice measurements from 31 people, in which 23 were diagnosed with Parkinson's disease. The best results were achieved by neural networks (around 92.9% of PD recognition rate).

Pereira et al. (PEREIRA et al., 2015, 2016) proposed to extract features from handwritten exams using visual features, which are learned from some drawings the patients were asked to perform, being the data used in the work made available in a dataset called "HandPD"[1]. Later on, Pereira et al. (PEREIRA et al., 2016) drove its approach to a deep learning application using the signals (time series) captured by the biometric pen *BiSP*® (BASHIR, 2012), which were further converted to the image domain with different resolutions and used as input to a Convolutional

[1]`http://wwwp.fc.unesp.br/~papa/pub/datasets/Handpd/`

Neural Network.

Another interesting methodology to learn discriminative features from data is related to the well-known Bag-of-words (BoW), though being quite difficult to establish the size of the bag (dictionary), as well as another open problem is how to choose the words that will compose that bag. Some years ago, Afonso et al. (AFONSO et al., 2012) proposed to use the unsupervised OPF (ROCHA; CAPPABIANCO; FALCÃO, 2009) to learn proper dictionaries since it does not require the number of words beforehand, thus becoming a useful tool for BoW purposes. Later on, Afonso et al. (AFONSO et al., 2016) presented a deep-hierarchical OPF (dOPF) clustering algorithm to make it way more efficient and validated it in the context of seismic-geological data classification.

Although BoW usage is not new in the context of time series for biomedical purposes (WANG et al., 2013), to best of our knowledge, it has not been applied for the identification of Parkinson's disease along with graph-based clustering algorithms so far, which turns out to be the main contribution of this work. Another main contribution is to use dOPF to learn dictionaries in a hierarchical way, where different layers of knowledge are used to compose the final dictionary. In short, the main idea of this work is to employ dOPF in the context of BoW applied for Parkinson's disease detection using the time series data from the HandPD dataset.

The remainder of this work is organized as follows: Our proposed approach is detailed in Section 10.2. The experimental setup, dataset and results are presented in Section 10.3. Finally, Section 10.4 states conclusions and future works.

10.2 Proposed Approach

This section describes all steps performed in the work to evaluate dOPF and BoW in the context of Parkinson's disease identification, as depicted in Figure 10.1.

Data acquisition Individuals were submitted to a series of tasks, in which they were asked to perform some hand movements and drawings using a biometric pen that contains six sensors in charge of recording hand movements (Figure 10.2) (BASHIR, 2012). The movements are represented by six different channels: microphone, finger grip, axial pressure of ink refill, and tilt and acceleration in the x, y and z directions.

Figure 10.3 depicts an example of an exam containing six tasks that evaluate the hand movements and help to detect any anomalies. In the first task (exam (a) in Figure 10.3), the individual is asked to draw a circle 12 times in the same place without stopping the movement between each circle. In the second task (exam (b) in Figure 10.3), the individual performs the same movement as in exam (a), but with its hand in the air. The third (exam (c) in Figure 10.3) and fourth (exam (d) in Figure 10.3) tasks concern drawing the spirals and meanders, respectively, over a guideline only once from the inner to the outer part. The last two tasks, i.e., exam (e) and exam (f) in Figure 10.3, stand for the diadochokinesis test, which is basically composed of hand-wrist movements performed with both hands. Each exam results in six different datasets, one for each task, and each sample from the dataset corresponds to an array of responses captured by each sensor in the interval of 1 ms.

Local descriptor extraction Given the recorded signals, the local descriptors are computed through a sliding window that goes along each of the six signals and computes a single-level Discrete Wavelet Transform (DWT) in each segment.

In fact, since there are six different signals, we work with six sliding windows, in which the segments of time within each of them have always the same initial and final times as they shift along the signals. The size of the sliding window and shifting are both user-defined. The DWT is applied to each segment of time separately, and the results in each segment are concatenated in order to form the final local descriptor[2].

Dictionary formulation The dictionary formulation aims to find the most representative "words" (descriptors) among a set of descriptors from the "bag" that are used in a later step for computing of a new sample representation. This step is usually performed by a clustering algorithm, in which the number of clusters defines the size of the dictionary, and each centroid becomes a "word" of the dictionary. It is usual to play with the size of the dictionary in order to find some trade-off between the computational cost and accuracy rate.

The new representation A signal can be represented by a set of descriptors, which can range from dozens to thousands. Some of these descriptors may be similar or only represent noisy information. Thus, in order to obtain a compressed and meaningful representation of the signal, the descriptors were quantized based on the dictionary computed previously. The quantization step will provide a histogram for each sample with length equals to the size of the dictionary, in which each bin will have the frequency of its closest word in the input signal. Then, the final histogram is further used as an input for machine learning algorithms.

[2]We used sliding windows of size 100 ms with a stride of 50 ms, being such values empirically chosen.

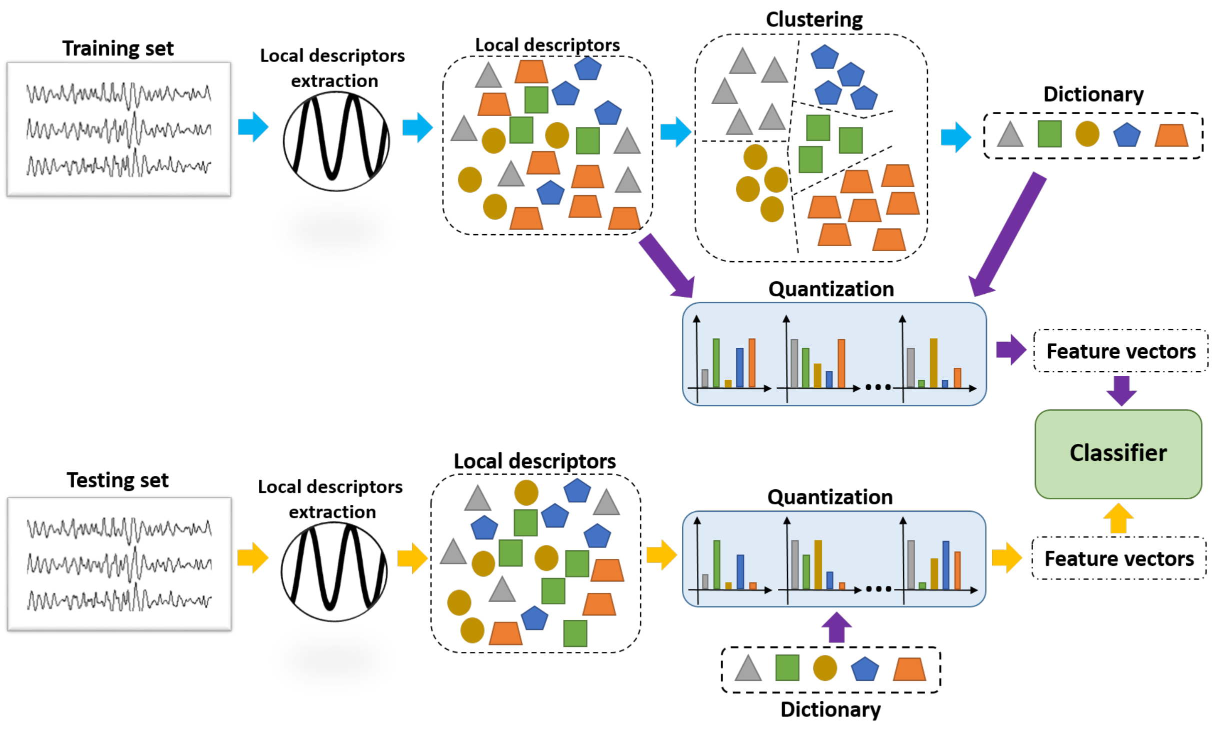


Figure 10.1: Proposed approach based on BoW and dOPF for computer-aided PD diagnosis. The main workflow is indicated by the light blue arrows: local descriptors are extracted and clustered in order to build the dictionary. The dictionary is used for the quantization of both training and testing signals that is the process of computing the feature vectors (flow indicated by purple arrows). Similarly to the training phase, testing signals have their descriptors computed and the signals are quantized (flow indicated by yellow arrows). Finally, a classifier is fed by the resulting training and testing feature vectors. Notice the two depicted dictionaries are the same.

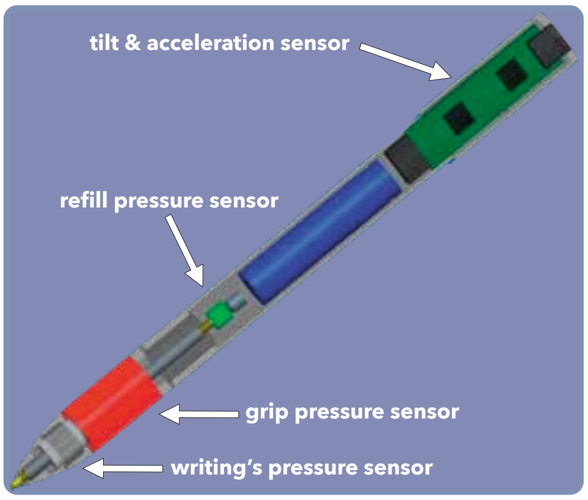
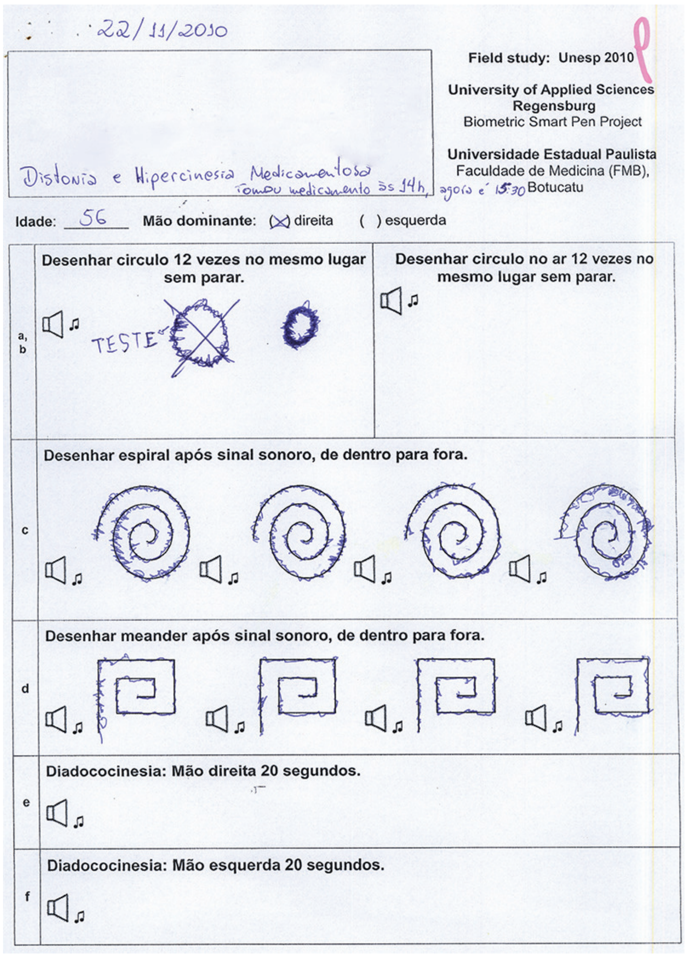


Figure 10.2: Biometric pen. Extracted from (PEREIRA et al., 2016).

10.3 Experiments and Results

The experimental setup used all data recorded from a total of 66 exams, being 35 control individuals and 31 patients. The output of the protocol discussed in the previous section results


22/11/2010

Field study: Unesp 2010

University of Applied Sciences Regensburg
Biometric Smart Pen Project

Universidade Estadual Paulista
Faculdade de Medicina (FMB), Botucatu

Distonia e Hipercinesia Medicamentosa
Tomou medicamento às 14h, agora é 15:30

Idade: 56 Mão dominante: [✓ ( ) direita] ( ) esquerda

a, b	Desenhar circulo 12 vezes no mesmo lugar sem parar. TESTE	Desenhar circulo no ar 12 vezes no mesmo lugar sem parar.
c	Desenhar espiral após sinal sonoro, de dentro para fora.	
d	Desenhar meander após sinal sonoro, de dentro para fora.	
e	Diadococinesia: Mão direita 20 segundos.	
f	Diadococinesia: Mão esquerda 20 segundos.	

Figure 10.3: Form used to assess the handwritten skills. Extracted from (PEREIRA et al., 2016).

in six different datasets, one for each task. The dictionary learning step was performed by means of three different techniques: dOPF, k-means[3] and OPF[4]. The main idea is to evaluate the quality of clustering of each technique through the accuracy rate obtained in the classification phase. The architecture used by dOPF is composed of four layers, in which the values of k_{max} are: 100 for the first layer, 1% of the number of clusters computed in the previous layer are used as an input for the second layer, and 10% of the number of clusters computed in their respective predecessor layers for the third and fourth layers. The value of k for k-means is always set as the number of clusters found by the fourth (last) layer of dOPF approach. Regarding the OPF algorithm, the values for k_{max} were empirically set as 2,500 for the Spiral and Meander datasets, and as 1,500 for the remaining datasets. The idea in using the same number of clusters for dOPF and k-means is to allow a fair comparison between them.

Table 10.1 presents the number of descriptors extracted from the training set of each dataset, as well as the number of words computed in each case. In the column regarding dOPF, it is

[3]Our own implementation.
[4]https://github.com/LibOPF/LibOPF

Table 10.1: Number of descriptors extracted from the training set and number of words computed by each technique.

dataset (task)	# descriptors	Deep-OPF	k-means	OPF
Circ-A exam (a)	18,000	5,682 - 2,584 - 228 - **68**	68	693
Circ-B exam (b)	11,898	538 - 376 - 43 - **17**	17	33
Spiral exam (c)	46,637	12,118 - 3,951 - 370 - **92**	92	1,424
Meander exam (d)	41,094	10,865 - 3,937 - 429 - **99**	99	1,591
Dia-A exam (e)	14,608	666 - 480 - 95 - **47**	47	80
Dia-B exam (f)	13,947	657 - 394 - 78 - **27**	27	70

shown the number of words found for each of the four layers, but only the ones computed in the last layer (bolded) are used for the quantization of both training and testing sets.

The experiments were performed using the hold-out procedure with 15 runs. Both training and testing sets were partitioned using 50% of the entire dataset each, being randomly generated in each new run. In this step, there were employed three different classifiers for comparison purposes: Naïve Bayes Classifier[5], supervised OPF (sOPF)[6] and SVM using a Radial Basis Function kernel with parameter optimization (SVM-RBF) (PEDREGOSA et al., 2011) .

Tables 10.2a— 10.2f present the mean recognition rates concerning all six exams, being the accuracy computed according to Papa et al. (PAPA; FALCÃO; SUZUKI, 2009). The best results are defined according to the Wilcoxon signed-rank (WILCOXON, 1945) with a significance of 0.05, which pointed out the best ones in bold for each exam. Further, we also considered the best among all the exams as the underlined ones.

Let us first analyze the best results among all. The statistical evaluation pointed out [OPF, SVM-RBF] and [k-means, BC] as the best pairs of [dictionary learner, classifier] with accuracies near to 81% and 83%, respectively. Comparing that recognition rates against some previous works, the proposed approach showed significant gains (from 10% to 30%) against the one presented by Pereira et al. (PEREIRA et al., 2016). Despite that our results were slightly below those achieved by a further work of the same authors that makes use of deep learning techniques (PEREIRA et al., 2016), our approach is way more efficient than using deep learning techniques taking into account a few architectures.

With respect to the best accuracies concerning each exam, dOPF obtained very much suitable results, being more accurate than naïve OPF in most cases. Supervised OPF obtained good results as well, but SVM-RBF achieved the best recognition rates in a few more situations. Additionally, we also evaluated the accuracy per class for all situations, as presented in Tables

[5]Our own implementation.

[6]https://github.com/LibOPF/LibOPF

Table 10.2: Overall accuracies.

(a) Circ-A dataset.

	BC	**sOPF**	**SVM-RBF**
dOPF	**82.96**±**2.88**	**81.71**±**5.12**	73.87±4.58
k-means	**83.38**±**4.22**	**82.01**±**5.11**	65.80±12.39
OPF	**81.06**±**4.36**	**81.90**±**4.89**	76.17±6.92

(b) Circ-B dataset.

	BC	**sOPF**	**SVM-RBF**
dOPF	68.75±7.96	69.14±6.95	**77.31**±**4.45**
k-means	67.80±7.44	65.58±6.79	**74.54**±**6.39**
OPF	70.81±4.62	**73.08**±**8.96**	**76.69**±**5.38**

(c) Spiral dataset.

	BC	**sOPF**	**SVM-RBF**
dOPF	**78.30**±**5.80**	76.73±6.83	77.25±3.46
k-means	73.37±5.37	73.11±5.31	78.83±2.20
OPF	75.40±3.09	75.57±3.13	**81.03**±**2.40**

(d) Meander dataset.

	BC	**sOPF**	**SVM-RBF**
dOPF	73.33±4.97	74.07±2.90	**80.45**±**2.42**
k-means	76.07±3.31	76.09±2.77	78.26±3.91
OPF	78.53±3.15	77.21±3.52	**81.07**±**2.60**

(e) Dia-A dataset.

	BC	**sOPF**	**SVM-RBF**
dOPF	**69.86**±**7.21**	**70.93**±**7.29**	**68.69**±**7.26**
k-means	**72.18**±**7.46**	**72.43**±**5.81**	**73.93**±**8.66**
OPF	**70.72**±**6.60**	**67.01**±**7.45**	**68.69**±**7.26**

(f) Dia-B dataset.

	BC	**sOPF**	**SVM-RBF**
dOPF	**67.96**±**8.10**	64.86±7.93	61.89±8.49
k-means	**72.92**±**8.51**	**69.84**±**9.03**	**67.24**±**9.31**
OPF	63.77±8.85	67.25±6.80	**66.30**±**7.38**

Table 10.3: Average accuracy rate for each class.

(a) Circ-A dataset.

	BC			sOPF			SVM-RBF		
	dOPF	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**
Patient	**83.33±5.62**	**84.17±7.79**	**79.17±12.19**	**77.5±10.24**	**79.58±10.15**	**80.83±8.34**	61.67±15.10	70.42±12.60	75.42±6.87
Control	**82.59±8.09**	**82.59±8.09**	**82.96±8.52**	**85.93±6.59**	**84.44±12.09**	**82.96±6.79**	**67.41±11.67**	**67.04±7.99**	**71.48±9.82**

(b) Circ-B dataset.

	BC			sOPF			SVM-RBF		
	dOPF	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**
Patient	**60.83±18.32**	**61.25±11.68**	**74.58±13.39**	**63.75±13.35**	**63.75±13.35**	**54.17±13.04**	**64.58±13.24**	**68.75±14.43**	**57.92±10.07**
Control	76.67±49.12	77.04±12.48	59.99±9.56	71.85±13.98	67.41±12.32	57.04±17.15	**77.04±10.51**	**77.41±6.24**	45.93±7.98

(c) Spiral dataset.

	BC			sOPF			SVM-RBF		
	dOPF	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**
Patient	**74.51±10.59**	65.49±10.63	**72.90±6.90**	**78.04±12.07**	67.06±10.14	**71.61±6.27**	67.81±2.17	**73.59±3.67**	**74.58±0.82**
Control	82.08±8.14	81.25±11.33	77.90±6.95	75.42±11.92	79.17±10.48	79.52±6.20	**89.43±1.83**	84.85±2.25	86.43±1.09

(d) Meander dataset.

	BC			sOPF			SVM-RBF		
	dOPF	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**
Patient	**76.61±4.04**	69.46±5.96	**76.77±8.44**	**75.38±4.62**	73.23±4.51	**77.85±3.93**	74.81±2.18	71.06±2.62	74.54±1.37
Control	73.33±4.97	82.67±4.88	80.29±4.68	72.76±5.47	78.95±4.99	76.57±5.47	**85.80±0.89**	**84.43±3.76**	87.99±0.72

(e) Dia-A dataset.

	BC			sOPF			SVM-RBF		
	dOPF	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**
Patient	**67.50±16.01**	**66.67±14.01**	**51.67±17.15**	**65.83±13.28**	**70.42±12.59**	**52.08±11.10**	**66.25±13.46**	**66.25±15.61**	**47.50±12.87**
Control	**72.22±9.51**	**75.19±5.69**	**50.74±9.70**	**78.52±8.33**	**74.44±6.97**	**55.56±12.67**	**75.19±7.55**	**67.78±12.37**	**50.00±13.46**

(f) Dia-B dataset.

	BC			sOPF			SVM-RBF		
	dOPF	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**	**dOPF**	***k*-means**	**OPF**
Patient	**63.33±12.88**	60.83±11.29	60.00±32.18	**72.50±10.89**	**67.08±13.39**	47.92±10.62	**71.25±11.81**	**73.75±11.23**	**52.08±15.92**
Control	**72.59±9.62**	**68.89±9.89**	50.37±28.07	**73.33±9.56**	**72.59±10.63**	48.52±12.42	**56.29±15.56**	60.74±11.38	**53.70±9.44**

10.3a— 10.3f, whose best results are also highlighted considering the Wilcoxon signed-rank. The best results for each class are in bold, and the best among all datasets is underlined. Actually, the main improvement concerns the accuracy for the identification of healthy individuals, since Pereira et al. (PEREIRA et al., 2016) obtained recognition rates nearly 50% over the Meander and Spirals datasets for the control class. The proposed approach increased not only the global accuracy with respect to the work by Pereira et al. (PEREIRA et al., 2016), but also the specificity and sensitivity for most of the cases. Also, Circ-A dataset provided two out of the five best results, thus showing as a good alternative for Parkinson's Disease identification.

Table 10.4 presents the mean computational load required by each technique to learn the dictionary. Notice the computational burden for dOPF considers the four layers. In this context,

k-means figured as the fastest one due to its simplicity. If one considers dOPF and OPF only, we can observe the former is about 78 times faster in Circ-B dataset, which is quite effective. The lowest gains can be observed in both Meander and Spiral datasets. The small differences come from the fact the value used for k_{max} in both situations is small, thus justifying the fact the dictionaries computed in these datasets have very high dimension when compared to others.

Table 10.4: Dictionary learning computational load [s] required by each technique.

dataset (task)	**dOPF**	***k*-means**	**OPF**
Circ-A (a)	968.167	37.008	49,087.137
Circ-B (b)	419.498	13.113	32,777.539
Spiral (c)	6,063.205	239.859	6,643.906
Meander (d)	5,003.233	208.443	5,168.819
Dia-A (e)	613.109	19.878	41,189.133
Dia-B (f)	569.053	11.025	39,367.844

10.4 Conclusions

This work introduced a deep-hierarchical version of the unsupervised OPF algorithm for dictionary learning in the context of computer-aided Parkinson's disease identification. The experiments were performed using data from handwriting dynamics, similarly to the work by Pereira et al. (PEREIRA et al., 2016), but now handled as signals and not images.

The application of the BoW paradigm can extract more information by computing local descriptors that can enhance the overall accuracy. Also, dOPF showed satisfactory results in its first application for BoW-based Parkinson's Disease identification. Experiments over six datasets considered dOPF against the well-known k-means and naïve OPF clustering for dictionary learning. Further, supervised techniques were used for classification purposes.

Future works will consider learning hierarchical BoWs, i.e., one bag for each layer in the dOPF formulation. We believe each layer can carry different information about the problem.

Chapter 11

HIERARCHICAL LEARNING USING DEEP OPTIMUM-PATH FOREST

This chapter presents the extension of the work introduced in Chapter 10 by proposing a hierarchical-based learning approach to design visual dictionaries through the Deep Optimum-Path Forest classifier, in which the dictionary is comprised of data from all layers instead of the last one only. This work was published in the Special Issue on Feature representations for Medical Images and Activity Understanding of the Journal of Visual Communication and Image Representation (AFONSO et al., 2020) (Qualis-CC A1) as a guest paper.

11.1 Introduction

Image and signal classification have been widely researched in the past decades, with a considerable effort towards deep learning (DL) techniques. Despite the fact that DL-driven approaches are known to be quite useful in generalizing over a number of problems, they still can not deal with some simple problems as well (NYE; SAXE, 2018). Also, specific neural architectures need to be designed to cope with signal classification problems since most of the models available in the literature are developed to handle image-based applications only.

Apart from being proposed many years ago (CSURKA et al., 2004), the well-known Bag-of-Visual-Words (BoVW) paradigm has been consistently employed and enhanced over the years to address both image- and signal-based classification problems. In a nutshell, the idea consists in extracting information (e.g., visual words/key points) from the data for further using them to compose a dictionary (i.e., bag) that can be employed as a final descriptor. Applications in medical data vary from X-ray categorization to histopathology image classification (AVNI et al., 2011; CAICEDO; CRUZ; GONZALEZ, 2009; SOUZA et al., 2017), among others.

Computer-assisted Parkinson's disease (PD) identification is another research area that can benefit from automated diagnosis and the BoVW paradigm. Such illness is known to be neurodegenerative, it has no cure, and its main symptoms include the freezing of gate, tremors, and speech alterations, to name a few. In this context, a considerable number of works that deal with automated PD diagnosis can be referred in the literature. Spadotto et al. (SPADOTTO et al., 2010), for instance, introduced the Optimum-Path Forest (OPF) (PAPA; FALCÃO; SUZUKI, 2009; PAPA et al., 2012; PAPA; FERNANDES; FALCÃO, 2017) for PD identification from speech signals, and later on the same group of authors employed evolutionary optimization techniques to select the most relevant features to deal with the same problem (SPADOTTO et al., 2011). Sama et al. (SAMà et al., 2017) and Bächlin et al. (BäCHLIN et al., 2010) explored wearable accelerometers to detect the freezing of gate and to provide assistance as soon as the condition is detected. Rigas et al. (RIGAS et al., 2012) investigated an automated method that estimates the type and severity of tremors based on data acquired from accelerometers attached to specific positions at a patient's body. The estimations are used to assess both resting and action tremors.

Other works used images to cope with PD recognition automatically. Pereira et al. (PEREIRA et al., 2016) proposed to extract features from handwriting exams that were further digitized to fulfill the aims of the work. They used the HfromandPD dataset[1], which comprises exams performed by healthy individuals and PD patients to detect subtle tremors when drawing spirals and meanders on a piece of paper. Since the exams were conducted using a pen equipped with sensors [2], the same group of authors further proposed to use the signals obtained from the pen as a means to perform automatic PD recognition (PEREIRA et al., 2016). Afonso et al. (AFONSO et al., 2019a) introduced the concept of "deep recurrence plots" for the identification of Parkinson's disease, where the idea is to employ recurrence plots (ECKMANN; KAMPHORST; RUELLE, 1997) to model the time dependency of the signals acquired during the exam.

Afonso et al. (AFONSO et al., 2017) also proposed a BoVW model to learn information from the signals used in the works mentioned earlier to be further used to cope with the problem of Parkinson's disease identification. The proposed approach first extracts key points from the signal, which are then used to compose the final dictionary. Further, the key points are clustered using the unsupervised OPF technique (ROCHA; CAPPABIANCO; FALCÃO, 2009) to select only the most informative ones that will compose the dictionary. The results showed that OPF could build more informative dictionaries than other clustering algorithms.

In this paper, we extend the work of Afonso et al. (AFONSO et al., 2017) by proposing a hierarchical-based learning methodology to design visual dictionaries. The proposed approach

[1]`http://wwwp.fc.unesp.br/~papa/pub/datasets/Handpd`

[2]`https://www.oth-regensburg.de/index.php?id=5312/biometrics.html`

makes use of the Deep OPF classifier (AFONSO et al., 2016), which aims at performing different levels of clustering to learn and encode distinct information at each phase. We showed results that outperformed the ones obtained by Afonso et al. (AFONSO et al., 2017) in the context of computer-assisted Parkinson's disease identification using signals derived from handwriting exams.

The remainder of this paper is organized as follows. Sections 11.2 and 11.3 describe how deep representations are learned through Deep OPF and the proposed approach, respectively. Section 11.4 presents the experiments, and Section 11.5 states conclusions and future works.

11.2 Deep-based Representations through Optimum-Path Forest

Deep-based representations are commonly employed in image classification applications, but they are not restricted to such ones. Such representations are obtained through deep learning architectures that are characterized by a model comprised of many layers. The introduction of such model allows learning numerous features from data as it flows through the layers. One of the most common models is the Convolutional Neural Network, which applies a series of convolutional kernels to the data, being each of them responsible for learning different information. The dOPF follows the same idea by learning multiple representations, being each of them the outcome of a clustering process from a different layer.

In the context of Bag-of-Visual Words using dOPF, the final bag could be a coarser model if only the outcome of the last layer was used to compose it (i.e., only the prototypes of the last layer comprise the bag), as proposed by Afonso et al. (AFONSO et al., 2016). However, an enriched model could be accomplished by adding information computed by the intermediate layers as well. As a comparison, the idea of using intermediate representations would be similar to using the features learned by the many hidden layers of a deep-learning model. Each layer can learn more complex features and, therefore, more robust representations.

As mentioned earlier, we propose to extend the work of Afonso et al. (AFONSO et al., 2017) by employing hierarchical learning in the context of BoVW, hereinafter called hOPF (hierarchical OPF) . The proposed approach will provide a more complex and more robust dictionary, being such representation the collection of selected visual words computed by all layers. Figure 11.1 illustrates both dictionary learning methods, i.e., dOPF and hOPF.

Although dOPF provides a simpler and coarser representation by using only the features learned in the last layer, hOPF outputs a more complex and robust representation that stands

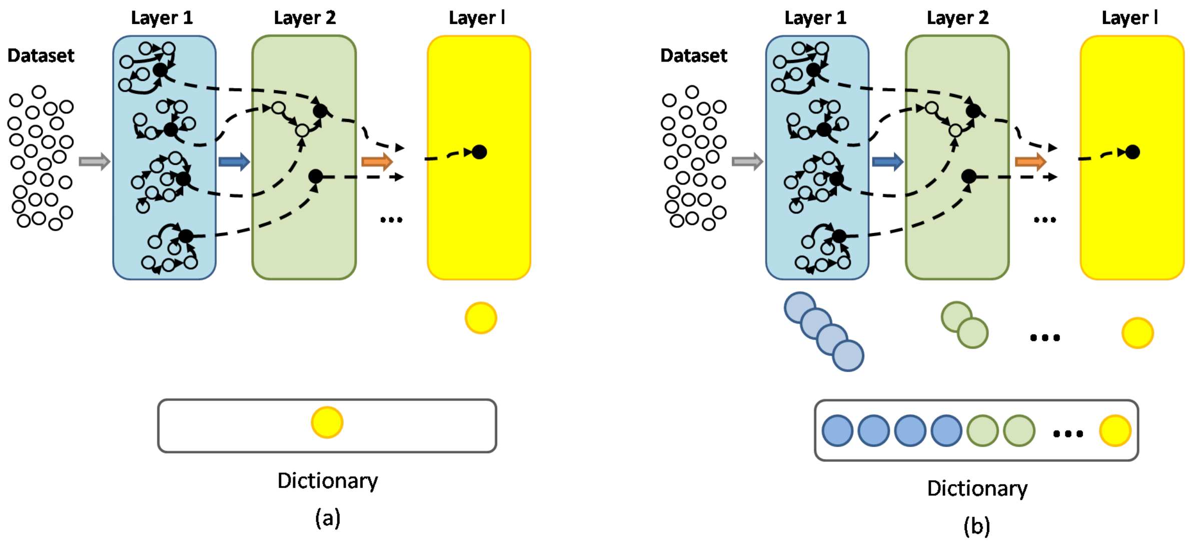


Figure 11.1: The main difference between (a) dOPF and (b) hOPF concerns the usage (or not) of features learned in the intermediate layers.

for the concatenation of features learned by all layers. In the context of BoVW, the resulting dictionary generated by hOPF will be of size $|\mathscr{P}_1| + |\mathscr{P}_2| + \ldots + |\mathscr{P}_l|$.

11.3 Proposed Approach

This section describes the steps employed in the assessment of dOPF and hOPF as visual dictionary learning methods for BoVW in the context of automatic Parkinson's disease identification, as illustrated in Figure 11.2. The workflow indicated by the light blue arrow concerns the training phase. The first step computes the local descriptors from the training signals to further clustering. The most representative samples from each cluster compose the dictionary, which is used for quantization (i.e., flow indicated by the purple arrow) of both training and testing signals. The outcome of such process is the new representation of each sample. Similarly, testing signals have their local descriptors extracted and quantized (i.e., flow indicated by the yellow arrows). The final step is to perform training and classification using the new computed representations. Adapted from Afonso et al. (AFONSO et al., 2017)

11.3.1 Data acquisition

The experimental data were collected from a series of tasks performed by individuals using a smart pen. The tasks exercise different hand movements that enable to capture the handwriting dynamics for further analysis. Furthermore, the exercises were elaborated in such way that are supposed not to be trivial to PD patients. All hand motion is captured by the smart pen that

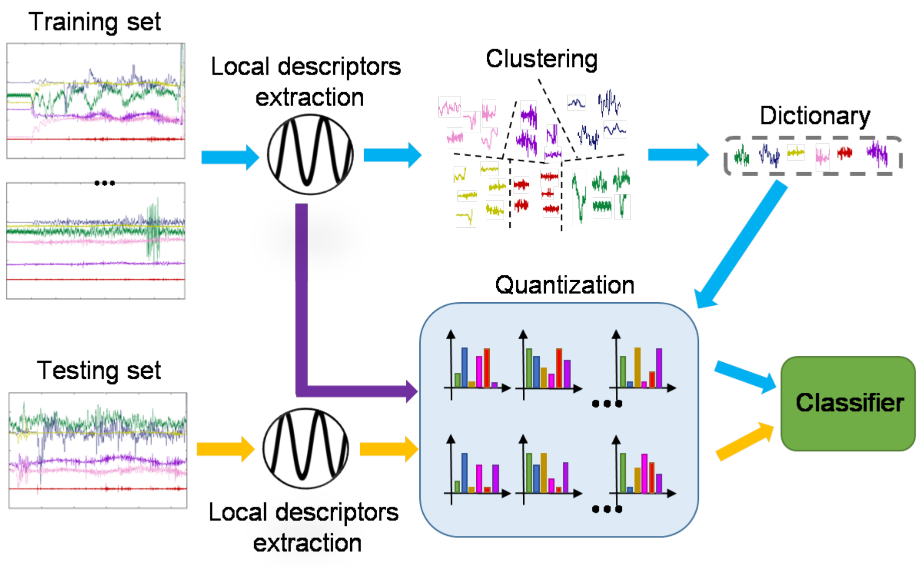


Figure 11.2: Proposed approach based on BoW and dOPF for computer-aided PD diagnosis.

contains sensors that provide information on finger grip, the axial pressure of ink refill, tilt and acceleration in the x, y, and z directions.

Figure 11.3 illustrates the set of six tasks employed to evaluate the hand movements and to support the detection of anomalies. In the first task (exam (a) in Figure 11.3), the individual is asked to draw a circle 12 times continuously. In the second task, the individual performs the circle-drawing movement (i.e., on the air) 12 times continuously (exam (b) in Figure 11.3). The third and fourth tasks also concern drawing activities. In the exam (c) in Figure 11.3, four spirals are drawn over a guideline from the inner to the outer part. The exam (d) in Figure 11.3 comprises the drawing of meander also four times and from the inner to the outer part. Last but not least, the fifth and sixth tasks are known as the diadochokinesis test and are used to evaluate the wrist movement of the right and left hands, as displayed in the exam (e) and exam (f)[3] in Figure 11.3, respectively.

11.3.2 Local descriptor extraction

The local descriptors are extracted from the recorded signals in a sliding-window fashion that goes through each of the six signals. The descriptors are computed using a single-level Discrete Wavelet Transform (DWT) applied to each segment delimited by the sliding window. Each time segment of the signal is in fact represented by the concatenation of the resulting DWT from six sliding windows (i.e., one sliding window applied to each signal), as depicted in

[3]Adapted from `http://physiologie.cc/XV.5.htm`

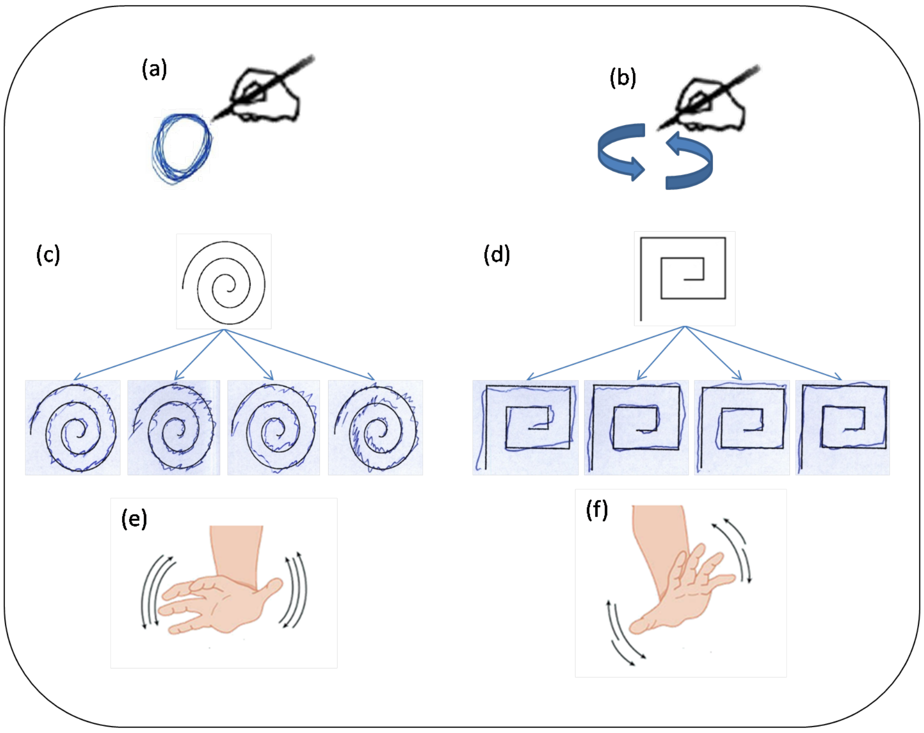


Figure 11.3: Tasks perfomed to the assessment of hand movements

Figure 11.4. Notice that all sliding windows comprise the same portion of time (i.e., the same initial and final times) as they go through the signals and the DWT is computed independently to each window. Moreover, the window length and shifting are user-defined. The experiments used windows of 100 ms of length and a stride of 50 ms, which were empirically chosen.

11.3.3 Dictionary formulation

The dictionary is formulated by selecting the most representative "words" (descriptors) among the set computed in the previous step, and it is further used to compute a new sample representation. The most representative words are usually selected by a clustering algorithm where each centroid becomes a "word" of the dictionary. Therefore, the dictionary size is defined by the number of clusters. Since it has some impact on the accuracy rate, it is common the use of different sizes for the dictionary to balance the computational cost and accuracy rate. The prototypes are very suitable to represent the samples of their trees (i.e., prototypes are equivalent to centroids of a cluster), thus being good representations to compose the dictionaries.

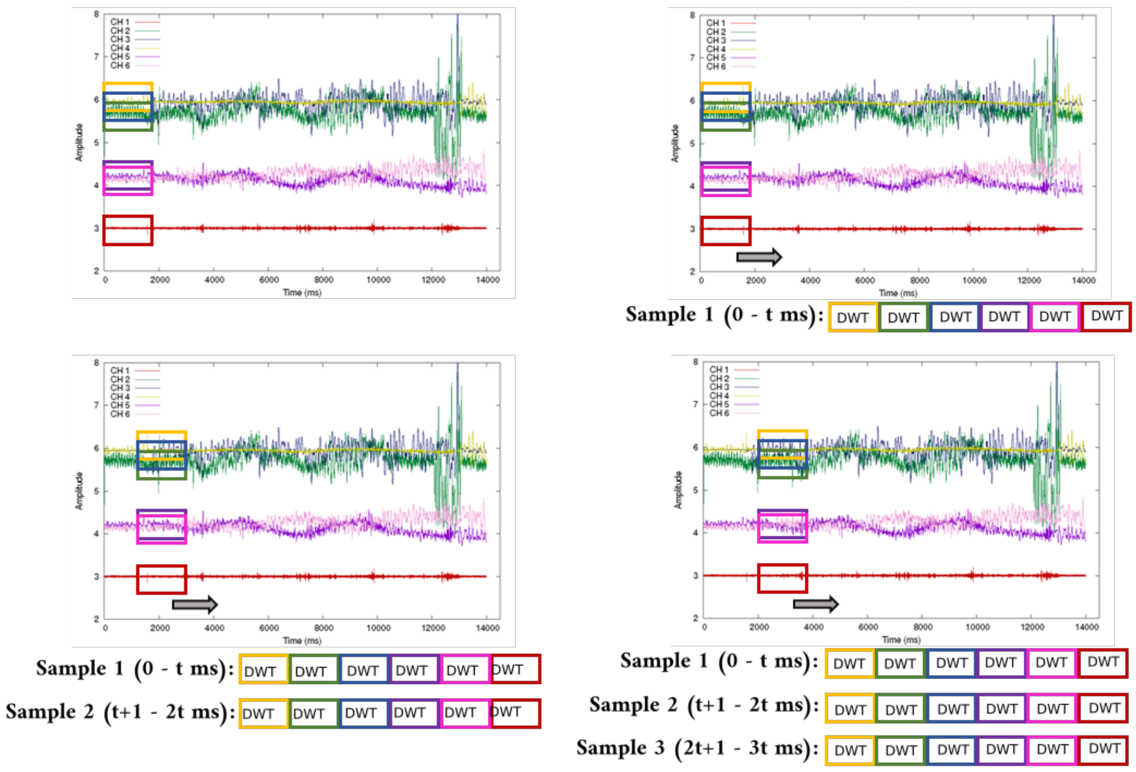

CH 1
CH 2
CH 3
CH 4
CH 5
CH 6
Amplitude
Time (ms)
Sample 1 (0 - t ms): DWT DWT DWT DWT DWT DWT
Sample 2 (t+1 - 2t ms): DWT DWT DWT DWT DWT DWT
Sample 3 (2t+1 - 3t ms): DWT DWT DWT DWT DWT DWT

Figure 11.4: Local feature extraction.

11.3.4 The new representation

A signal can be represented by its set of descriptors, which can range from dozens to thousands. However, a few of these descriptors might be variations of another one or only represent noisy information. Moreover, machine learning techniques cannot be directly applied to the sets of descriptors since their dimension is not the same to all signals. Therefore, quantization is performed so that signals can be mapped into the same feature space. The outcome of the process is a histogram of length equals to the size of the dictionary, where each bin stores the frequency of its closest word in the input signal. Finally, machine learning techniques can be applied using the histograms as input.

11.4 Experiments and results

In this section, we provide details concerning the experiments carried out on the assessment of deep-based dictionaries in the context of automatic Parkinson's disease identification. The experiments were divided into two parts: (i) the former one evaluates and compares dOPF-based dictionaries against the traditional OPF-based bags and the traditional BoVW method that computes the bags using the well-known *k*-means; and (ii) the second part provides a comparison of the proposed approach (i.e., hOPF) with the method presented in the work of Afonso et al. (AFONSO et al., 2016). Additionally, the second section of experiments includes the hOPF performance evaluation using compressed versions of the representations learned. For that purpose, we applied the Restricted Boltzmann Machine (RBM) (HINTON, 2002) to provide different compression levels. Notice that both experiments used data collected from 66 exams (35 healthy individuals and 31 PD patients), and the output of the protocol discussed in the previous section results in six different datasets, one for each task. The following sections describe the particularities of each experiment and the results using the proposed methodology.

11.4.1 Single-scale deep-based representations

This experiment aims at evaluating the clustering quality of dOPF, k-means[4] and OPF[5] through the accuracy rate obtained in the classification phase. The dOPF used in the work comprises an architecture with four layers, being the values of k_{max} set as follows: 100 for the first layer, 1% of the number of clusters computed in the previous layer are used as an input for the second layer, and 10% of the number of clusters computed in their respective antecessor layers for the third and fourth layers[6]. The parameter k for k-means is always set as the number of clusters found by the fourth (last) layer of dOPF approach to allow a fair comparison. Regarding the OPF algorithm, the values for k_{max} were empirically set as $2,500$ for the Spiral and Meander datasets, and as $1,500$ for the remaining datasets.

Table 11.1 presents the number of local descriptors obtained from each dataset, as well as the number of visual words selected by each clustering technique. Notice the values concerning the dOPF column stand for the number of visual words selected by each layer. As aforementioned, dOPF dictionaries are comprised of the visual words computed by the last layer only (i.e., bolded values).

[4]Our implementation.

[5]https://github.com/jppbsi/LibOPF

[6]Those values were empirically set.

Table 11.1: Number of descriptors extracted from the training set and number of words computed by each technique.

dataset (task)	# descriptors	dOPF	*k*-means	OPF
Circ-A exam (a)	18,000	5,682 - 2,584 - 228 - **68**	68	693
Circ-B exam (b)	11,898	538 - 376 - 43 - **17**	17	33
Spiral exam (c)	46,637	12,118 - 3,951 - 370 - **92**	92	1,424
Meander exam (d)	41,094	10,865 - 3,937 - 429 - **99**	99	1,591
Dia-A exam (e)	14,608	666 - 480 - 95 - **47**	47	80
Dia-B exam (f)	13,947	657 - 394 - 78 - **27**	27	70

The clustering quality was assessed under a hold-out procedure with 15 runs, being the training and testing sets randomly partitioned in each new run and always with 50% of the dataset each. For the sake of classification purposes, we also performed a comparison between Naïve Bayes Classifier (BC)[7], supervised OPF (sOPF)[8] and SVM using a Radial Basis Function (RBF) kernel with fine-tunned parameters (SVM-RBF) (PEDREGOSA et al., 2011).

Tables 11.3— 11.8 present the mean recognition rates concerning all six exams, being the accuracy computed according to Papa et al. (PAPA; FALCÃO; SUZUKI, 2009), which considers unbalanced datasets. The best results are defined according to the Wilcoxon signed-rank (WILCOXON, 1945) with a significance of 0.05, which pointed out the best ones in bold for each exam. Further, we also considered the best among all exams as the underlined ones.

The statistical evaluation pointed out [OPF, SVM-RBF] and [*k*-means, BC] as the best pairs of [dictionary learner, classifier] with accuracies near to 81% and 83%, respectively. Comparing that recognition rates against some previous works (PEREIRA et al., 2016), the proposed approach showed significant gains, ranging from 10% to 30%.

Concerning the best accuracies regarding each exam, dOPF obtained very much suitable results, being more accurate than naïve OPF in most cases. Supervised OPF obtained good results as well, but SVM-RBF achieved the best recognition rates in a few more situations. Additionally, we also evaluated the accuracy per class for all situations, as presented in Tables 11.9— 11.14, whose best results are also highlighted considering the Wilcoxon signed-rank. The best results for each class are in bold, and the best among all datasets is underlined. Actually, the main improvement concerns the accuracy for the identification of healthy individuals, since Pereira et al. (PEREIRA et al., 2016) obtained recognition rates nearly to 50% over the Me-

[7]Our implementation.
[8]https://github.com/LibOPF/LibOPF

ander and Spirals datasets for the control class. The proposed approach increased not only the global accuracy with respect to the work by Pereira et al. (PEREIRA et al., 2016), but also the specificity and sensitivity for most of the cases. Also, Circ-A dataset provided two out of the five best results, thus showing as a good alternative for the Parkinson's Disease identification.

Table 11.2 presents the mean computational load required by each technique for dictionary learning. Notice the computational burden for dOPF considers the four layers. In this context, k-means figured as the fastest one due to its simplicity. If one considers dOPF and OPF only, we can observe the former is about 78 times faster in Circ-B dataset, which is quite effective. The lowest gains can be observed in both Meander and Spiral datasets. The small differences come from the fact the value used for k_{max} in both situations is small, thus justifying the fact the dictionaries computed in these datasets have very high dimension when compared to others.

Table 11.2: Dictionary learning computational load [s] required by each technique.

dataset (task)	**dOPF**	***k*-means**	**OPF**
Circ-A (a)	968.167	37.008	49,087.137
Circ-B (b)	419.498	13.113	32,777.539
Spiral (c)	6,063.205	239.859	6,643.906
Meander (d)	5,003.233	208.443	5,168.819
Dia-A (e)	613.109	19.878	41,189.133
Dia-B (f)	569.053	11.025	39,367.844

11.4.2 Multi-scale deep-based representations

As aforementioned, this round of experiments aims at providing a performance comparison between dOPF and hOPF. To fulfill that purpose, the quality of the dictionaries provided by both techniques was compared using the protocol described in Section 11.4.1. As more visual words were added, hOPF dictionaries provide higher-dimensional representations. Hence, an additional experiment evaluates the quality of compressed representations computed by RBM. There were used representation sizes of 25% (hOPF-25), 50% (hOPF-50), and 75% (hOPF-75) of the original one (hOPF). Figure 11.5 illustrates the workflow of representation compression.

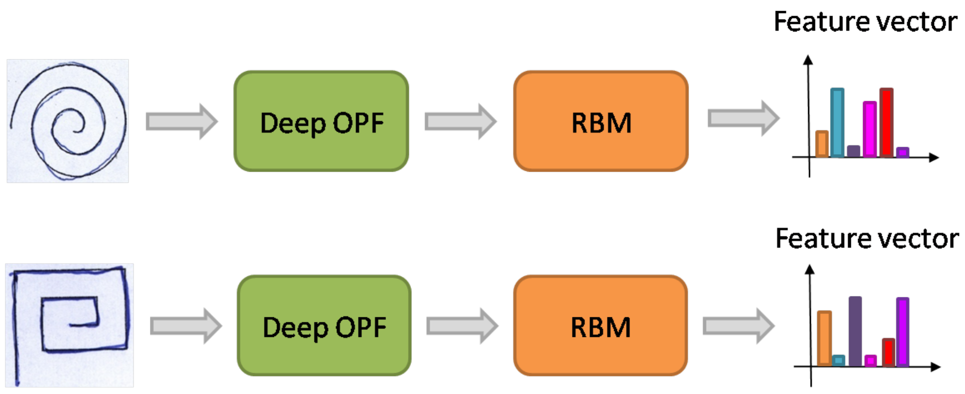


Figure 11.5: The Deep OPF block represents the workflow depicted in Figure 11.2. The outcome of such process is used as input of RBM that outputs a compressed representation used by classifiers.

Tables 11.3– 11.8 provide the overall accuracy rates concerning each dataset. The accuracy rate was computed using the same formulation as in Section 11.4.1, and the best results (i.e., bolded ones) were determined by the Wilcoxon signed-rank with significance as 0.05.

In general, hOPF-based dictionaries achieved competitive results in all six datasets and always figured among the best ones. Also, slight improvements compared to dOPF can be observed in most scenarios, being the most significant ones achieved in the Dia-B dataset (Table 11.8). The average gain in that dataset varies from 5.79% (BC) to 8.83% (SVM-RBF).

An interesting aspect to be highlighted, it is the fact that compressed representations computed by RBM also figured among the best results, even the most compressed ones (hOPF-25). The representations hOPF-50 and hOPF-75 achieved the best performance among the compressed versions with best results in 11 out of 18 scenarios against 6 out 18 of hOPF-25.

Concerning the classifiers employed in the work, it can be observed a similar situation as the one illustrated in Section 11.4.1. The classifiers obtained good results, but SVM-RBF outperformed them all in more situations. The highest accuracy among all datasets was achieved by the pair [hOPF, SVM-RBF] with 85.29%.

Table 11.3: Circ-A dataset - Overall accuracies.

	dOPF	**hOPF**	**hOPF-25**	**hOPF-50**	**hOPF-75**
BC	**82.94±6.00**	**82.94±5.69**	**79.80±6.75**	**81.96±7.53**	**80.20±6.14**
sOPF	**82.16±6.43**	**82.94±5.69**	**79.22±6.71**	**80.39±7.18**	**79.80±6.14**
SVM-RBF	**84.51±5.82**	**85.29±3.69**	**83.14±4.19**	**84.12±8.23**	**81.76±4.87**

Table 11.4: Circ-B dataset - Overall accuracies.

	dOPF	**hOPF**	**hOPF-25**	**hOPF-50**	**hOPF-75**
BC	**80.98±3.49**	**79.22±4.65**	69.80±5.50	68.63±8.73	68.04±8.67
sOPF	**80.00±3.88**	**79.02±5.87**	69.61±5.63	68.63±8.73	68.04±8.81
SVM-RBF	**79.61±6.33**	**79.02±4.57**	**78.24±6.07**	**78.04±5.76**	**79.41±7.94**

Table 11.5: Spiral dataset - Overall accuracies.

	dOPF	**hOPF**	**hOPF-25**	**hOPF-50**	**hOPF-75**
BC	**77.22±3.57**	**78.94±2.58**	74.04±3.64	72.88±2.35	72.32±3.36
sOPF	**76.97±3.64**	**77.88±2.59**	73.33±3.64	72.83±2.35	71.77±3.36
SVM-RBF	**79.49±2.33**	**81.21±2.13**	**80.35±1.44**	**80.15±2.73**	**80.91±2.61**

Table 11.6: Meander dataset - Overall accuracies.

	dOPF	**hOPF**	**hOPF-25**	**hOPF-50**	**hOPF-75**
BC	**77.02±3.39**	**78.64±3.05**	68.89±4.59	68.79±3.92	68.38±4.26
sOPF	75.15± 3.09	**77.42±3.29**	68.89±4.37	68.28±4.13	67.68±3.66
SVM-RBF	**82.17±3.82**	**83.79±2.51**	79.04±2.21	79.55±2.18	78.74±3.63

Table 11.7: Dia-A dataset - Overall accuracies.

	dOPF	**hOPF**	**hOPF-25**	**hOPF-50**	**hOPF-75**
BC	**73.33±7.90**	**73.33±4.95**	66.47±6.53	**69.22±7.28**	**69.99±8.63**
sOPF	**73.53±7.86**	**73.33±5.83**	66.47±6.92	**68.24±7.31**	**68.82±7.36**
SVM-RBF	**79.22±5.38**	**77.25±4.51**	**76.86±4.56**	**75.69±4.37**	**79.61±7.49**

Table 11.8: Dia-B dataset - Overall accuracies.

	dOPF	**hOPF**	**hOPF-25**	**hOPF-50**	**hOPF-75**
BC	68.43±5.71	68.43±5.72	69.80±6.53	**75.29±5.54**	**73.14±7.45**
sOPF	67.45±6.43	67.45±6.43	69.41±6.65	**74.71±6.27**	**73.14±7.02**
SVM-RBF	70.39±6.43	74.51±5.52	74.31±9.06	**77.84±8.60**	**80.59±5.07**

We also investigated the accuracy rates in each class, as shown in Tables 11.9– 11.14. The representations learned by the hierarchical approach also figured among the best results in many situations. Once again, the more significant improvements (i.e., compared to dOPF) can be observed in the HC class in almost all scenarios, such as the ones in Circ-B and Dia-B datasets (i.e., the greatest ones). Compressed representations also presented competitive results, especially the most compressed one (hOPF-25) as one can observe in Circ-A and Circ-B datasets for HC class, and Dia-A dataset for PD class.

Table 11.9: Circ-A - Average accuracy rate for each class.

	BC		**sOPF**		**SVM-RBF**	
	HC	**PD**	**HC**	**PD**	**HC**	**PD**
dOPF	**81.85±10.17**	**84.17±13.34**	**81.85±10.17**	**82.50±12.32**	**80.74±11.85**	**88.75±11.62**
hOPF	77.78±8.91	**85.83±11.68**	78.52±8.62	**86.25±12.54**	77.41±8.26	**88.75±13.81**
hOPF-25	**85.56±8.08**	73.33±14.07	**85.93±8.36**	71.67±13.75	**80.74±12.04**	**85.83±10.94**
hOPF-50	**85.19±7.76**	**78.33±13.12**	**84.81±7.41**	75.42±11.68	**85.56±7.21**	**82.50±14.98**
hOPF-75	**84.44±7.33**	75.42±8.34	**84.44±7.33**	74.58±8.67	**84.44±7.62**	78.75±11.76

Table 11.10: Circ-B - Average accuracy rate for each class.

	BC		**sOPF**		**SVM-RBF**	
	HC	**PD**	**HC**	**PD**	**HC**	**PD**
dOPF	78.52±9.12	**83.75±9.97**	78.15±9.02	**82.08±9.99**	**79.63±11.04**	**79.58±12.38**
hOPF	78.52±6.92	**80.00±7.17**	77.41±8.26	**80.83±9.59**	**80.74±9.12**	**77.08±8.41**
hOPF-25	**86.67±5.86**	50.83±12.47	**86.67±5.86**	50.42±12.60	**86.30±10.47**	69.17±8.34
hOPF-50	**85.56±6.90**	49.58±16.61	**85.93±7.23**	49.17±16.68	**81.85±9.26**	**73.75±14.02**
hOPF-75	78.15±13.36	56.67±14.84	**78.15±14.77**	56.67±14.26	**86.67±8.08**	**71.25s±16.33**

Table 11.11: Spiral - Average accuracy rate for each class.

	BC		**sOPF**		**SVM-RBF**	
	HC	**PD**	**HC**	**PD**	**HC**	**PD**
dOPF	78.76±6.91	**75.48±5.89**	78.19±6.71	**75.59±6.09**	**84.95±5.91**	**73.33±5.91**
hOPF	**83.24±3.48**	**74.09±4.42**	**82.38±4.17**	**72.80±4.35**	**86.38±5.79**	**75.38±6.72**
hOPF-25	**84.95±4.07**	61.72±6.75	**84.38±3.98**	60.86±6.66	**87.14±6.01**	**72.69±6.17**
hOPF-50	81.90±4.06	62.69±5.13	82.10±3.58	62.37±4.79	**84.10±4.61**	**75.70±4.46**
hOPF-75	**82.67±4.48**	60.65±6.86	**82.19±4.58**	60.00±6.93	**87.52±6.33**	**73.44±6.06**

Table 11.12: Meander - Average accuracy rate for each class.

	BC		**sOPF**		**SVM-RBF**	
	HC	**PD**	**HC**	**PD**	**HC**	**PD**
dOPF	**80.76±5.50**	**72.80±7.00**	**77.43±7.56**	**72.58±7.44**	89.43±4.41	**73.98±7.41**
hOPF	**82.19±5.29**	**74.62±3.19**	**80.10±5.83**	**74.41±3.49**	**92.48±3.98**	**73.98±3.33**
hOPF-25	71.14±7.41	66.34±6.87	71.14±7.21	66.34±7.00	**90.38±4.09**	66.24±3.93
hOPF-50	72.86±6.39	64.19±6.63	72.10±6.48	63.98±6.81	**90.76±4.58**	66.88±4.35
hOPF-75	72.48±5.49	63.76±4.60	71.14±4.52	63.76±4.60	**90.29±5.92**	65.70±5.08

Table 11.13: Dia-A - Average accuracy rate for each class.

	BC		**sOPF**		**SVM-RBF**	
	HC	**PD**	**HC**	**PD**	**HC**	**PD**
dOPF	**79.26±7.99**	**66.67±13.08**	**78.52±6.60**	**67.92±12.47**	**72.96±9.59**	**86.25±10.88**
hOPF	72.96±8.19	**73.75±13.04**	72.96±8.62	**73.75±14.98**	72.22±7.86	**82.92±13.04**
hOPF-25	64.81±6.79	**68.33±15.13**	64.81±7.76	**68.33±14.46**	**73.70±9.02**	**80.42±10.79**
hOPF-50	72.59±10.59	**65.42±14.15**	**70.74±10.60**	**65.42±14.54**	**80.37±9.82**	70.42±11.68
hOPF-75	67.41±12.58	**72.92±19.00**	65.19±12.68	**72.92±18.85**	**76.67±10.33**	**82.92±14.07**

Table 11.14: Dia-B - Average accuracy rate for each class.

	BC		**sOPF**		**SVM-RBF**	
	HC	**PD**	**HC**	**PD**	**HC**	**PD**
dOPF	67.41±8.36	**69.58±11.78**	66.67±9.62	**68.33±12.60**	**73.70±13.36**	66.67±12.20
hOPF	67.41±8.36	**69.58±11.78**	66.67±9.62	**68.33±12.60**	**83.70±11.40**	64.17±11.92
hOPF-25	82.96±7.99	55.00±13.81	83.70±7.99	53.33±13.95	**81.85±7.11**	65.83±16.68
hOPF-50	**89.99±6.01**	58.75±11.52	**89.63±5.50**	**57.92±13.25**	**79.63±13.72**	**75.833±20.44**
hOPF-75	81.48±8.31	**63.75±14.01**	81.11±9.10	**64.17±14.07**	**78.52±11.67**	**82.92±13.04**

Since the difference between dOPF and hOPF relies on whether the visual words selected in the intermediate layers are used or not in the final dictionary, it must be concluded that the computational load for dictionary learning is the same.

11.5 Conclusion and Future Works

This work introduced a hierarchical learning approach using the Deep Optimum-Path Forest to design visual dictionaries. The proposed approach was assessed and compared against a previous approach proposed by Afonso et al. (AFONSO et al., 2017) in the context of Parkinson's disease identification. The experiments used six datasets derived from signal data collected when individuals were submitted to a handwriting exam. The exam is comprised of tasks supposed not to be trivial to Parkinson's disease patients, and the usage of signals allows to detect subtle variations.

The main contributions of this work rely on the introduction of the proposed approach itself, its application in the context of automatic PD detection, and the usage of Restricted Boltzmann Machine for data compression. Experimental results showed the potential of hierarchical learning approaches where interesting results were achieved. A general analysis pointed improvements in a few scenarios and the proposed approach always figured the best results. An in-depth investigation showed a more considerable improvement in accuracy in the healthy individuals class in most scenarios.

With respect to the compressed representations, RBM provided good models and achieved very interesting results (it either outperformed or was statistically similar to the original-sized representation and dOPF) in 12 out of 18 configurations (i.e., pair [dictionary learner, classifier]) for the HC class, and 10 out of 18 configurations for the PD class. Regarding future works, we aim to study different ways to create hierarchical representations instead of the concatenation.

Chapter 12

Conclusions

This dissertation focused on the study of advances in pattern recognition using the Optimum-Path Forest framework. To fulfill that purpose, it was proposed a survey on OPF (Chapter 2) covering its original algorithms and their applications, as well as studies that investigated and evaluated the supervised (Chapters 3–7) and unsupervised (Chapters 8–11) models under different scenarios. The findings allowed a better understanding of the framework and raised a few questions answered by a few of the works presented in this thesis.

How would a more straightforward and faster learner pattern recognition algorithm such as Optimum-Path Forest classifier behave in text classification? This question was answered by applying the original algorithm of the supervised OPF (OPF$_{cg}$) for event classification in drilling reports (Chapter 3). This task is challenging since the information is stored in a free-text format. The main idea of the work was to apply more straightforward techniques for both feature extraction and classification before going to more complex ones, although the latter one is not covered. Given that the problem is characterized by highly unbalanced classes, the techniques achieved satisfactory results being an interesting option, especially as a baseline for other techniques, for instance. OPF's performance was compared against k-NN and Bayesian classifier showing promising results with similar results to the other classifiers but with a lesser computational load.

How much the application of kernel functions could benefit OPF's learning process and consequently improve its accuracy? The work presented in Chapter 4 proposed the kernel-OPF (kOPF), which was evaluated under three kernel functions but not restricted to them: (i) identity, (ii) RBF, and (iii) Sigmoid. By employing kernels, OPF accuracy improved in some cases compared to the traditional OPF and with competitive results compared to SVM. Differently from SVM, OPF computes the kernel explicitly, which is a point that can be improved in later versions. It was also observed that OPF requires data normalization, which is another point to be further studied and improved.

Would the combination of an unsupervised manifold learning algorithm with the Optimum-Path Forest provide more accurate recognition rates? The experimental results showed the answer is yes (Chapter 5). The unsupervised manifold learning algorithm provided better distance, which can be observed in the improvement in accuracy over the traditional OPF. The improvement is even bigger for small training sets.

Can an evolutionary-based approach estimate better prototypes than using the minimum spanning tree? Experimental results showed a better performance of the evolutionary-based OPF (OPF$_{mh}$) over OPF in some situations (Chapter 6). The evolutionary-based approach computed a lower number of prototypes, which in most cases lead to better accuracy rates for testing samples. Considering the accuracy rates over the training set, a higher number of prototypes may cause overfitting, which is observed a drop in the accuracy over the testing set. Although the additional cost of incorporating a meta-heuristic technique, such an approach may be interesting for obtaining better data generalization. The study also suggests that changing the prototype selection mechanism can improve the learning process.

How does OPF behave in the context of Multiple Instance-learning? The study in Chapter 7 introduced OPF$_{cg}$ and OPF$_{knn}$ in the context of MI-learning. The label of a sample (bag) is based o the label of its instances. The proposed approach maps each bag as a node of the graph. The OPF classifiers were evaluated under three datasets with competitive results in two out of the three datasets. The sparsity in one of the datasets was an issue. By computing denser representations, the proposed MI-OPF had its accuracy improved.

Is the performance of BSO using OPF as good as to its traditional algorithm and how much the usage of OPF can benefit the optimization process? The work present in Chapter 8 showed that in terms of achieving the best average minimum fitness values, OPF was the best option by outperforming the original BSO algorithm and its version using Self-Organizing Maps for clustering. The results may be justified by the facts that OPF computes clusters on-the-fly (i.e., it does not require the number of clusters a priori), and the number of ideas is different at each iteration, as in a natural brainstorming process. However, OPF requires a higher computational load, which can be decreased by applying deep-driven OPF-based architecture, as presented in Chapter 9.

What if we develop a deep-driven OPF-based architecture to learn features? Will such representations provide better recognition rates than those from a "shallower" OPF? As aforementioned, OPF$_{uns}$ is mainly characterized for not requiring the number of clusters a priori. The first work (Chapter 9) on a deep-driven OPF-based architecture was developed to overcome cases where the number of clusters is known. Such an approach also copes with the time consumed by OPF$_{uns}$ for computing clusters. Situations with a considerable number of samples (e.g., hundreds of thousands) can be prohibitive. Therefore, a deep-based approach comes at hand. Although the term "*deep*", the approach resembles a hierarchical approach.

From the work in Chapter 9, it was observed the potential for learning features as in deep neural networks. The first work (Chapter 10) in this sense used features extracted from the last layer only. A latter work (Chapter 11) made use of a combination of features learned from all layers, which improved the accuracy rates in the context of Parkinson's disease recognition. Moreover, deep-OPF requires a considerably lower computational load than OPF$_{uns}$.

What about the future? As future works, there will be studied the application of polynomial kernels and simulation of different scenarios to better understand the OPF behavior. There will also be studies on how to cope with sparse representations since OPF accuracy is lower in such situations.

REFERENCES

AFONSO, L. et al. Automatic visual dictionary generation through optimum-path forest clustering. In: *2012 19th IEEE International Conference on Image Processing*. [S.l.: s.n.], 2012. p. 1897–1900.

AFONSO, L. et al. Learning to classify seismic images with deep optimum-path forest. In: *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2016. p. 401–407.

AFONSO, L. C. et al. Hierarchical learning using deep optimum-path forest. *Journal of Visual Communication and Image Representation*, v. 71, p. 102823, 2020.

AFONSO, L. C. et al. A recurrence plot-based approach for parkinson's disease identification. *Future Generation Computer Systems*, v. 94, p. 282–292, 2019. ISSN 0167-739X.

AFONSO, L. C. S. et al. Multiple-instance learning through optimum-path forest. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2019. p. 1–7.

AFONSO, L. C. S.; JUNIOR, L. P.; PAPA, J. P. Enhancing brain storm optimization through optimum-path forest. In: *IEEE 12th International Symposium on Applied Computational Intelligence and Informatics*. [S.l.: s.n.], 2018.

AFONSO, L. C. S.; PEDRONETTE, D. C. G.; PAPA, J. P. Improving optimum-path forest classification using unsupervised manifold learning. In: *24th International Conference on Pattern Recognition (accepted)*. [S.l.: s.n.], 2018.

AFONSO, L. C. S. et al. Parkinson's disease identification through deep optimum-path forest clustering. In: *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2017. p. 163–169.

AFONSO, L. C. S.; PEREIRA, D. R.; PAPA, J. P. A kernel-based optimum-path forest classifier. In: MENDOZA, M.; VELASTÍN, S. (Ed.). *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. [S.l.]: Springer International Publishing, 2018. p. 652–660.

AHMADI, S.-A. Human behavior-based optimization: a novel metaheuristic approach to solve complex optimization problems. *Neural Computing and Applications*, 2016.

ALLèNE, C. et al. Some links between extremum spanning forests, watersheds and min-cuts. *Image Vision Comput.*, Butterworth-Heinemann, Newton, MA, USA, v. 28, n. 10, p. 1460–1471, 2010. ISSN 0262-8856.

AMORIM, W. P.; FALCÃO, A. X.; CARVALHO, M. H. de. Semi-supervised pattern classification using optimum-path forest. In: *27th SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2014, Rio de Janeiro, Brazil, August 27-30, 2014*. [S.l.: s.n.], 2014. p. 111–118.

AMORIM, W. P.; FALCÃO, A. X.; PAPA, J. P. Multi-label semi-supervised classification through optimum-path forest. *Information Sciences*, v. 465, p. 86–104, 2018.

AMORIM, W. P. et al. Improving semi-supervised learning through optimum connectivity. *Pattern Recognition*, v. 60, n. Supplement C, p. 72–85, 2016.

ANDREWS, S.; TSOCHANTARIDIS, I.; HOFMANN, T. Support vector machines for multiple-instance learning. In: *Proceedings of the 15th International Conference on Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2002. p. 577–584.

ANTONIAK, M. et al. Natural language processing techniques on oil and gas drilling data. In: *SPE Intelligent Energy International Conference & Exhibition*. [S.l.]: Society of Petroleum Engineers, 2016. p. 1–6.

AVNI, U. et al. X-ray categorization and retrieval on the organ and pathology level, using patch-based visual words. *IEEE Transactions on Medical Imaging*, v. 30, n. 3, p. 733–746, March 2011.

BäCHLIN, M. et al. Wearable assistant for parkinson's disease patients with the freezing of gait symptom. *IEEE Transactions on Information Technology in Biomedicine*, IEEE Press, Piscataway, NJ, USA, v. 14, n. 2, p. 436–446, 2010.

BASHIR, M. A novel multisensoric system recording and analyzing human biometric features for biometric and biomedical applications. 2012. Disponível em: <https://epub.uni-regensburg.de/19673/>.

BELLET, A.; HABRARD, A.; SEBBAN, M. A survey on metric learning for feature vectors and structured data. *arXiv preprint*, 2013. Disponível em: <http://arxiv.org/abs/1306.6709>.

BIE, T. D.; MOMMA, M.; CRISTIANINI, N. Efficiently learning the metric with side-information. In: GAVALDÁ, R.; JANTKE, K. P.; TAKIMOTO, E. (Ed.). *Algorithmic Learning Theory: 14th International Conference*. [S.l.]: Springer Berlin Heidelberg, 2003. p. 175–189.

BISHOP, C. *Neural networks for pattern recognition*. [S.l.]: Oxford University Press, 1995.

BORGWARDT, K. M. et al. Protein function prediction via graph kernels. *Bioinformatics*, v. 21, p. i47, 2005.

BURGES, C. J. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, v. 2, n. 2, p. 121–167, 1998.

CAICEDO, J. C.; CRUZ, A.; GONZALEZ, F. A. Histopathology image classification using bag of features and kernel functions. In: COMBI, C.; SHAHAR, Y.; ABU-HANNA, A. (Ed.). *Artificial Intelligence in Medicine*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 126–135.

CARSON, C. et al. *Blobworld: A System for Region-based Image Indexing and Retrieval*. Berkeley, CA, USA, 1999.

CASTELO-FERNÁNDEZ, C.; CALDERÓN-RUIZ, G. Automatic video summarization using the optimum-path forest unsupervised classifier. 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. 760–767. ISBN 978-3-319-25750-1.

CASTELO-FERNÁNDEZ, C. et al. Improving the accuracy of the optimum-path forest supervised classifier for large datasets. In: BLOCH, I.; CESAR, R. M. (Ed.). *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 467–475.

CHAPELLE, O.; SCHLKOPF, B.; ZIEN, A. *Semi-Supervised Learning*. 1st. ed. [S.l.]: The MIT Press, 2010. ISBN 0262514125, 9780262514125.

CHEN, J.; XIE, Y.; NI, J. Brain storm optimization model based on uncertainty information. In: *2014 Tenth International Conference on Computational Intelligence and Security*. [S.l.: s.n.], 2014. p. 99–103.

CHENG, S. et al. Brain storm optimization algorithm: a review. *Artificial Intelligence Review*, v. 46, n. 4, p. 445–458, 2016.

CHENG, S. et al. Maintaining population diversity in brain storm optimization algorithm. In: *2014 IEEE Congress on Evolutionary Computation (CEC)*. [S.l.: s.n.], 2014. p. 3230–3237.

CHIACHIA, G. et al. Infrared face recognition by optimum-path forest. In: *2009 16th International Conference on Systems, Signals and Image Processing*. [S.l.: s.n.], 2009. p. 1–4.

CIVICIOGLU, P. Backtracking search optimization algorithm for numerical optimization problems. *Applied Mathematics and Computation*, Elsevier, v. 219, n. 15, p. 8121–8144, 2013.

COMANICIU, D. An algorithm for data-driven bandwidth selection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, v. 25, n. 2, p. 281–288, 2003.

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

COSTA, K. A. P. et al. A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks. *Information Sciences*, p. 95–108, 2015.

CRISTIANINI, N.; SHAWE-TAYLOR, J. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. [S.l.]: Cambridge University Press, 2000.

CSURKA, G. et al. Visual categorization with bags of keypoints. In: *Proceedings of the Workshop on Statistical Learning in Computer Vision*. [S.l.: s.n.], 2004. p. 1–22.

DAS, R. A comparison of multiple classification methods for diagnosis of parkinson disease. *Expert Systems with Applications*, v. 37, n. 2, p. 1568 – 1572, 2010.

DAWSON, C.; VERKUIL, H. From a daily drilling report to a data and performance management tool. In: *SPE Intelligent Energy Conference & Exhibition*. [S.l.]: Society of Petroleum Engineers, 2014. p. 1–8.

DHEERU, D.; KARRA, E. K. *UCI Machine Learning Repository*. 2017. Disponível em: <http://archive.ics.uci.edu/ml>.

DIETTERICH, T. G.; LATHROP, R. H.; LOZANO-PÉREZ, T. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, v. 89, n. 1, p. 31 – 71, 1997. ISSN 0004-3702.

DORIGO, M.; CARO, G. D. Ant colony optimization: a new meta-heuristic. In: *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*. [S.l.: s.n.], 1999. v. 2, p. 1477 Vol. 2.

DUAN, H.; LI, C. Quantum-behaved brain storm optimization approach to solving loney's solenoid problem. *IEEE Transactions on Magnetics*, v. 51, n. 1, p. 1–7, 2015.

EBERHART, R.; KENNEDY, J. A new optimizer using particle swarm theory. In: *Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on*. [S.l.: s.n.], 1995. p. 39–43.

ECKMANN, J.-P.; KAMPHORST, S. O.; RUELLE, D. Recurrence plots of dynamical systems. *Europhysics Lettes*, v. 9, n. 4, p. 973–977, 1997.

EL-ABD, M. Brain storm optimization algorithm with re-initialized ideas and adaptive step size. In: *2016 IEEE Congress on Evolutionary Computation (CEC)*. [S.l.: s.n.], 2016. p. 2682–2686.

EROL, O. K.; EKSIN, I. A new optimization method: Big bang–big crunch. *Advances in Engineering Software*, v. 37, n. 2, p. 106 – 111, 2006.

ESMAEL, B. et al. A hybrid multiple classifier system for recognizing usual and unusual drilling events. In: *IEEE International Instrumentation and Measurement Technology Conference*. [S.l.: s.n.], 2012. (I2MTC), p. 1754–1758.

FALCÃO, A. X.; STOLFI, J.; LOTUFO, R. A. The image foresting transform: theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 26, n. 1, p. 19–29, 2004.

FERNANDES, S. E. N.; PAPA, J. P. Improving optimum-path forest learning using bag-of-classifiers and confidence measures. *Pattern Analysis and Applications*, Springer, p. 1–14, 2017.

FOULDS, J.; FRANK, E. A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, Cambridge University Press, v. 25, n. 1, p. 1–25, 2010.

FUNDATION, N. P. *The Parkinson's Foundation's Moving Day Walks Fund Local Parkinson's Programs in 2017*. 2017. `http://http://www.parkinson.org/`.

GäRTNER, T. A survey of kernels for structured data. *SIGKDD Explorations Newsletter*, ACM, New York, NY, USA, v. 5, n. 1, p. 49–58, 2003.

GÄRTNER, T.; FLACH, P.; WROBEL, S. On graph kernels: Hardness results and efficient alternatives. In: ____. *Learning Theory and Kernel Machines*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 129–143.

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

GUILHERME, I. R. et al. Fast petroleum well drilling monitoring through optimum-path forest. *Journal of Next Generation Information Technology*, v. 1, p. 77–85, 2010.

GUILHERME, I. R. et al. Petroleum well drilling monitoring through cutting image analysis and artificial intelligence techniques. *Engineering Applications of Artificial Intelligence*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 24, n. 1, p. 201–207, 2011.

GUILHERME, I. R. et al. An ontology based for drilling report classification. In: ____. *Advances in Artificial Intelligence: 5th Mexican International Conference on Artificial Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016. (MICAI), p. 1037–1046.

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

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

HINTON, G. E. Training products of experts by minimizing contrastive divergence. *Neural Computation*, v. 14, n. 8, p. 1771–1800, 2002.

HOFFIMANN, J. et al. Sequence mining and pattern analysis in drilling reports with deep natural language processing. *arXiv.org*, p. 1–7, 2017. Disponível em: <https://arxiv.org/abs/1712.01476>.

HOFMANN, T.; SCHöLKOPF, B.; SMOLA, A. J. Kernel methods in machine learning. *Annals of Statistics*, v. 36, n. 3, p. 1171–1220, 2008.

HOI, S. C.; LIU, W.; CHANG, S.-F. Semi-supervised distance metric learning for collaborative image retrieval and clustering. *ACM Transactions on Multimedia Computing, Communications, and Applications*, ACM, New York, NY, USA, v. 6, n. 3, p. 18:1–18:26, 2010. ISSN 1551-6857.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

IWASHITA, A. S. et al. A path- and label-cost propagation approach to speedup the training of the optimum-path forest classifier. *Pattern Recognition Letters*, v. 40, p. 121–127, 2014.

JADHAV, H. T. et al. Brain storm optimization algorithm based economic dispatch considering wind power. In: *2012 IEEE International Conference on Power and Energy (PECon)*. [S.l.: s.n.], 2012. p. 588–593.

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, v. 31, n. 8, p. 651 – 666, 2010. ISSN 0167-8655.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: a review. *ACM Computing Surveys*, ACM, New York, NY, USA, v. 31, n. 3, p. 264–323, 1999.

JELINEK, F.; MERCER, R. L. Interpolated estimation of markov source parameters from sparse data. In: GELSEMA, E. S.; KANAL, L. N. (Ed.). *Proceedings, Workshop on Pattern Recognition in Practice*. Amsterdam: North Holland, 1980. p. 381–397.

JONES, K. S.; WALKER, S.; ROBERTSON, S. E. A probabilistic model of information retrieval: Development and comparative experiments. *Information Processing and Management*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 36, n. 6, p. 779–808, 2000. ISSN 0306-4573.

JORDEHI, A. R. Brainstorm optimisation algorithm (bsoa): An efficient algorithm for finding optimal location and setting of facts devices in electric power systems. *International Journal of Electrical Power & Energy Systems*, v. 69, p. 48 – 57, 2015. ISSN 0142-0615.

JOULIN, A. et al. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016. Disponível em: <http://arxiv.org/abs/1607.01759>.

KARABOGA, D.; AKAY, B.; OZTURK, C. Artificial bee colony (abc) optimization algorithm for training feed-forward neural networks. In: *Proceedings of the 4th International Conference on Modeling Decisions for Artificial Intelligence*. Berlin, Heidelberg: Springer-Verlag, 2007. (MDAI '07), p. 318–329.

KARABOGA, D.; BASTURK, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of Global Optimization*, v. 39, n. 3, p. 459–471, 2007.

KASHIMA, H.; TSUDA, K.; INOKUCHI, A. Marginalized kernels between labeled graphs. In: *Proceedings of the Twentieth International Conference on Machine Learning*. [S.l.]: AAAI Press, 2003. p. 321–328.

KASHIMA, H.; TSUDA, K.; INOKUCHI, A. Kernels for graphs. In: ____. *Kernel Methods in Computational Biology*. [S.l.]: MIT Press, 2004.

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

KEELER, J. D.; RUMELHART, D. E.; LEOW, W.-K. *Integrated segmentation and recognition of hand-printed numerals*. 1981. (MCC technical report, ACT-NN-010-91).

KOHONEN, T. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, Springer-Verlag, v. 43, n. 1, p. 59–69, 1982.

KOHONEN, T. *The Self-Organizing Maps*. 3rd. ed. [S.l.]: Springer, 2001. 501 p.

KONDOR, R. I.; LAFFERTY, J. D. Diffusion kernels on graphs and other discrete input spaces. In: *Proceedings of the Nineteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002. (ICML '02), p. 315–322.

KPALMA, K.; RONSIN, J. An overview of advances of pattern recognition systems in computer vision. In: *Vision Systems: Segmentation and Pattern Recognition*. [S.l.]: InTech, 2007.

LECUN, Y.; BENGIO, Y.; HINTON, G. E. Deep learning. *Nature*, v. 521, p. 436–444, 2015.

LEE, J.-E.; JIN, R.; JAIN, A. K. Rank-based distance metric learning: An application to image retrieval. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2008. p. 1–8. ISSN 1063-6919.

LEES, A. J.; HARDY, J.; REVESZ, T. Parkinson's disease. *The Lancet*, v. 373, n. 9680, p. 2055–2066, 2009.

LIONG, V. E.; LU, J.; GE, Y. Regularized local metric learning for person re-identification. *Pattern Recognition Letters*, v. 68, p. 288 – 296, 2015. ISSN 0167-8655.

LU, J.; TAN, Y. P.; WANG, G. Discriminative multimanifold analysis for face recognition from a single training sample per person. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 35, n. 1, p. 39–51, 2013.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, California: University of California Press, 1967. p. 281–297.

MAHÉ, P. et al. Extensions of marginalized graph kernels. In: *Proceedings of the Twenty-first International Conference on Machine Learning*. New York, NY, USA: ACM, 2004. (ICML '04), p. 70–.

MAKI, B. E.; MCILROY, W. E. Change-in-support balance reactions in older persons: An emerging research area of clinical importance. *Neurologic Clinics*, Elsevier BV, v. 23, n. 3, p. 751–783, 2005.

MARCHETTI, G. F.; WHITNEY, S. L. Older adults and balance dysfunction. *Neurologic Clinics*, Elsevier BV, v. 23, n. 3, p. 785–805, 2005.

MARON, O.; RATAN, A. L. Multiple-instance learning for natural scene classification. In: *Proceedings of the Fifteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (ICML '98), p. 341–349.

MIKA, S. et al. Fisher discriminant analysis with kernels. In: *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop*. [S.l.: s.n.], 1999. p. 41–48.

MÜLLER, H.; PUN, T.; SQUIRE, D. Learning from user behavior in image retrieval: Application of market basket analysis. *International Journal of Computer Vision*, v. 56, n. 1, p. 65–77, 2004. ISSN 1573-1405.

NICKABADI, A.; EBADZADEH, M. M.; SAFABAKHSH, R. A novel particle swarm optimization algorithm with adaptive inertia weight. *Applied Soft Computing*, v. 11, p. 3658–3670, 2011.

NYE, M.; SAXE, A. Are efficient deep representations learnable? In: *Proceedings of the International Conference on Learning Representations*. [s.n.], 2018. Disponível em: <https://openreview.net/forum?id=B1HI4FyvM>.

ORENGO, V. M.; BURIOL, L. S.; COELHO, A. R. A study on the use of stemming for monolingual ad-hoc portuguese information retrieval. In: PETERS, C. et al. (Ed.). *Evaluation of Multilingual and Multi-modal Information Retrieval*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 91–98.

PAATERO, P.; TAPPER, U. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, John Wiley & Sons, Ltd., v. 5, n. 2, p. 111–126, 1994. ISSN 1099-095X.

PAL, A.; PAL, S. K. Pattern recognition: Evolution of methodologies and data mining. In: ____. *Pattern Recognition*. [S.l.]: World Scientific, 2011. p. 1–23.

PAPA, J. P.; FALCÃO, A. X. A new variant of the optimum-path forest classifier. In: *Advances in Visual Computing, 4th International Symposium, ISVC 2008, Las Vegas, NV, USA, December 1-3, 2008. Proceedings, Part I*. [S.l.: s.n.], 2008. p. 935–944.

PAPA, J. P.; FALCÃO, A. X. A learning algorithm for the optimum-path forest classifier. In: TORSELLO, A.; ESCOLANO, F.; BRUN, L. (Ed.). *Graph-Based Representations in Pattern Recognition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. (Lecture Notes in Computer Science, v. 5534), p. 195–204.

PAPA, J. P.; FALCÃO, A. X. On the training patterns pruning for optimum-path forest. In: FOGGIA, P.; SANSONE, C.; VENTO, M. (Ed.). *Image Analysis and Processing – ICIAP 2009*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 259–268.

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. et al. Fast and accurate holistic face recognition using optimum-path forest. In: *2009 16th International Conference on Digital Signal Processing*. [S.l.: s.n.], 2009. p. 1–6.

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. et al. A discrete approach for supervised pattern recognition. In: *Combinatorial Image Analysis, 12th International Workshop, IWCIA 2008, Buffalo, NY, USA, April 7-9, 2008. Proceedings*. [S.l.: s.n.]. p. 136–147.

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. LibOPT: An open-source platform for fast prototyping soft optimization techniques. *ArXiv e-prints*, 2017. Http://adsabs.harvard.edu/abs/2017arXiv170405174P.

PAPA, J. P.; SUZUKI, C. T. N.; FALCÃO, A. X. LibOPF: A library for the design of optimum-path forest classifiers. 2014.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PEDRONETTE, D. C. G.; GONÇALVES, F. M. F.; GUILHERME, I. R. Unsupervised manifold learning through reciprocal knn graph and connected components for image retrieval tasks. *Pattern Recognition*, v. 75, p. 161 – 174, 2018. ISSN 0031-3203.

PEDRONETTE, D. C. G.; TORRES, R. da S. Image re-ranking and rank aggregation based on similarity of ranked lists. *Pattern Recognition*, v. 46, n. 8, p. 2350 – 2360, 2013. ISSN 0031-3203.

PENATTI, O. A.; VALLE, E.; TORRES, R. da S. Comparative study of global color and texture descriptors for web image retrieval. *Journal of Visual Communication and Image Representation*, v. 23, n. 2, p. 359 – 380, 2012. ISSN 1047-3203.

PEREIRA, C. R. et al. An optimum-path forest framework for intrusion detection in computer networks. *Engineering Applications of Artificial Intelligence*, v. 25, n. 6, p. 1226 – 1234, 2012.

PEREIRA, C. R. et al. A new computer vision-based approach to aid the diagnosis of parkinson's disease. *Computer Methods and Programs in Biomedicine*, Elsevier North-Holland, Inc., New York, NY, USA, v. 136, p. 79–88, 2016.

PEREIRA, C. R. et al. A step towards the automated diagnosis of parkinson's disease: Analyzing handwriting movements. In: *IEEE 28th International Symposium on Computer-Based Medical Systems*. [S.l.: s.n.], 2015. p. 171–176.

PEREIRA, C. R. et al. Deep learning-aided parkinson's disease diagnosis from handwritten dynamics. In: *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2016. p. 340–346.

PISANI, R. et al. Land use image classification through optimum-path forest clustering. In: *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*. [S.l.]: IEEE Press, 2011. p. 826–829.

PONTI, M.; RIVA, M. An incremental linear-time learning algorithm for the optimum-path forest classifier. *Information Processing Letters*, v. 126, p. 1–6, 2017.

PRIYADARSHY, S. et al. Framework for prediction of npt causes using unstructured reports. In: *Offshore Technology Conference*. [S.l.]: Offshore Technology Conference, 2017. p. 1–6.

QIN, D. et al. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In: *Computer Vision and Pattern Recognition 2011*. [S.l.: s.n.], 2011. p. 777–784.

QUELLEC, G. et al. Multiple-instance learning for medical image and video analysis. *IEEE Reviews in Biomedical Engineering*, v. 10, p. 213–234, 2017.

QUINLAN, J. R. Induction of decision trees. *Machine Learning*, v. 1, n. 1, p. 81–106, Mar 1986.

RAMON, J.; RAEDT, L. D. Multi instance neural networks. In: *Proceedings of the ICML-2000 workshop on attribute-value and relational learning*. [S.l.: s.n.], 2000.

RASSENFOSS, S. Mining daily driller's reports looking for telling patterns. *Journal of Petroleum Technology*, Society of Petroleum Engineers, v. 67, n. 6, p. 1–2, 2015.

RAUBER, A.; MERKL, D.; DITTENBACH, M. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, v. 13, n. 6, p. 1331–1341, 2002.

RIGAS, G. et al. Assessment of tremor activity in the parkinson's disease using a set of wearable sensors. *IEEE Transactions on Information Technology in Biomedicine*, v. 16, n. 3, p. 478–487, 2012.

ROCHA, L. M.; CAPPABIANCO, F. A. M.; FALCÃO, A. X. Data clustering as an optimum-path forest problem with applications in image analysis. *International Journal of Imaging Systems and Technology*, Wiley Periodicals, v. 19, n. 2, p. 50–68, 2009.

ROCHA, L. M.; FALCÃO, A. X.; MELONI, L. G. P. A robust extension of the mean shift algorithm using optimum-path forest. In: *Image Analysis - From Theory to Applications. Proceedings of IWCIA 2008 Special Track on Applications, Buffalo, NY, USA, April 7-9, 2008.* [S.l.: s.n.], 2008. p. 29–38.

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, 2014. ISSN 0957-4174.

ROKACH, L.; MAIMON, O. Top-down induction of decision trees classifiers - a survey. *Transactions on Systems, Man, and Cybernetics Part C*, IEEE Press, Piscataway, NJ, USA, v. 35, n. 4, p. 476–487, 2005.

ROSA, G. H. et al. On the training of artificial neural networks with radial basis function using optimum-path forest clustering. In: *22nd International Conference on Pattern Recognition.* [S.l.: s.n.], 2014. p. 1472–1477.

S., K. P. F. R. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Taylor & Francis, v. 2, n. 11, p. 559–572, 1901.

SALTON, G.; MCGILL, M. J. *Introduction to Modern Information Retrieval.* New York, NY, USA: McGraw-Hill, Inc., 1986. ISBN 0070544840.

SAMà, A. et al. Determining the optimal features in freezing of gait detection through a single waist accelerometer in home environments. *Pattern Recognition Letters*, 2017.

SCHöLKOPF, B.; SMOLA, A.; MüLLER, K.-R. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computing*, MIT Press, Cambridge, MA, USA, v. 10, n. 5, p. 1299–1319, 1998.

SCHWENKER, F.; TRENTIN, E. Pattern classification and clustering: A review of partially supervised learning approaches. *Pattern Recognition Letters*, v. 37, p. 4 – 14, 2014.

SHAHBAKHI, M.; FAR, D. T.; TAHAMI, E. Speech analysis for diagnosis of parkinson's disease using genetic algorithm and support vector machine. *Journal of Biomedical Science and Engineering*, v. 07, n. 04, p. 147–156, 2014.

SHI, J.; MALIK, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, v. 22, n. 8, p. 888–905, Aug 2000.

SHI, Y. Brain storm optimization algorithm. In: ____. *Advances in Swarm Intelligence: Second International Conference, ICSI 2011, Chongqing, China, June 12-15, 2011, Proceedings, Part I.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 303–309.

SHI, Y. An optimization algorithm based on brainstorming process. *International Journal of Swarm Intelligence Research*, v. 2, p. 35–62, 10 2011.

SIDAHMED, M.; COLEY, C. J.; SHIRZADI, S. Augmenting operations monitoring by mining unstructured drilling reports. In: *SPE Digital Energy Conference & Exhibition.* [S.l.]: Society of Petroleum Engineers, 2015. p. 1–13.

SMOLA, A. J.; KONDOR, I. R. Kernels and regularization on graphs. In: *Proceedings of the Annual Conference on Computational Learning Theory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 144–158.

SOUSA, G. J. et al. Pattern analysis in drilling reports using optimum-path forest. In: *IEEE 12th International Symposium on Applied Computational Intelligence and Informatics (accepted)*. [S.l.: s.n.], 2018. p. 1–8.

SOUZA, L. A. D. et al. Barrett's esophagus identification using optimum-path forest. In: *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. [S.l.: s.n.], 2017. p. 308–314.

SOUZA, R.; RITTNER, L.; LOTUFO, R. A. A comparison between k-optimum path forest and k-nearest neighbors supervised classifiers. *Pattern Recognition Letters*, Elsevier Science Inc., New York, NY, USA, v. 39, p. 2–10, 2014. ISSN 0167-8655.

SPADOTTO, A. et al. Improving parkinson's disease identification through evolutionary-based feature selection. In: *Proceedings of the 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. [S.l.: s.n.], 2011. p. 7857–7860.

SPADOTTO, A. et al. Parkinson's disease identification through optimum-path forest. In: *Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. Buenos Aires, Argentina: [s.n.], 2010. p. 6087–6090.

STEHLING, R. O.; NASCIMENTO, M. A.; FALCÃO, A. X. A compact and efficient image retrieval approach based on border/interior pixel classification. In: *Proceedings of the Eleventh International Conference on Information and Knowledge Management*. New York, NY, USA: ACM, 2002. (CIKM '02), p. 102–109. ISBN 1-58113-492-4.

THEODORAKOPOULOS, I. et al. Local manifold distance based on neighborhood graph reordering. *Pattern Recognition*, v. 53, p. 195 – 211, 2016. ISSN 0031-3203.

TSUDA, K.; KIN, T.; ASAI, K. Marginalized kernels for biological sequences. *Bioinformatics*, v. 18, p. S268, 2002.

VISHWANATHAN, S. V. N. et al. Graph kernels. *Journal of Machine Learning Research*, JMLR.org, v. 11, p. 1201–1242, 2010.

WANG, J. et al. Bag-of-words representation for biomedical time series classification. *Biomedical Signal Processing and Control*, v. 8, n. 6, p. 634 – 644, 2013. ISSN 1746-8094.

WEIDMANN, N.; FRANK, E.; PFAHRINGER, B. A two-level learning method for generalized multi-instance problems. In: *Proceedings of the 14th European Conference on Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 2003. (ECML'03), p. 468–479.

WIEMER-HASTINGS, P.; WIEMER-HASTINGS, K.; GRAESSER, A. Latent semantic analysis. In: *Proceedings of the 16th International Joint Conference on Artificial Intelligence*. [S.l.: s.n.], 2004. p. 1–14.

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

YANG, S.-S.; KARAMANOGLU, M.; HE, X. Flower pollination algorithm: A novel approach for multiobjective optimization. *Engineering Optimization*, v. 46, n. 9, p. 1222–1237, 2014.

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

YANG, X.-S.; XINGSHI, H. Firefly algorithm: Recent advances and applications. *International Journal of Swarm Intelligence*, v. 1, 08 2013.

YU, T. et al. Interpretative topic categorization via deep multiple instance learning. In: *2018 International Joint Conference on Neural Networks*. [S.l.: s.n.], 2018. p. 1–7.

ZHAN, Z. h. et al. A modified brain storm optimization. In: *2012 IEEE Congress on Evolutionary Computation*. [S.l.: s.n.], 2012. p. 1–8. ISSN 1089-778X.

ZHANG, Q.; GOLDMAN, S. A. Em-dd: An improved multiple-instance learning technique. In: DIETTERICH, T. G.; BECKER, S.; GHAHRAMANI, Z. (Ed.). *Advances in Neural Information Processing Systems 14*. [S.l.]: MIT Press, 2002. p. 1073–1080.

ZHAO, Y. J. et al. Factors affecting health-related quality of life amongst asian patients with parkinson's disease. *European Journal of Neurology*, Wiley-Blackwell, v. 15, n. 7, p. 737–742, 2008.

ZHOU, D. et al. Ranking on data manifolds. In: *Proceedings of the 16th International Conference on Neural Information Processing Systems*. [S.l.: s.n.], 2003. (NIPS'03), p. 169–176.

GLOSSARY

ABC – *Artificial Bee Colony*

ACC-SPy – *Auto Color Correlograms Spatial Pyramid*

AIR – *Articulation-Invariant Representation*

AIWPSO – *Adaptive Inertia Weight Particle Swarm Optimization*

ANN-MLP – *Artificial Neural Networks using Multi-Layer Perceptrons*

ASC – *Aspect Shape Context*

BAS – *Beam Angle Statistics*

BA – *Bat Algorithm*

BC – *Naïve Bayes classifer*

BF – *best fitness value*

BHA – *Black Hole algorithm*

BIC – *Border/Interior Pixel Classification*

BM – *Best Matching*

BSA – *Backtracking Search Optimization Algorithm*

BSO – *Brain Storm Optimization*

BSO – *Brainstorm Optimization*

CBIR – *Content-Based Image Retrieval*

CCOM – *Color Co-Occurrence Matrix*

CC – *Connected Component*

CEDD-SPy – *Color and Edge Directivity Descriptor Spatial Pyramid*

CFD – *Contour Features Descriptor*

CNN-Caffe – *Convolutional Neural Network by Caffe*

CS – *Cuckoo Search*

C – *Completeness*

DWDR – *Daily well drilling reports*

DWT – *Discrete Wavelet Transform*

FA – *Firefly Algorithm*

FCTH-SPy – *Fuzzy Color and Texture Histogram Spatial Pyramid*

FPA – *Flower Polinization Algorithm*

GA – *Genetic Algorithm*

HS – *Harmony Search*

H – *Homogeneity*

IDSC – *Inner Distance Shape Context*

JCD-SPy – *Joint Composite Descriptor Spatial Pyramid*

LAS – *Local Activity Spectrum*

LBP-SPy – *Local Binary Patterns Spatial Pyramid*

LBP – *Local Binary Patterns*

LSA – *Latent Semantic Analysis*

MBF – *mean of best fitness value*

MI – *Multiple-instance*

NLP – *natural language processing*

NMF – *Non-negative Matrix Factorization*

OPF$_{mh}$ – *Metaheuristic-based OPF*

OPF – *Optimum-Path Forest*

PCA – *Principal Component Analysis*

PD – *Parkinson's disease*

PR – *Pattern recognition*

RBF – *Radial Basis Function*

RBM – *Restricted Boltzmann Machine*

RSLP – *Removedor de Sufixos da Língua Portuguesa*

SDBF – *standard deviation of BF*

SOM – *Self-Organizing Maps*

SS – *Segment Saliences*

SVM-RBF – *SVM using a Radial Basis Function kernel with parameter optimization*

SVM – *Support Vector Mechines*

TF-IDF – *Term Frequency - Inverse Document Frequency*

V – *V-measure*

dOPF – *deep-hierarchical OPF*

hOPF – *Hierarchical OPF*

kOPF – *kernel-based OPF*

sOPF – *supervised OPF*

Appendix A

OTHER PUBLICATIONS

This Appendix lists other works published or under review during the Ph.D. program in chronological order from the most recent to the oldest ones.

A.1 Related works

This section presents the studies related to the OPF framework.

A.1.1 Journal papers

- *Learning Visual Representations with Optimum-Path Forest and its Applications to Barrett's Esophagus and Adenocarcinoma Diagnosis:* This work was accepted in the Neural Computing and Applications (Qualis-CC A1).

A.1.2 Conference papers

- *Information Ranking Using Optimum-Path Forest:* This work was accepted and presented in the 2020 International Joint Conference on Neural Networks (IJCNN) (Qualis-CC - A2).
- *Discovering Patterns within the Drilling Reports using Artificial Intelligence for Operation Monitoring:* This work was accepted and presented in the Offshore Technology Conference Brasil (OCT) (Qualis-CC not available).
- *Qualidade no fornecimento de energia: contribuições de uma ferramenta inteligente para gestão de falhas incipientes em transformadores:* This works was accepted and presented

in the XIII Conferência Brasileira sobre Qualidade da Energia Elétrica (CBQEE) - 2019 (Qualis-CC not available).

- *Barrett's Esophagus Identification Using Optimum-Path Forest:* This work was accepted and presented in the 30th Conference on Graphics, Pattern and Images (SIBGRAPI) - 2017 (Qualis-CC - A3).

A.2 Other works

This section presents the studies that are not related to the OPF framework.

A.2.1 Journal papers

- *Evolving Neural Conditional Random Fields for drilling report classification:* This work was accepted in the Journal of Petroleum Science and Engineering (Qualis-CC A1).
- *Bag of Samplings for Computer-assisted Parkinson's Disease Diagnosis based on Recurrent Neural Networks:* This work was accepted in the Computers in Biology and Medicine (Qualis-CC A2).
- *A recurrence plot-based approach for Parkinson's disease identification:* This work was accepted in the Future Generation Computer Systems (Qualis-CC A1) - first author.
- *A Fast Approach for Unsupervised Karst Feature Identification using GPU:* This work was accepted in the Computer & Geosciences journal (Qualis-CC A2).
- *Feature selection through binary brain storm optimization:* This work was accepted in the Computers and Electrical Engineering (Qualis-CC A2).

A.2.2 Conference papers

- *A Hybrid Approach For Breast Mass Categorization:* This work was accepted and presented in VII ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing (VipIMAGE) - 2019 (Qualis-CC not available).
- *Campos Aleatórios Condicionais Evolutivos para Classificação de Boletins de Perfuração:* This works was accepted and presented in the Encontro Nacional de Construção de Poços de Petróleo e Gás (ENAHPE) - 2019 (Qualis-CC not available).

- *Quaternionic Flower Pollination Algorithm:* This work was accepted and presented in the International Conference on Computer Analysis of Images and Patterns (CAIP) - 2017 (Qualis-CC - A4).