Multiple-Instance Learning through Optimum-Path Forest

Luis C. S. Afonso
Department of Computing
UFSCar - Federal University of São Carlos
São Carlos, Brazil
sugi.luis@ufscar.br

Danilo Colombo
Cenpes
Petróleo Brasileiro S.A.
Rio de Janeiro - RJ, Brazil
colombo.danilo@petrobras.com.br

Kelton A. P. Costa and João P. Papa
Department of Computing
UNESP - São Paulo State University
Bauru, Brazil
{kelton.costa, joao.papa}@unesp.br

***Abstract*—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 this paper, we introduced the Optimum-Path Forest (OPF) classifier to the context of multiple-instance learning paradigm, and we evaluated 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 are very much suitable to handle problems in the multiple-instance learning paradigm.**

I. 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. 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 [1].

Keeler et al. [2] and Dietterich et al. [3] 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. [3] proposed the three Axis-Parallel Rectangle (APR) 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 [4], which was one of the most popular benchmark datasets used in the multiple-instance learning research community for years.

Quellec et al. [5] 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. [6] 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.

Some years ago, Papa et al. [7], [8] proposed the Optimum-Path Forest (OPF), which is a framework for the design of classifiers based on graph partitions. The OPF models the problem of pattern recognition as a reward-competition process, where some predefined samples called *prototypes* compete among themselves to conquer the remaining samples. The competition process aims to assign the lowest cost to each non-prototype sample and ends up partitioning the dataset into optimum-path trees. The OPF framework comprises supervised [7]–[9], semi-supervised [10], [11], and unsupervised learning approaches [12].

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. Recently, Sousa et al. [13] 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

The authors are grateful to FAPESP grants #2013/07375-0, #2014/12236-1, #2016/19403-6, as well as #2017/22905-6, CNPq grant #307066/2017-7 and #429003/2018-8, and Petrobras (grant #2014/00545-0) for their financial support.

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 II briefly revisits the theoretical background concerning the MI learning paradigm and OPF classifiers. Section III presents the proposed approach and the methodology, while Section IV discusses the experiments. Finally, Section V states conclusions and future works.

II. Theoretical Background

In this section, we present the theoretical background concerning the multiple-instance learning paradigm and pattern classification through Optimum-Path Forest.

A. Multiple-Instance Learning

Let $\mathcal{I} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$ be a set of labeled instances (i.e., samples) such that $\boldsymbol{x}_i \in \Re^n$ and $y_i \in \{-1, 1\}$ denote a sample and its label, respectively. Standard machine learning techniques, hereinafter called "single-instance" approaches (SI), usually partition $\mathcal{I}$ into training and testing sets for learning purposes. Such 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 $\mathcal{B} = \{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_p\}$ a set of bags derived from $\mathcal{I}$ such that $\mathcal{B}_i$ contains a set of samples. Additionally, assume that $\mathcal{B}_i \cap \mathcal{B}_j = \emptyset$, $i \neq j$, and $\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2 \cup \ldots \cup \mathcal{B}_p$. In a nutshell, MI-based techniques aim at learning a function $f : \mathcal{B} \rightarrow \{-1, 1\}$.

In binary-driven MI problemas, the label of $\mathcal{B}_i$ is considered positive if there exists, at least, a single positive sample that belongs to it. On the other hand, the label of $\mathcal{B}_i$ is considered negative when all its samples are assigned to the negative class. Such an approach is also regarded as *presence-based* [14]. 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.

B. Optimum-Path Forest

In this section, we revisit the theory concerning Optimum-Path Forest-based classification. As aforementioned, OPF is rather a framework than a single technique, which means we can design our own classifier by just changing some modules (i.e., the adjacency relation, prototype estimation methodology, and path-cost function). Regarding supervised learners, we currently have two variants: the first one that makes use of a complete graph as adjacency relation [7], [8] and the other that employs a k-neighborhood graph [9]. Sections II-B1 and II-B2 discuss these two approaches.

1) Optimum-Path Forest with Complete Graph (OPF$_{cpl}$): Any OPF-based classifier encodes a given sample $(\boldsymbol{x}_i, y_i) \in \mathcal{I}$ as a graph node[1], $i = 1, 2, \ldots, |\mathcal{I}|$. Additionaly, we have that $\mathcal{I} = \mathcal{I}^{tr} \cup \mathcal{I}^{ts}$, in which $\mathcal{I}^{tr}$ and $\mathcal{I}^{ts}$ stand for training and testing partitions (i.e., sets), respectively. The connections between nodes $\boldsymbol{x}_i$ and $\boldsymbol{x}_j \in \mathcal{I}$ are weighted by a distance function $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and defined according to some pre-defined adjacency relation $\mathcal{A}$, provided that $j = 1, 2, \ldots, |\mathcal{I}|$ and $i \neq j$.

In the method described in this section, one can define a graph $G^{tr} = (\mathcal{I}^{tr}, \mathcal{A})$ that contains connections among nodes defined by a complete graph adjacency relation, i.e., all nodes $\boldsymbol{x}_i \in \mathcal{I}^{tr}$ are connected to each other. A path $\pi_{\boldsymbol{x}_i}$ is defined as a sequence of adjacent and distinct nodes in G^{tr} with terminus at node $\boldsymbol{x}_i$. On the other hand, a path comprised of a single node $\boldsymbol{x}_j$ is called *trivial* and denoted as $\langle \boldsymbol{x}_j \rangle$.

At a glance, the OPF aims at computing $\min\{f(\pi_{\boldsymbol{x}_i})\}$ for any path starting at the prototype set $\mathcal{P}$ and terminus at sample $\boldsymbol{x}_i$, $\forall\ \boldsymbol{x}_i \in \mathcal{I}^{tr}$, being $f(\pi_{\boldsymbol{x}_i})$ a real-positive valued path-cost function. Two fundamental points are raised on the solving of this problem: finding the appropriate set $\mathcal{P}$ and choosing the path-cost function $f(\cdot)$.

The prototypes are key samples for computing optimum-path trees. Papa et al. [7] proposed selecting the samples located at the boundaries among classes since they are more subject to misclassification, i.e., samples from different classes located close to each other. These samples can be found by computing a Minimum Spanning Tree (MST) over G^{tr}. The MST properties ensure a minimum error during the training phase when all arc-weights are different to each other [15].

Concerning the second point, CG-OPF requires a smooth path-cost function [16] as follows:

$$
\begin{aligned}
f_{max}(\langle \boldsymbol{x}_i \rangle) &= \begin{cases} 0 & \text{if } \boldsymbol{x}_i \in \mathcal{P} \\ +\infty & \text{otherwise,} \end{cases} \\
f_{max}(\pi_{\boldsymbol{x}_i} \cdot (\boldsymbol{x}_i, \boldsymbol{x}_j)) &= \max\{f_{max}(\pi_{\boldsymbol{x}_i}), d(\boldsymbol{x}_i, \boldsymbol{x}_j)\}, \quad (1)
\end{aligned}
$$

in which $\pi_{\boldsymbol{x}_i} \cdot (\boldsymbol{x}_i, \boldsymbol{x}_j)$ stands for the concatenation of path $\pi_{\boldsymbol{x}_i}$ and the arc $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{A}$. Iteratively computing Equation 1 for all sample $\boldsymbol{x}_i \in \mathcal{I}^{tr}$ results in a collection of optimum-path trees rooted at $\mathcal{P}$, i.e., an optimum-path forest. In summary, OPF builds the optimum-path forest by minimizing Equation 1 for every sample $\boldsymbol{x}_i \in \mathcal{I}^{tr}$.

The next step concerns the testing phase, where each sample $\boldsymbol{x}_z \in \mathcal{I}^{ts}$, $z = 1, 2, \ldots, |\mathcal{I}^{ts}|$, is classified individually as follows: $\boldsymbol{x}_z$ is connected to all training nodes from the optimum-path forest learned in the training phase, and it is evaluated the node $\boldsymbol{x}^*_z \in \mathcal{I}^{tr}$ that conquers $\boldsymbol{x}_z$, i.e., the one that satisfies the following equation:

$$
C(\boldsymbol{x}_z) = \min_{\boldsymbol{x}_i \in \mathcal{I}^{tr}} \max\{C(\boldsymbol{x}_i), d(\boldsymbol{x}_i, \boldsymbol{x}_z)\}. \quad (2)
$$

In summary, the classification is performed by finding the training node $\boldsymbol{x}_i$ that minimizes $C(\boldsymbol{x}_z)$ and assigning its label, i.e., y_i, to $\boldsymbol{x}_z$.

[1]For the sake of simplicity, we shall denote $(\boldsymbol{x}_i, y_i)$ by $\boldsymbol{x}_i$ only.

2) OPF with k-nearest neighbors Graph (OPF$_{knn}$): The second supervised OPF classifier model was proposed by Papa et al. [9], which also encodes samples as nodes of a graph. However, connections among nodes are build based on the k-nearest neighbors (k-nn) adjacency relation. A change in the adjacency relation requires a modification in the mentioned fundamental points, i.e., computing $\mathcal{P}$ and choosing the path-cost function $f(\cdot)$. Since a k-nn graph does not guarantee a connected structure, applying an MST is no longer an option to find the prototypes. Instead, kNN-OPF builds the set $\mathcal{P}$ through a region density computation, being the samples located at the regions of highest density elected as prototypes. This strategy is similar to selecting samples nearby the centroids of clusters [17]. Therefore, kNN-OPF can be understood as a dual version of CG-OPF (minimization problem) since it aims at maximizing the cost of each sample, as follows:

$$\max f(\pi_{\boldsymbol{x}_i}),\ \forall\ \boldsymbol{x}_i \in \mathcal{I}^{tr}. \tag{3}$$

Differently, kNN-OPF weights arcs and nodes, and the training and testing graphs are defined as $G^{tr} = (\mathcal{I}^{tr}, \mathcal{A}_k)$, and $G^{ts} = (\mathcal{I}^{ts}, \mathcal{A}_k)$, respectively, where $\mathcal{A}_k$ stands for a k-nearest neighbors adjacency relation. We also define a function $\rho(\cdot)$ that computes the probability density value of a given sample $\boldsymbol{x}_i \in \mathcal{I}^{tr}$ as follows:

$$\rho(\boldsymbol{x}_i) = \frac{1}{\sqrt{2\pi\sigma^2}k} \sum_{\forall \boldsymbol{x}_j \in \mathcal{A}_k(\boldsymbol{x}_i)} \exp\left(\frac{-d(\boldsymbol{x}_i, \boldsymbol{x}_j)}{2\sigma^2}\right), \tag{4}$$

where $\mathcal{A}_k(\boldsymbol{x}_i)$ stands for the k-nearest neighbors of sample $\boldsymbol{x}_i$, and $\sigma = d_{max}/3$, being $d_{max} = \max\{d(\boldsymbol{x}_i, \boldsymbol{x}_j) \in G^{tr}\}$. Using a Gaussian function, we guarantee that 99.7% of the samples within $d(\boldsymbol{x}_i, \boldsymbol{x}_j) \in [0, 3\sigma]$ are included in the computation of $\rho(\boldsymbol{x}_i)$. Moreover, overclustering caused by the asymmetry of $\mathcal{A}_k$ is avoided too.

Once all nodes have their probability density values computed, the competition process among prototypes takes place through the path-cost function f_{min}, given as follows:

$$\begin{aligned} f_{min}(\langle \boldsymbol{x}_i \rangle) &= \begin{cases} \rho(\boldsymbol{x}_i) & \text{if } \boldsymbol{x}_i \in \mathcal{P} \\ \rho(\boldsymbol{x}_i) - 1 & \text{otherwise} \end{cases} \\ f_{min}(\pi_{\boldsymbol{x}_i} \cdot (\boldsymbol{x}_i, \boldsymbol{x}_j)) &= \min\{f_{min}(\pi_{\boldsymbol{x}_i}), \rho(\boldsymbol{x}_j)\}. \end{aligned} \tag{5}$$

The above equation details the two-step competitive process. The upper formulation concerns the first step, which is responsible for computing the initial cost of each sample in $\mathcal{I}^{tr}$ using the probability density function (pdf). Firstly, it is computed the pdf of all samples from the training set, and the values are stored in a *priority queue*. Due to the existence of plateau of densities, it is necessary to verify whether the sample popped out from the priority queue has or not predecessor. If the sample does not have a predecessor (i.e., it is a prototype), its cost will be equal to its initial density value. Otherwise, the density value will be decreased in one unit. Notice that highest-cost samples are removed from the priority queue first.

The lower formulation stands for the propagation of optimum-paths among samples. This process can be understood as a competition among samples that try to conquer each other by offering optimum-cost paths. The cost offered by a sample $\boldsymbol{x}_i$ to conquer a sample $\boldsymbol{x}_j$ is $\min\{f_{min}(\pi_{\boldsymbol{x}_i}), \rho(\boldsymbol{x}_j)\}$. Since the prototype holds the higher cost of its optimum-path tree, the idea is to conquer samples with lower costs. Finally, the sample that maximizes f_{min} for a given sample $\boldsymbol{x}_j$ will be the one to conquer it.

The classification of samples in $\mathcal{I}^{tr}$ is performed similarly to the conquering process. The first step computes the k-nearest neighbors from $\mathcal{I}^{tr}$ to a testing sample $\boldsymbol{x}_z \in \mathcal{I}^{tr}$. Finally, it is verified which node $\boldsymbol{x}_i^* \in \mathcal{I}^{tr}$ satisfies the equation below:

$$C(\boldsymbol{x}_z) = \arg\max_{\boldsymbol{x}_i \in \mathcal{I}^{tr}} \min\{C(\boldsymbol{x}_i), \rho(\boldsymbol{x}_z)\}. \tag{6}$$

Finally, to select suitable values for the parameter $k \in [1, k_{max}]$, say that k^*, we chose the one that maximizes the classification accuracy over a validating set, as further mentioned in the methodology section.

III. Proposed Approach and Methodology

In this section, we introduce the proposed approach for multiple-instance learning using OPF-based classifiers, i.e., MI-OPF$_{cpl}$ and MI-OPF$_{knn}$, which aim at classifying each bag as either positive or negative. The difference between MI-OPF$_{cpl}$ and MI-OPF$_{knn}$ 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 1 depicts an example of a complete graph approach used by OPF$_{cpl}$, 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.

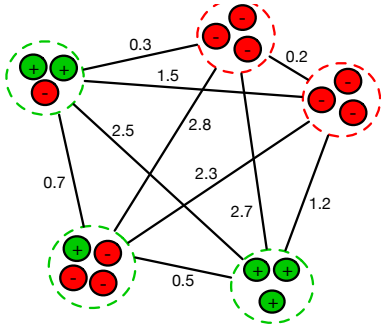

Fig. 1: Complete graph where each node encodes a bag of instances.

As described in Section II-B1, the next step concerning OPF$_{cpl}$ stands for the prototype estimation. To fulfill that purpose, we compute an MST on the complete graph and select

the connected bags with different classes as the prototypes. Figure 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.

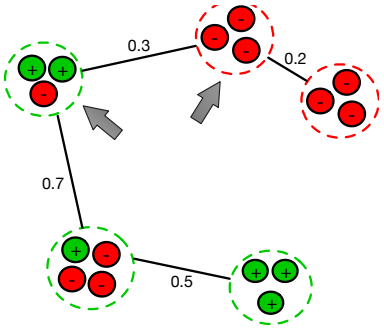

Fig. 2: Minimum Spanning Tree and the prototypes indicated by the gray arrows.

After computing the prototypes, the competition process described in Section II-B1 takes place. As aforementioned, the main idea of OPF$_{cpl}$ is to minimize the cost of each node based on the f_{max} path-cost function (Equation 1). The prototypes are assigned a zero cost, meanwhile a large cost (i.e., ∞) is assigned to all remaining nodes, as depicted in Figure 3.

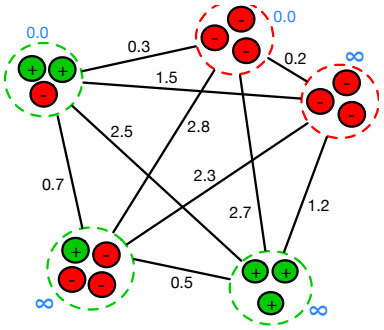

Fig. 3: Complete graph with costs in blue assigned to all bags.

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 4. Notice that the optimum-path forest generated during the training phase (Figure 4) has a close similarity to the shape of the minimum spanning tree (Figure 2) over that same training set. As a matter of fact, such characteristic was considered in the work of Iwashita et al. [18], which proposed a modification of the OPF training algorithm that runs faster than its naïve version.

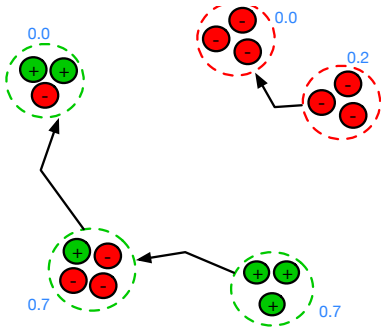

Fig. 4: Optimum-path forest generated during the training phase.

A. 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[2].

a) Image Categorization: The automatic image categorization is comprised of three MI datasets derived from the Corel dataset. Andrews et al. [19] preprocessed and segmented the images through the Blobworld system [20]. 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 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.

b) 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.

c) 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

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

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.

d) 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 [21]. 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 I 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 I: Dataset information.

Datasets	# features (Non-zero)	Positive bags	Negative bags	Positive instances	Negative instances
DWDR	$50(0)$	300	300	997	$5,003$
Elephant	$230(143)$	100	100	762	629
Fox	$230(143)$	100	100	647	673
Tiger	$230(143)$	100	100	544	676
MUSK1	$166(0)$	47	45	207	269
MUSK2	$166(0)$	39	63	$1,017$	$5,581$
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	$1,818$	$1,635$

B. Experimental Setup

The experimental setup of this work follows the same protocol performed by Andrews et al. [19]. 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 OPF$_{cpl}$ and OPF$_{knn}$ 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 OPF$_{cpl}$ and OPF$_{knn}$ 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 OPF$_{knn}$ as 20. The OPF$_{knn}$ 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 ($\mathcal{I}^{pre}$) and an evaluating sets ($\mathcal{I}^{eval}$), such that $\mathcal{I}^{tr} = \mathcal{I}^{pre} \cup \mathcal{I}^{eval}$. Once k^* is found, $\mathcal{I}^{pre}$ and $\mathcal{I}^{eval}$ are merged and a final proper training is performed once more using k^* over $\mathcal{I}^{tr}$. The OPF$_{cpl}$ is parameterless. Regarding the source-codes, we used the OPF implementations from the LibOPF [22].

IV. 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 II–V 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 II. The analysis of accuracy in the original feature space shows that both MI-OPF$_{cpl}$ and MI-OPF$_{knn}$ 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-OPF$_{cpl}$ (i.e., Fox) and MI-OPF$_{knn}$ (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 III also show that compressed representations can be helpful in the classification task. The highest gains are reported by MI-OPF$_{cpl}$ 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 IV 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

TABLE II: Accuracy results on the Image Categorization datasets.

		Elephant	Fox	Tiger
EMDD [23]		78.3	56.1	72.1
mi-SVM [19]	**Linear**	82.2	58.2	78.4
	Polynomial	78.1	55.2	78.1
	RBF	80.0	57.9	78.9
MI-OPF$_{cpl}$	**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-OPF$_{knn}$	**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-OPF$_{cpl}$ **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-OPF$_{knn}$ **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 III: Accuracy results on the MUSK datasets.

		MUSK1	MUSK2
EMDD [23]		84.8	84.9
DD [24]		88.0	84.0
MI-NN [25]		88.9	82.5
IAPR [3]		92.4	89.2
mi-SVM [19]	**RBF**	87.4	83.6
MI-OPF$_{cpl}$	**Original**	76.3	70.5
	PCA-50	85.3	76.4
	PCA-25	85.3	76.5
	PCA-15	81.5	79.0
MI-OPF$_{knn}$	**Original**	79.8	68.3
	PCA-50	82.3	78.6
	PCA-25	83.0	68.1
	PCA-15	82.5	71.4
MI-OPF$_{cpl}$ **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-OPF$_{knn}$ **cosine**	**Original**	84.0	78.3
	PCA-50	86.5	79.7
	PCA-25	82.3	77.9
	PCA-15	83.8	72.1

higher accuracies, especially when the Euclidean distance is applied.

TABLE IV: Accuracy results on the TST datasets.

		TST1	TST2	TST3	TST4	TST7	TST9	TST10
EMDD [23]		85.8	**84.0**	69.0	80.5	75.4	65.5	78.5
mi-SVM [19]	**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-OPF$_{cpl}$	**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-OPF$_{knn}$	**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-OPF$_{cpl}$ **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-OPF$_{knn}$ **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 descriptions 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-OPF$_{cpl}$ and MI-OPF$_{knn}$ 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 V: Accuracy results on the Anomaly Detection dataset.

		DWDR
MI-OPF$_{cpl}$	**Original**	99.0
	PCA-25	99.0
	PCA-15	99.0
MI-OPF$_{knn}$	**Original**	98.3
	PCA-25	98.2
	PCA-15	98.0
MI-OPF$_{cpl}$ **cosine**	**Original**	99.2
	PCA-25	99.0
	PCA-15	99.2
MI-OPF$_{knn}$ **cosine**	**Original**	99.3
	PCA-25	99.5
	PCA-15	98.0

Table VI presents the average training time of both OPF$_{cpl}$ and OPF$_{knn}$ in each scenario. It is worth noting the OPF$_{knn}$ 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-OPF$_{knn}$ that features a fine-tuning parameter step.

V. 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 (OPF$_{cpl}$) and k-nn (OPF$_{knn}$) adjacency relations under different scenarios. The experiments 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-OPF$_{cpl}$ and MI-OPF$_{knn}$ 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

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

Techniques/Datasets		DWDR	Elephant	Fox	Tiger	MUSK1	MUSK2	TST1	TST2	TST3	TST4	TST7	TST9	TST10
MI-OPF$_{cpl}$	**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-OPF$_{knn}$	**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-OPF$_{cpl}$ 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-OPF$_{knn}$ 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

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-OPF$_{cpl}$ and MI-OPF$_{knn}$ 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.

REFERENCES

[1] J. Foulds and E. Frank, "A review of multi-instance learning assumptions," *The Knowledge Engineering Review*, vol. 25, no. 1, pp. 1–25, 2010.

[2] J. D. Keeler, D. E. Rumelhart, and W.-K. Leow, "Integrated segmentation and recognition of hand-printed numerals," 1981.

[3] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Prez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, no. 1, pp. 31 – 71, 1997.

[4] D. Dheeru and E. K. Karra, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[5] G. Quellec, G. Cazuguel, B. Cochener, and M. Lamard, "Multiple-instance learning for medical image and video analysis," *IEEE Reviews in Biomedical Engineering*, vol. 10, pp. 213–234, 2017.

[6] T. Yu, M. Wang, Y. Lv, L. Xue, and J. Liu, "Interpretative topic categorization via deep multiple instance learning," in *2018 International Joint Conference on Neural Networks*, 2018, pp. 1–7.

[7] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki, "Supervised pattern classification based on optimum-path forest," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 120–131, 2009.

[8] J. P. Papa, A. X. Falcão, V. H. C. Albuquerque, and J. M. R. S. Tavares, "Efficient supervised optimum-path forest classification for large datasets," *Pattern Recognition*, vol. 45, no. 1, pp. 512–520, 2012.

[9] J. P. Papa, S. E. N. Fernandes, and A. X. Falcão, "Optimum-path forest based on k-connectivity: Theory and applications," *Pattern Recognition Letters*, vol. 87, pp. 117–126, 2017.

[10] W. P. Amorim, A. X. Falcão, J. P. Papa, and M. H. Carvalho, "Improving semi-supervised learning through optimum connectivity," *Pattern Recognition*, vol. 60, no. Supplement C, pp. 72–85, 2016.

[11] W. P. Amorim, A. X. Falcão, and J. P. Papa, "Multi-label semi-supervised classification through optimum-path forest," *Information Sciences*, vol. 465, pp. 86–104, 2018.

[12] L. M. Rocha, F. A. M. Cappabianco, and A. X. Falcão, "Data clustering as an optimum-path forest problem with applications in image analysis," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 50–68, 2009.

[13] G. J. Sousa, D. C. G. Pedronette, A. Baldassin, P. I. M. Privatto, M. Gaseta, I. R. Guilherme, D. Colombo, L. C. S. Afonso, and J. P. Papa, "Pattern analysis in drilling reports using optimum-path forest," in *2018 International Joint Conference on Neural Networks*, 2018, pp. 1–8.

[14] N. Weidmann, E. Frank, and B. Pfahringer, "A two-level learning method for generalized multi-instance problems," in *Proceedings of the 14th European Conference on Machine Learning*, ser. ECML'03. Berlin, Heidelberg: Springer-Verlag, 2003, pp. 468–479.

[15] C. Allène, J.-Y. Audibert, M. Couprie, and R. Keriven, "Some links between extremum spanning forests, watersheds and min-cuts," *Image and Vision Computing*, vol. 28, no. 10, pp. 1460–1471, 2010, image Analysis and Mathematical Morphology.

[16] A. X. Falcão, J. Stolfi, and R. A. Lotufo, "The image foresting transform: theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, 2004.

[17] G. H. Rosa, K. A. P. Costa, L. A. P. Júnior, J. P. Papa, A. X. Falcão, and J. M. R. S. Tavares, "On the training of artificial neural networks with radial basis function using optimum-path forest clustering," in *22nd International Conference on Pattern Recognition*, 2014, pp. 1472–1477.

[18] A. S. Iwashita, J. P. Papa, A. N. Souza, A. X. Falcão, R. A. Lotufo, V. M. Oliveira, V. H. C. de Albuquerque, and J. M. R. S. Tavares, "A path- and label-cost propagation approach to speedup the training of the optimum-path forest classifier," *Pattern Recognition Letters*, vol. 40, pp. 121–127, 2014.

[19] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Proceedings of the 15th International Conference on Neural Information Processing Systems*, ser. NIPS'02. Cambridge, MA, USA: MIT Press, 2002, pp. 577–584.

[20] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik, "Blobworld: A system for region-based image indexing and retrieval," Berkeley, CA, USA, Tech. Rep., 1999.

[21] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *CoRR*, vol. abs/1607.01759, 2016. [Online]. Available: http://arxiv.org/abs/1607.01759

[22] J. P. Papa, C. T. N. Suzuki, and A. X, "LibOPF: A library for the design of optimum-path forest classifiers."

[23] Q. Zhang and S. A. Goldman, "Em-dd: An improved multiple-instance learning technique," in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2002, pp. 1073–1080.

[24] O. Maron and A. L. Ratan, "Multiple-instance learning for natural scene classification," in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 341–349.

[25] J. Ramon and L. D. Raedt, "Multi instance neural networks," in *Proceedings of the ICML-2000 workshop on attribute-value and relational learning*, 2000.