A Probabilistic Optimum-Path Forest Classifier for Non-Technical Losses Detection

Silas E. N. Fernandes, Danillo R. Pereira, Caio C. O. Ramos, André N. Souza, Danilo S. Gastaldello, and João P. Papa

***Abstract*—Probabilistic-driven classification techniques extend the role of traditional approaches that output labels (usually integer numbers) only. Such techniques are more fruitful when dealing with problems where one is not interested in recognition/identification only, but also into monitoring the behavior of consumers and/or machines, for instance. Therefore, by means of probability estimates, one can take decisions to work better in a number of scenarios. In this paper, we propose a probabilistic-based optimum-path forest (OPF) classifier to handle the problem of non-technical losses (NTL) detection in power distribution systems. The proposed approach is compared against naïve OPF, probabilistic support vector machines, and logistic regression, showing promising results for both NTL identification and in the context of general-purpose applications.**

***Index Terms*—Optimum-path forest, probabilistic classification, non-technical losses.**

I. Introduction

PROBABILISTIC techniques play an important role in machine learning, since they extend the classification process to a greater range than simply labels. Very often we face problems where it is desirable to obtain some probability than just the label itself. Consider the problem of theft identification in energy distribution systems: electrical power companies consider much more fruitful to monitor the probability of a certain user to become a thief along the time instead of purely identifying such user. With the probabilities over time in hands, the company can take some preventive approach, which can be much more cost-effective than just punishing the user.

Manuscript received November 6, 2017; revised February 6, 2018; accepted March 28, 2018. Date of publication April 2, 2018; date of current version April 19, 2019. This work was supported in part by Capes, in part by CNPq under Grant 306166/2014-3 and Grant 307066/2017-7, and in part by FAPESP under Grant 2013/07375-0, Grant 2014/16250-9, Grant 2014/12236-1, Grant 2016/19403-6, and Grant 2017/02286-0. Paper no. TSG-01617-2017. *(Corresponding author: João P. Papa.)*

S. E. N. Fernandes was with the Department of Computing, Federal University of São Carlos, São Carlos 13565, Brazil.

D. R. Pereira is with the Institute of Informatics, University of Western São Paulo, Presidente Prudente 19065, Brazil.

C. C. O. Ramos is with the Department of Electrical Engineering, Catarinense Federal Institute, Rio do Sul 89163-356, Brazil.

A. N. Souza and D. S. Gastaldello are with the Department of Electrical Engineering, São Paulo State University, Bauru 17033-360, Brazil.

J. P. Papa is with the Department of Computing, São Paulo State University, Bauru 17033-360, Brazil (e-mail: papa.joaopaulo@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSG.2018.2821765

A seminal work conducted by Platt [1] extended the well-known Support Vector Machines (SVM) to probabilistic classification. The idea is quite simple: to use SVMs' outputs (labels) to feed a logistic function. Therefore, the initial outputs are mapped within the range $[0, 1]$. However, in order to cope with problems related to different quantities (SVMs' outputs before taking the signal to consider the final label), the author considered to use an optimization process over the whole training set in order to find out variables that regularize the label-probability mapping process. This technique is often referred to as "Platt Scaling".

Despite having a crucial role in the context of theft detection in power distribution systems, probabilistic-based approaches for non-technical losses (NTL) identification seem to not be the main focus of the related literature. As a matter of fact, most works rely on the detection of thefts using abstract-based machine learning algorithms. Ramos *et al.* [2]–[4] presented different approaches based on meta-heuristic techniques for firstly identifying the subset of features that describe better illegal consumers (i.e., characterization) for further recognizing them. Such approaches assume the problem of illegal consumer characterization can be modeled as a feature selection task. Recently, Leite and Mantovani [5] proposed an approach that performs detection and localization of illegal consumers in a single framework.

Non-technical losses detection is an issue often found in the so-called underdeveloped countries, and it mainly consists in the illegal tapping of distribution lines and tampering home meters. Also, NTL strongly affects the economy, since it influences the company's revenues that could be applied to social programs and even enhancing the quality of the service provided by them. In a nutshell, non-technical losses refer to the amount of non-billed and billed electricity that is not paid for. Very recent works have also highlighted the importance of studies in such areas. Viegas *et al.* [6], for instance, surveyed a number of solutions that could be applied to cope with the problem of non-technical losses. The authors also pointed out that the main negative effects of NTL are commonly found in those countries with a fragile economy.

Other recent works also focused on the NTL problem. Han and Xiao [7] proposed a novel fraud detection for smart grids in the context of Colluded Non-Technical Losses, i.e., when many fraudsters act together to cheat the system. The proposed approach makes use of mathematical models that can identify multiple impostors. Lin *et al.* [8] considered the problem of NTL detection using non-cooperative games.

In a nutshell, the main idea is to exploit previous consumption models and to use self-synchronization methods to learn the difference between those models and the real time meter-readings. Then, an approach based on non-cooperative games aims at identifying the abnormalities that are then correlated to the frauds.

Chatterjee *et al.* [9] employed the well-known Long-Short Term Memory networks to analyze the user's consumption over time. The authors claim they can shortlist illegal consumers in real-time. Ahmad [10] considered low-voltage data from commercial, industrial, residential, and agricultural consumer to analyzed and inspected non-technical losses in Pakistan. The work presented a discussion concerning the problem and how we can safeguard from further problems related to frauds in smart meters.

The work by Chatterjee *et al.* [9] considers an import aspect that is related to keep tracking users over time. Therefore, so essential as the identification of illegal consumers is their monitoring over time. By taking into account the identification framework only, one may recognize unlawful consumers only when their behavior is considerably far away from a normal one. In this context, the fraudster may be acting for a considerable time ago, which is quite money-taking regarding industrial consumers, for instance. However, probabilistic-based approaches produce soft outputs that can be analyzed and tracked over time. As an example, if the probabilities for a particular profile have increased considerably, we can decide to take precautions much earlier.

Some years ago, Papa *et al.* [11]–[13] introduced the supervised Optimum-Path Forest (OPF) classifier, which is a graph-based pattern recognition technique that uses a generalization of Dijkstra algorithm for multiple sources and path-cost functions. The OPF classifier has demonstrated interesting results in terms of efficiency and effectiveness, being some of them comparable to the ones obtained by SVMs, but usually faster for training, since OPF is parameterless and its training step has a quadratic complexity [11], [12]. As a matter of fact, Ramos *et al.* [14] and Júnior *et al.* [15] were the first to evaluate the supervised [11], [12] and unsupervised versions [16] of the OPF classifier for NTL detection, respectively. Some years later, Trevizan *et al.* [17] combined Discrete Cosine Transform and the OPF classifier for automatic NTL identification. Later on, Guerrero *et al.* [18] combined artificial neural networks and text mining to reduce non-technical losses in power utilities.

However, the OPF classifier works with abstract (i.e., hard classification) outputs only. Also, as far as we know, there is only one very recent work that considered confidence-based OPF, but not for probability estimates [19], [20]. That work proposed to learn the confidence level (reliability) of each training sample when classifying others. Additionally, the cost-function used for conquering purposes was adapted to consider such reliability level. The authors showed the proposed confidence-based OPF works better in datasets with high concentration of overlapped samples. Therefore, to the best of our knowledge, there is no probabilistic-driven OPF to date, which turns to be the main contribution of this work. The proposed approach is evaluated in the context of general-purpose problems and further for NTL identification in a private dataset obtained from a Brazilian power utility.

The remainder of this paper is organized as follows. Sections II and III present the OPF theoretical background and the probabilistic-driven approach, respectively. Section IV discusses the methodology and presents experiments. Finally, Section V states conclusions and future works.

II. Optimum-Path Forest

The OPF framework is a recent highlight to the development of pattern recognition techniques based on graph partitions. The nodes are the data samples, which are represented by their corresponding feature vectors, and are connected according to some predefined adjacency relation. Although the OPF classifier framework comprises supervised [11]–[13], semi-supervised [21] and unsupervised versions [16], this work focused only on the supervised model, which is used to develop the proposed approach. Currently, there are two distinct versions of the OPF for supervised learning: (i) one that makes use of a complete graph [11], [12], and (ii) another version that uses a graph k-neighborhood [11], [13], [22], [23].

The idea of the OPF classifier is to rule a competition process among some key samples (prototypes). Thus, the algorithm outputs an optimum path forest, which is a collection of optimum-path trees (OPTs) rooted at each prototype. Each OPT defines a discrete optimal partition (cluster) of samples that belong to the very same class, thus labelling the whole dataset with the label of each prototype sample. The next sections describe the OPF working mechanism in more details. This work employs the OPF classifier that use a complete graph, which is parameterless for training phase and is explained below.

A. Training Phase

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

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

Let $f(\pi_s)$ be a path-cost function that essentially assigns a real and positive value to a given path π_s, and $\mathcal{S}$ be a set of prototype nodes. Roughly speaking, OPF aims at minimizing $f(\pi_{\mathbf{s}})$ for every sample in $\mathbf{s} \in \mathcal{D}^{tr}$. The good point is that one does not need to deal with mathematical constraints, and the only rule to solve this problem concerns that all paths must be rooted at $\mathcal{S}$. Therefore, we must choose two principles now: how to compute $\mathcal{S}$ (prototype estimation heuristic) and $f(\pi)$ (path-cost function).

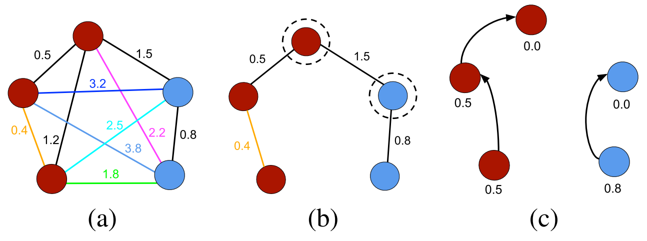

Fig. 1. Illustration of the OPF working mechanism: (a) a two-class (red and blue labels) training graph with weighted arcs, (b) an MST with prototypes highlighted, and (c) optimum-path forest generated during the training phase with costs over the nodes (notice the prototypes have zero cost).

Since prototypes play a major role, Papa *et al.* [11] proposed to place them at the regions with the highest probabilities of misclassification, i.e., at the boundaries among samples from different classes. In fact, we are looking for the nearest samples from different classes, which can be computed by means of a Minimum Spanning Tree (MST) over $\mathcal{G}^{tr}$. The MST has interesting properties, which ensure OPF can be errorless during training when all arc-weights are different to each other [24].

Finally, with respect to the path-cost function, OPF requires f to be a smooth one [11], [12], [25]. Previous experience in image segmentation led the authors to use a chain code-invariant path-cost function, that basically computes the maximum arc-weight along a path, being denoted as f_{max} and given by:

$$f_{max}(\langle s\rangle) = \begin{cases} 0 & \text{if } \mathbf{s} \in \mathcal{S} \\ +\infty & \text{otherwise,} \end{cases}$$
$$f_{max}(\pi_s \cdot (\mathbf{s}, \mathbf{t})) = \max\{f_{max}(\pi_s), d(\mathbf{s}, \mathbf{t})\}, \qquad (1)$$

where $\pi_s \cdot (\mathbf{s}, \mathbf{t})$ stands for the concatenation between path π_s and arc $(\mathbf{s}, \mathbf{t}) \in \mathcal{A}$, and $d(\boldsymbol{s}, \boldsymbol{t})$ stands for the distance between nodes $\boldsymbol{s}$ and $\boldsymbol{t}$. Therefore, $f_{\max}(\pi_{\boldsymbol{s}})$ computes the maximum distance among adjacent samples in $\pi_{\boldsymbol{s}}$, when $\pi_{\boldsymbol{s}}$ is not a trivial path. In short, by computing f_{max} for every sample $\mathbf{s} \in \mathcal{D}^{tr}$, we obtain a collection of OPTs rooted at $\mathcal{S}$, which then originate an optimum-path forest. A sample that belongs to a given OPT means it is more strongly connected to it than to any other in $\mathcal{G}^{tr}$. Roughly speaking, the OPF training step aims at solving Equation (1) in order to build the optimum-path forest. Figure 1 illustrates such training phase, and Algorithm 1 implements the above procedure for the OPF training phase [11], [12].

Lines $1-4$ initialize maps and insert prototypes in Q (the function $\lambda(\cdot)$ in Line 4 assigns the true label to each training sample).[1] The main loop computes an optimum path from $\mathcal{S}$ to every sample $\boldsymbol{s}$ in a non-decreasing order of cost (Lines $5-14$). At each iteration, a path of minimum cost C_s is obtained in P when we remove its last node $\boldsymbol{s}$ from Q (Line 6). Ties are broken in Q using first-in-first-out policy. That is, when two optimum paths reach an ambiguous sample $\boldsymbol{s}$ with the same minimum cost, $\boldsymbol{s}$ is assigned to the first path that reached it. Note that $C_t > C_s$ in Line 8 is false when $\boldsymbol{t}$ has been removed from Q and, therefore, $C_t \neq +\infty$ in Line 11 is true only when

[1]The cost map C stores the optimum-cost of each training sample.

Algorithm 1: Optimum-Path Forest Training Algorithm

```
Input: A labeled training set $\mathcal{D}^{tr}$.
Output: Optimum-path forest $P$, cost map $C$, label map
        $L$, and ordered set $\hat{D}^{tr}$.
Auxiliary: Priority queue $Q$, set $\mathcal{S}$ of prototypes, and
           cost variable cst.

1  Set $\hat{D}^{tr} \leftarrow \emptyset$ and compute the set of prototypes
   $\mathcal{S} \subset \mathcal{D}^{tr}$ by MST ;
2  For each $\boldsymbol{s} \in \mathcal{D}^{tr} \backslash \mathcal{S}$, set $C_{\boldsymbol{s}} \leftarrow +\infty$;
3  for each $\boldsymbol{s} \in \mathcal{S}$ do
4      $C_s \leftarrow 0$, $P_s \leftarrow nil$, $L_s \leftarrow \lambda(\boldsymbol{s})$, and insert $\boldsymbol{s}$ in $Q$;
5  while $Q$ is not empty do
6      Remove from $Q$ a sample $\boldsymbol{s}$ such that $C_s$ is
       minimum;
7      Insert $\boldsymbol{s}$ in $\hat{D}^{tr}$;
8      for each $\boldsymbol{t} \in \mathcal{D}^{tr}$ such that $C_t > C_s$ do
9          Compute cst$\leftarrow \max\{C_s, d(\boldsymbol{s}, \boldsymbol{t})\}$;
10         if $cst < C_t$ then
11             if $C_t \neq +\infty$ then
12                 Remove $\boldsymbol{t}$ from $Q$;
13             $P_t \leftarrow \boldsymbol{s}, L_t \leftarrow L_s, C_t \leftarrow$ cst;
14             Insert $\boldsymbol{t}$ in $Q$;
15 return $[P, C, L, \hat{D}^{tr}]$
```

$\boldsymbol{t} \in Q$. Lines $10-14$ evaluate whether the path that reaches an adjacent node $\boldsymbol{t}$ through $\boldsymbol{s}$ is cheaper than the current path with terminus $\boldsymbol{t}$, and update the position of $\boldsymbol{t}$ in Q, C_t, L_t and P_t accordingly.

B. Classification Phase

The next step concerns the testing phase, where each sample $\mathbf{t} \in \mathcal{D}^{ts}$ is classified individually as follows: $\mathbf{t}$ is connected to all training nodes from the optimum-path forest learned in the training phase (Figure 2a), and it is evaluated the node $\mathbf{v}^* \in \mathcal{D}^{tr}$ that conquers $\mathbf{t}$, which is basically to evaluate the optimum cost C_t as follows [11], [12]:

$$C_{\mathbf{t}} = \arg\min_{\mathbf{v} \in \mathcal{D}^{tr}}\{\max\{C_{\mathbf{v}}, d(\mathbf{v}, \mathbf{t})\}\}. \qquad (2)$$

Let the node $\mathbf{v}^* \in \mathcal{D}^{tr}$ be the one that satisfies Equation (2). The classification step simply assigns the label $\mathbf{t} = \lambda(\mathbf{v}^*)$, as depicted in Figure 2b, where $\lambda(\mathbf{v}^*)$ stands for the true label of sample $\mathbf{v}^*$. Roughly speaking, the testing step aims at finding the training node $\mathbf{v}$ that minimizes $C_{\mathbf{t}}$.

Algorithm 2 implements the OPF classification procedure [11], [12]. The main loop (Lines $1-8$) performs the classification of all nodes in $\mathcal{D}^{ts}$. The inner loop (Lines $3-8$) visits each node $\boldsymbol{k}_{i+1} \in \hat{\mathcal{D}}^{tr}, i = 1, 2, \ldots, |\mathcal{D}_1'| - 1$ until an optimum path $\pi_{\boldsymbol{k}_{i+1}} \cdot \langle \boldsymbol{k}_{i+1}, \boldsymbol{t}\rangle$ is found. In the worst case, the algorithm visits all nodes in $\hat{\mathcal{D}}^{tr}$. Line 4 evaluates $f_{\max}(\pi_{\boldsymbol{k}_{i+1}} \cdot \langle \boldsymbol{k}_{i+1}, \boldsymbol{t}\rangle)$ and Lines $6-7$ update the cost, label and predecessor of $\boldsymbol{t}$.

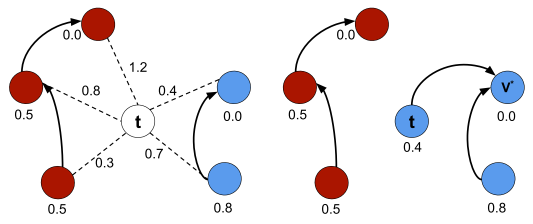

Fig. 2. Illustration of the OPF classification mechanism: (a) sample **t** is connected to all training nodes, and (b) **t** is conquered by $\mathbf{v}^*$, and it receives the "blue" label.

Algorithm 2: Optimum-Path Forest Classification Algorithm

```
Input: Classifier [$P, C, L, \hat{\mathcal{D}}^{tr}$] and a test set $\mathcal{D}^{ts}$ .
Output: Label $L'$, cost $C'$ and predecessor $P'$ maps
        defined for $\mathcal{D}^{ts}$.
Auxiliary: Cost variables tmp and mincost.
1 for each $t \in \mathcal{D}^{ts}$ do
2     $i \leftarrow 0$, mincost $\leftarrow \max\{C_{\boldsymbol{k}_i}, d(\boldsymbol{k}_i, \boldsymbol{t})\}$ $L'_{\boldsymbol{t}} \leftarrow L_{\boldsymbol{k}_i}$,
      $P'_{\boldsymbol{t}} \leftarrow \boldsymbol{k}_i$ and $C'_{\boldsymbol{t}} \leftarrow$mincost;
3     while $i < |\hat{\mathcal{D}}^{tr}|$ and mincost $> C_{\boldsymbol{k}_{i+1}}$ do
4         Compute tmp $\leftarrow \max\{C_{\boldsymbol{k}_{i+1}}, d(\boldsymbol{k}_{i+1}, \boldsymbol{t})\}$;
5         if tmp <mincost then
6             mincost $\leftarrow$ tmp and $C'_{\boldsymbol{t}} \leftarrow$mincost;
7             $L'_{\boldsymbol{t}} \leftarrow L_{\boldsymbol{k}_{i+1}}$ and $P'_{\boldsymbol{t}} \leftarrow \boldsymbol{k}_{i+1}$;
8         $i \leftarrow i + 1$;
9 return [$L', C', P'$]
```

III. Probabilistic Optimum-Path Forest

The probabilistic OPF is inspired in the Platt Scaling approach, which basically ends up mapping the SVMs' output to probability estimates. Therefore, before introducing the proposed approach, one must master the Platt Scaling mechanism.

A. Probabilistic Support Vector Machines

Considering the labeled dataset $\mathcal{D}$ described in Section II, let us assume each sample $\mathbf{x}_i \in \mathcal{D}$ can be assigned to a class label $y_i \in \{-1, +1\}$, $i = 1, 2, \ldots, |\mathcal{D}|$. Platt proposed to approximate the posterior class probability $P(y_i = 1|\mathbf{x}_i)$ as follows [1]:

$$P(y_i = 1|\mathbf{x}_i) \approx P_{A,B}(f_i) = \frac{1}{1 + \exp(Af_i + B)}, \tag{3}$$

where f_i stands for the output (decision function) of SVMs concerning sample $\mathbf{x}_i$. Let $\theta = (A^*, B^*)$ be the best set of parameters that can be determined by the following maximum likelihood problem:

$$\arg\min_{\theta} F(\theta) = -\sum_{i=1}^{m} (y_i \log(p_i) + (1 - y_i)\log(1 - p_i)), \tag{4}$$

where $p_i = P_{A,B}(f_i)$ and m denotes the number of samples to be considered. Essentially, the above equation stands for the cost function of the well-known Logistic Regression classifier.

In order to avoid overfitting, Platt proposed to regularize Equation (4) as follows:

$$\arg\min_{\theta} F(\theta) = -\sum_{i=1}^{m} (t_i \log(p_i) + (1 - t_i)\log(1 - p_i)), \tag{5}$$

where t_i is formulated as follows:

$$t_i = \begin{cases} \frac{N_+ + 1}{N_+ + 2} & \text{if } y_i = +1 \\ \frac{1}{N_- + 2} & \text{if } y_i = -1. \end{cases} \tag{6}$$

In the above formulation, N_+ and N_- stand for the number of positive and negative samples, respectively. In short, t_i can be used to handle unbalanced datasets as well.

B. Modified Probabilistic Support Vector Machines

Almost a decade later the seminal work of Platt, Lin *et al.* [26] highlighted some numerical instabilities related to Equation (5):

- we know that log and exp functions can easily cause an overflow, since $\exp(Af_i + B) \to \infty$ when $Af_i + B$ is large enough. Additionally, $\log(p_i) \to -\infty$ when $p_i \to 0$.
- according to Goldberg [27], $1 - p_i = 1 - \frac{1}{1+\exp(Af_i+B)}$ is a "catastrophic cancellation" when p_i is close to one. Such term arises from the fact we need to subtract two relatively close number that are already results of previous floating-point operations. Lin *et al.* [26] described an interesting example: suppose $f_i = 1$ and $(A, B) = (-64, 0)$. In this case, $1 - p_i$ returns 0, but its equivalent formulation $\frac{\exp(Af_i+B)}{1+\exp(Af_i+B)}$ gives a more accurate result. Also, the very same group of authors stated the aforementioned catastrophic cancellation induces most of the $\log(0)$ occurrences.

In order to deal with the aforementioned situation, Lin *et al.* [26] proposed to reformulate the cost function $F(\theta)$ as follows:

$$F(\theta) = -\sum_{i=1}^{m} (t_i \log(p_i) + (1 - t_i)\log(1 - p_i)) \tag{7}$$

$$= \sum_{i=1}^{m} ((t_i - 1)(q_i) + \log(1 + \exp(q_i))) \tag{8}$$

$$= \sum_{i=1}^{m} (t_i q_i + \log(1 + \exp(-Af_i - B))), \tag{9}$$

where $q_i = Af_i + B$. Therefore, considering the above formulation, $1 - p_i$ and $\log(0)$ do not happen.[2]

However, even if using Equations (8) and (9), the overflow problem may still occur. In order to cope with such problem, Lin *et al.* [26] proposed to apply Equation (9) when $Af_i + B \geq 0$; otherwise, one should use Equation (8).

[2]Please, consider taking a look at the work of Lin *et al.* [26] for a more detailed explanation of the mathematical formulation.

C. Proposed Approach for Probabilistic Outputs

In this section, we present the theoretical background concerning the Probabilistic Optimum-Path Forest (P-OPF). Since the cost assigned to each sample during training and classification with OPF is positive (Equation (2)), we need some minor adjustments with respect to Equation (3), which can be rewritten to accommodate OPF requirements:

$$P\big(\hat{y}_i = y_i|\mathbf{x}_i\big) \approx P_{A,B}(C_i) = \frac{1}{1+\exp\left(Ay_iC_i+B\right)}, \qquad (10)$$

where C_i stands for the cost assigned to sample $\mathbf{x}_i$ during OPF training or classification step, and $\hat{y}_i$ denotes the label predicted by the classifier. Basically, we ended up replacing f_i by y_iC_i, since the cost function C_i is not signed, while $\text{sgn}(f_i) \in \{-,+\}$. The rationale behind the proposed approach is to assume the lower the cost assigned to sample $\mathbf{x}_i$, i.e., C_i, the higher the probability of that sample be correctly classified.

The cost function $F(\theta)$ needs to be reformulated to consider P-OPF model, as follows:

$$\begin{aligned} F(\theta) &= -\sum_{i=1}^{m}(t_i\log(p_i)+(1-t_i)\log(1-p_i)) \\ &= \sum_{i=1}^{m}((t_i-1)(q_i)+\log(1+\exp(q_i))) \\ &= \sum_{i=1}^{m}(t_iq_i+\log(1+\exp(-Ay_iC_i-B))), \qquad (11) \end{aligned}$$

where $p_i = P_{A,B}(C_i)$. Last but not least, one must use the trick proposed by Lin *et al.* [26], i.e., if $Ay_iC_i + B \geq 0$, then one shall use

$$\frac{\exp(-Ay_iC_i-B)}{1+\exp(-Ay_iC_i-B)}, \qquad (12)$$

otherwise one shall use

$$\frac{1}{1+\exp\left(Ay_iC_i+B\right)}. \qquad (13)$$

After learning parameters A and B, we then compute the probability of each sample to belong to class $+1$, i.e., $P(\hat{y}_i = 1|\mathbf{x}_i)$. If $P(\hat{y}_i = 1|\mathbf{x}_i) > P(\hat{y}_i = 0|\mathbf{x}_i)$, then P-OPF assigns the label $+1$ to that sample; otherwise the sample is assigned to class -1. In this work, we adopted $\Theta = 0.5$, since it models a single chance. However, one can easily fine-tune that threshold using a linear-search or any other optimization algorithm. Algorithm 3 implements the probabilistic OPF classifier.

Line 1 executes the OPF training algorithm presented in Section II-A, and it outputs the label, cost and predecessor maps, as well as the ordered set of training samples by the cost map. Line 2 classifies the validating set according to the procedure described in Section II-B, and the loop in Lines $12-25$ are in charge of optimizing parameters A and B, i.e., they aim at computing the best set of parameters θ using a grid search over the ranges $A \in [L_A, U_A]$ and $B \in [L_B, U_B]$, where L_A and U_A stand for the lower and upper bounds considering parameter A. The same can be defined for variable B. Finally, Line 26 stores the best set of parameters θ, which are then used to compute the probability of each test sample in Lines $27-31$.

Algorithm 3: Probabilistic Optimum-Path Forest Algorithm

```
Input: A $\lambda$-labeled training $\mathcal{G}^{tr} = (\mathcal{D}^{tr}, \mathcal{A})$ and validating
       $\mathcal{G}^{vl} = (\mathcal{D}^{vl}, \mathcal{A})$ sets, unlabeled test $\mathcal{G}^{ts} = (\mathcal{D}^{ts}, \mathcal{A})$
       set, and the ranges $[L_A, U_A]$ and $[L_B, U_B]$
       concerning parameters $A$ and $B$, respectively.
Output: Probability output for each sample in $\mathcal{D}^{ts}$.
1  $[P, C, L, \hat{\mathcal{D}}^t r] \leftarrow$ Algorithm 1($\mathcal{D}^{tr}$);
2  $[P', C', L'] \leftarrow$ Algorithm 2($[P, C, L, \hat{\mathcal{D}}^t r], \mathcal{D}^{vl}$);
3  $N_+ \leftarrow 0$ and $N_- \leftarrow 0$;
4  for each $v \in \mathcal{D}^{vl}$ do
5      if $\lambda(v) = +1$ then
6          $N_+ \leftarrow N_+ + 1$;
7      else
8          $N_- \leftarrow N_- + 1$;
9  $t_+ \leftarrow \frac{N_+ + 1}{N_+ + 2}$;
10 $t_- \leftarrow \frac{1}{N_- + 2}$;
11 $A^\star \leftarrow 0$, $B^\star \leftarrow 0$, minF$\leftarrow +\infty$;
12 for each $A_i \in [L_A, U_A], B_j \in [L_B, U_B]$ do
13     for each $v \in \mathcal{D}^{vl}$ do
14         if $\lambda(v) = +1$ then
15             $t \leftarrow t_+$;
16         else
17             $t \leftarrow t_-$;
18         if $A_iC_v + B_j \geq 0$ then
19             $F_{ij} \leftarrow F_{ij} + (t(A_i\lambda(\mathbf{v})C_\mathbf{v} + B_j) + \log(1 +$
               $\exp(-A_i\lambda(\mathbf{v})C_\mathbf{v} - B_j)))$;
20         else
21             $F_{ij} \leftarrow F_{ij} + (t-1) * (A_i\lambda(\mathbf{v})C_\mathbf{v} + B_j) +$
               $\log(1 + \exp(A_i\lambda(\mathbf{v})C_\mathbf{v} + B_j))$;
22     if $F_{ij} < minF$ then
23         minF$\leftarrow F_{ij}$;
24         $A^* \leftarrow A_i$;
25         $B^* \leftarrow B_j$;
26 $\theta \leftarrow (A^\star, B^\star)$;
27 for each $i \in \mathcal{D}^{ts}$ do
28     if $(A^\star\lambda(i)C_i + B^\star) \geq 0$ then
29         $P_\mathbf{i} \leftarrow \frac{\exp(-A^\star\lambda(\mathbf{i})C_\mathbf{i} - B^\star)}{1+\exp(-A^\star\lambda(\mathbf{i})C_\mathbf{i} - B^\star)}$;
30     else
31         $P_\mathbf{i} \leftarrow \frac{1}{1+\exp(A^\star\lambda(\mathbf{i})C_\mathbf{i} + B^\star)}$;
```

IV. Data and Experimental Design

In this section, we present the methodology and experiments employed to validate the effectiveness and efficiency of the proposed approach. Firstly, in Section IV-A, we validate the effectiveness of P-OPF against standard OPF in some benchmark problems. Further, in Section IV-B, we compare the proposed approach in the context of NTL identification.

TABLE I
INFORMATION ABOUT THE BENCHMARKING DATASETS USED IN THIS WORK

Dataset	No. of samples	No. of features	No. of classes
UCI-Adult	$32,561$	123	2
Statlog-Australian	690	14	2
Pima-Indians-Diabetes	768	8	2
UCI-Ionosphere	351	34	2
UCI-Leukemia	72	$7,129$	2
UCI-Liver-disorders	345	6	2
UCI-Madelon	$2,600$	500	2
UCI-Sonar	208	60	2
UCI-Splice	$3,175$	60	2
Synthetic01	$1,000$	2	2
Synthetic02	$1,000$	2	2

TABLE II
MEAN ACCURACY RESULTS CONSIDERING STANDARD OPF, P-OPF AND ITS VARIATIONS UNDER DIFFERENT OPTIMIZATION TECHNIQUES

Dataset	P-OPF-NB	P-OPF-PSO	OPF
UCI-Adult	61.87 ± 2.13	$\mathbf{67.42} \pm 1.02$	65.42 ± 1.14
Statlog-Australian	$\mathbf{78.54} \pm 1.97$	78.35 ± 2.00	78.35 ± 2.00
Pima-Indians-Diabetes	$\mathbf{65.41} \pm 2.44$	$\mathbf{63.36} \pm 8.01$	$\mathbf{65.39} \pm 2.41$
UCI-Ionosphere	78.88 ± 3.78	$\mathbf{78.57} \pm 12.04$	$\mathbf{80.78} \pm 3.68$
UCI-Leukemia	$\mathbf{79.35} \pm 7.84$	$\mathbf{75.73} \pm 16.14$	$\mathbf{79.35} \pm 7.84$
UCI-Liver-disorders	60.37 ± 4.70	60.13 ± 4.90	$\mathbf{60.96} \pm 4.60$
UCI-Madelon	$\mathbf{52.72} \pm 1.38$	$\mathbf{52.72} \pm 1.38$	$\mathbf{52.72} \pm 1.38$
UCI-Sonar	$\mathbf{82.21} \pm 4.33$	$\mathbf{82.21} \pm 4.33$	$\mathbf{82.21} \pm 4.33$
UCI-Splice	54.09 ± 1.78	$\mathbf{55.99} \pm 1.63$	54.42 ± 1.56
Synthetic01	60.52 ± 1.82	$\mathbf{60.99} \pm 1.82$	$\mathbf{60.99} \pm 1.82$
Synthetic02	89.50 ± 1.80	$\mathbf{90.20} \pm 1.54$	$\mathbf{90.20} \pm 1.54$

TABLE III
MEAN *F*-MEASURE VALUES

Dataset	P-OPF-NB	P-OPF-PSO	OPF
UCI-Adult	0.6134 ± 0.0148	$\mathbf{0.6384} \pm 0.0122$	$\mathbf{0.6384} \pm 0.0132$
Statlog-Australian	$\mathbf{0.7860} \pm 0.0194$	0.7840 ± 0.0196	0.7840 ± 0.0196
Pima-Indians-Diabetes	$\mathbf{0.6544} \pm 0.0233$	$\mathbf{0.6311} \pm 0.0896$	$\mathbf{0.6542} \pm 0.0230$
UCI-Ionosphere	0.8078 ± 0.0380	$\mathbf{0.8013} \pm 0.1317$	$\mathbf{0.8257} \pm 0.0361$
UCI-Leukemia	$\mathbf{0.8009} \pm 0.0780$	$\mathbf{0.7621} \pm 0.1734$	$\mathbf{0.8009} \pm 0.0780$
UCI-Liver-disorders	0.6034 ± 0.0476	0.6010 ± 0.0496	$\mathbf{0.6093} \pm 0.0467$
UCI-Madelon	$\mathbf{0.5252} \pm 0.0138$	$\mathbf{0.5252} \pm 0.0138$	$\mathbf{0.5252} \pm 0.0138$
UCI-Sonar	$\mathbf{0.8235} \pm 0.0437$	$\mathbf{0.8235} \pm 0.0437$	$\mathbf{0.8235} \pm 0.0437$
UCI-Splice	0.5354 ± 0.0141	$\mathbf{0.5472} \pm 0.0134$	0.5381 ± 0.0138
Synthetic01	0.6048 ± 0.0186	$\mathbf{0.6097} \pm 0.0183$	$\mathbf{0.6097} \pm 0.0183$
Synthetic02	0.8950 ± 0.0181	$\mathbf{0.9020} \pm 0.0154$	$\mathbf{0.9020} \pm 0.0154$

A. Experiments Over General-Purpose Datasets

In order to fine-tune parameters A and B, we employed two different strategies, being one of them based on meta-heuristics, and another one purely mathematical. In regard to the meta-heuristic-driven techniques, we opted to use Particle Swarm Optimization (PSO) [28], hereinafter denoted as P-OPF-PSO, and with respect to the mathematical approach we used the well-known Newton method with Backtracking line search for solving Equation (4), hereinafter denoted as P-OPF-NB, as described by Lin *et al.* [26]. To study the behavior of P-OPF under different scenarios, we used two synthetic (synhetic01, synhetic02) and nine public benchmarking datasets.[3] These datasets have been frequently used in the evaluation of different classification methods. The datasets were normalized as follows:

$$\mathbf{t}' = \frac{\mathbf{t} - \mu}{\rho}, \tag{14}$$

where μ denotes the mean, and ρ stands for its standard deviation. Also, t and t' correspond to the original and normalized features, respectively. Table I presents the main characteristics of each dataset.

Regarding the methodology, each dataset was partitioned into three subsets, training (40%), validating (20%) and testing sets (40%), hereinafter denoted as 40:20:40. For each range, training, validating, and testing sets were selected randomly and the process was repeated 30 times (cross-validation)[4] It is worth noting the standard OPF was trained over $\mathcal{D}^{tr} \cup \mathcal{D}^{vl}$ considering the aforementioned three stages. After learning, P-OPF-PSO and P-OPF-NB were trained once more using the original training set (i.e., the very same one used by OPF). In order to compare the proposed approach, we computed the mean accuracy and execution time for the further usage of the Wilcoxon signed rank test [29] with significance of 0.05.

Regarding the optimization techniques, we used 500 agents (initial solutions) concerning PSO, $c_1 = 1.4$, $c_2 = 0.6$, $w = 0.5$, as well as 100 iterations for convergence. Variables c_1 and c_2 are used to weight the importance of a possible solution being far or close to the local and global optimum, respectively. Variable w stands for the well-known "inertia weight", which is used as a step size towards better solutions. The search space for $A \times B$ was defined within $[-5, 5] \times [-5, 5]$. Once again, these values have been empirically chosen. With respect to the Newton method, we used a maximum number of iterations of 1000, minimum step in the linear search of $1e - 12$ and $\sigma = 1e - 12$.

Table II presents the mean accuracies and standard deviation over all datasets, being the recognition rates computed according to Papa *et al.* [11], and Table III presents the mean F-measure values concerning the very same group of datasets. The most accurate techniques considering the Wilcoxon test are highlighted in bold. As one can observe, both P-OPF-PSO and standard OPF showed close results in almost all datasets, except "UCI-Adult", which is better generalized by P-OPF using PSO, and "UCI-Liver-disorders", which is better generalized by standard OPF.

The F-measure statistical analysis showed a similar behavior to the ones obtained by standard accuracy, as it can be observed in Table III. Both P-OPF and standard OPF showed close results in 8 out of 11 datasets concerning the accuracy results. Only "UCI-Liver-disorders" dataset can be better generalized by standard OPF. Furthermore, "UCI-Adult" dataset showed a similar result for both P-OPF-PSO and standard OPF regarding the F-measure. In general, P-OPF using PSO was more effective for finding the best set of parameters. Although P-OPF and standard OPF achieved similar results according to accuracy and F-measure, it is worth noting that the P-OPF was able to encode output probability estimates. This idea can be promising in several cases, such as in multiple classifiers that use confidence-oriented techniques instead of abstract outputs.

Table IV presents the mean computational load (in seconds) for the training and validating phases concerning standard OPF

[3]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

[4]Notice the percentages have been empirically chosen, being more intuitive to provide a larger validating set for learning parameters.

TABLE IV
COMPUTATIONAL LOAD (IN SECONDS) CONCERNING STANDARD OPF, P-OPF, AND ITS VARIATIONS UNDER DIFFERENT OPTIMIZATION TECHNIQUES WITH RESPECT TO THE TRAINING TIME (TRAINING AND VALIDATING)

Dataset	P-OPF-NB	P-OPF-PSO	OPF
UCI-Adult	216.14 ± 62.81	243.50 ± 56.63	124.41 ± 34.80
Statlog-Australian	0.0396 ± 0.0012	0.6331 ± 0.0075	0.0234 ± 0.0011
Pima-Indians-Diabetes	0.0418 ± 0.0010	0.6950 ± 0.0066	0.0246 ± 0.0034
UCI-Ionosphere	0.0149 ± 0.0007	0.3169 ± 0.0033	0.0092 ± 0.0005
UCI-Leukemia	0.0414 ± 0.0023	0.1175 ± 0.0044	0.0239 ± 0.0010
UCI-Liver-disorders	0.0092 ± 0.0007	0.3148 ± 0.0100	0.0053 ± 0.0003
UCI-Madelon	3.8750 ± 0.0246	6.1343 ± 0.0343	2.1743 ± 0.0154
UCI-Sonar	0.0075 ± 0.0004	0.1947 ± 0.0035	0.0043 ± 0.0002
UCI-Splice	1.2923 ± 0.0054	3.9445 ± 0.0251	0.7400 ± 0.0034
Synthetic01	0.0605 ± 0.0017	0.9663 ± 0.0802	0.0343 ± 0.0013
Synthetic02	0.0686 ± 0.0010	0.9415 ± 0.0074	0.0408 ± 0.0010

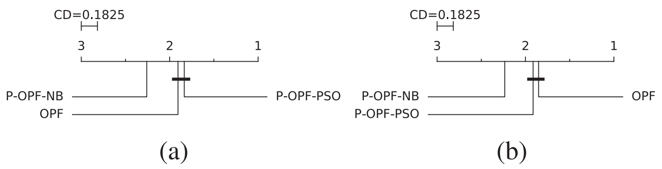

Fig. 3. Comparison between standard OPF and P-OPF with the Nemenyi test concerning the (a) accuracy results and (b) F-measure values. Groups of techniques that are not significantly different (at p=0.05) are connected.

and P-OPF with parameter fine-tuning. Since PSO is a swarm-based technique, which means all possible solutions (agents) are updated at each iteration, its has been consistently more costly than NB.

In order to provide a robust statistical analysis, we performed the non-parametric Friedman test, which is used to rank the algorithms for each dataset separately. In case of Friedman test provides meaningful results to reject the null-hypothesis (h_0: all techniques are equivalent), we can perform a post-hoc test further. For this purpose, we conducted the Nemenyi test, proposed by Nemenyi [30] and described by Demšar [31], which allows us to verify whether there is a critical difference (CD) among techniques or not. The results of the Nemenyi test can be represented in a simple diagram, in which the average ranks of the methods are plotted on the horizontal axis, where the lower the average rank is, the better the technique is. Moreover, the groups with no significant difference are then connected with a horizontal line. In this paper, we employed a significance as of 0.05.

Figure 3 depicts the statistical analysis considering the average accuracy (Figure 3a) and F-measure (Figure 3b) values for all datasets. As one can observe, P-OPF using PSO can be considered the most accurate technique by Nemenyi test regarding to the accuracy results. Such point reflects that P-OPF-PSO technique achieved the best accuracy rates in the majority of datasets. However, the statistical test did not point out a CD between OPF and P-OPF-PSO in the group, which means they performed similarly, thus evidencing the robustness of the proposed probabilistic OPF classifier. On the other hand, regarding the F-measure values, standard OPF can be considered the most accurate one, but the statistical test did not point out a CD between OPF and P-OPF-PSO, i.e., they performed similarly.

TABLE V
MEAN ACCURACY RESULTS CONSIDERING BAYES, LOGISTIC, P-OPF-PSO, AND SVM TECHNIQUES

Dataset	Bayes	Logistic Regression	P-OPF-PSO	SVM
UCI-Adult	53.38 ± 7.60	$\mathbf{76.13} \pm 0.45$	67.42 ± 1.02	75.80 ± 0.64
Statlog-Australian	84.42 ± 1.36	$\mathbf{86.28} \pm 1.13$	78.35 ± 2.00	85.70 ± 1.45
Pima-Indians-Diabetes	70.43 ± 2.31	$\mathbf{71.95} \pm 2.00$	63.36 ± 8.01	$\mathbf{71.39} \pm 2.52$
UCI-Ionosphere	87.53 ± 3.73	83.41 ± 3.18	78.57 ± 12.04	$\mathbf{92.62} \pm 4.15$
UCI-Leukemia	$\mathbf{93.86} \pm 4.16$	$\mathbf{86.30} \pm 18.29$	75.73 ± 16.14	$\mathbf{84.56} \pm 20.96$
UCI-Liver-disorders	57.36 ± 3.34	62.84 ± 6.64	60.13 ± 4.90	$\mathbf{64.97} \pm 4.39$
UCI-Madelon	$\mathbf{59.44} \pm 1.15$	55.73 ± 1.44	52.72 ± 1.38	50.50 ± 2.09
UCI-Sonar	72.80 ± 6.19	75.11 ± 5.09	$\mathbf{82.21} \pm 4.33$	77.04 ± 5.73
UCI-Splice	50.15 ± 0.28	50.00 ± 0.00	$\mathbf{55.99} \pm 1.63$	50.67 ± 0.54
Synthetic01	53.69 ± 2.16	49.03 ± 2.70	$\mathbf{60.99} \pm 1.82$	58.10 ± 4.63
Synthetic02	64.46 ± 6.57	49.22 ± 6.41	$\mathbf{90.20} \pm 1.54$	$\mathbf{90.86} \pm 1.72$

Additionally, we compared the proposed approach using PSO for fine-tuning parameters against probabilistic SVM, the well-known Bayesian classifier and the Logistic Regression,[5] since PSO was more effective than NB in the previous section. The idea is to perform the very same experimental setup but now considering a SVM classifier with Radial Basis Function (RBF) kernel, the Logistic Regression and Bayesian classifier with isotonic calibration.[6] The parameters for SVM[7] and Logistic Regression[8] are fine-tuned through a cross-validation procedure over a validation set using a grid-search. In order to fulfill such purpose, we apply the same procedure, i.e., a 40:20:40 range for training, validating, and testing sets, respectively, repeated 30 times (cross-validation) each. Tables V, VI, and VII present the mean accuracy, F-measure results and Log loss values, respectively. The values in bold stand for the most accurate techniques according to the Wilcoxon signed-rank test.

As one can observe in Table V, the SVM probabilistic, Logistic classifier, and P-OPF showed close results, in which SVM achieved the best results in 2 datasets (i.e., the "UCI-Ionosphere" and "UCI-Liver-disorders") and similar accuracies in 3 out of 11 datasets, while the Logistic Regression achieved the best results in 2 datasets (i.e., the "UCI-Adult" and "Statlog-Australian"), and similar accuracies in 2 out of 11 datasets. Regarding P-OPF, it obtained better results in 3 datasets (i.e., "UCI-Sonar", "UCI-Splice", and "Synthetic01"), and similar accuracies in 1 out of 11 datasets. The F-measure statistical analysis (Table VI) showed a similar results to the ones obtained by standard accuracy, except for "UCI-Liver-disorders " dataset, which was better generalized by SVM and Logistic Regression as well.

We also considered evaluating the techniques using the *log loss* measure L, which is formulated as follows:

$$L = -\frac{1}{m}\sum_{i=1}^{m}(y_i \log(p_i) + (1 - y_i)\log(1 - p_i)). \qquad (15)$$

Notice the above formula stands for the average value of $F(\theta)$ (Equation (4)). Table VII presents the results concerning the *log loss function*. As one can observe, probabilistic SVM showed the best results but being closely followed

[5]We used the SVM probabilistic available in the scikit-learn toolkit [32].

[6]We used the probability calibration available in the scikit-learn library [32].

[7]The RBF parameters were optimized within the ranges $\gamma \in \{0.01, 0.1, 1\}$ and $C \in \{1, 10, 100\}$.

[8]The Logistic Regression parameters were optimized within the ranges *penalty* $= [L1, L2]$, *solver* $= [$*liblinear*, *saga*$]$ and $C \in \{0.01, 0.1, 1, 10, 100\}$.

TABLE VI
MEAN F-MEASURE VALUES CONSIDERING BAYES, LOGISTIC, P-OPF-PSO, AND SVM TECHNIQUES

Dataset	Bayes	Logistic Regression	P-OPF-PSO	SVM
UCI-Adult	0.4556 ± 0.0548	$\mathbf{0.7767} \pm 0.0037$	0.6384 ± 0.0122	0.7747 ± 0.0050
Statlog-Australian	0.8422 ± 0.0134	$\mathbf{0.8600} \pm 0.0106$	0.7840 ± 0.0196	0.8520 ± 0.0143
Pima-Indians-Diabetes	0.7111 ± 0.0225	$\mathbf{0.7289} \pm 0.0207$	0.6311 ± 0.0896	$\mathbf{0.7243} \pm 0.0265$
UCI-Ionosphere	0.8863 ± 0.0340	0.8470 ± 0.0314	0.8013 ± 0.1317	$\mathbf{0.9269} \pm 0.0357$
UCI-Leukemia	$\mathbf{0.9450} \pm 0.0352$	$\mathbf{0.8369} \pm 0.2223$	0.7621 ± 0.1734	$\mathbf{0.8196} \pm 0.2564$
UCI-Liver-disorders	0.5484 ± 0.0570	$\mathbf{0.6069} \pm 0.1101$	0.6010 ± 0.0496	$\mathbf{0.6452} \pm 0.0535$
UCI-Madelon	$\mathbf{0.5908} \pm 0.0114$	0.5572 ± 0.0144	0.5252 ± 0.0138	0.3704 ± 0.0759
UCI-Sonar	0.7255 ± 0.0642	0.7495 ± 0.0512	$\mathbf{0.8235} \pm 0.0437$	0.7694 ± 0.0584
UCI-Splice	0.4628 ± 0.0066	0.4589 ± 0.0000	$\mathbf{0.5472} \pm 0.0134$	0.4746 ± 0.0117
Synthetic01	0.5016 ± 0.0454	0.4235 ± 0.0780	$\mathbf{0.6097} \pm 0.0183$	0.5751 ± 0.0549
Synthetic02	0.6309 ± 0.0835	0.4144 ± 0.0991	$\mathbf{0.9020} \pm 0.0154$	$\mathbf{0.9086} \pm 0.0173$

TABLE VII
MEAN LOG LOSS VALUES CONSIDERING BAYES, LOGISTIC, P-OPF-NB, P-OPF-PSO, AND SVM CLASSIFIERS

Dataset	Bayes	Logistic	P-OPF-NB	P-OPF-PSO	SVM
UCI-Adult	0.504 ± 0.041	$\mathbf{0.327} \pm 0.003$	2.068 ± 0.137	8.222 ± 0.503	0.334 ± 0.007
Statlog-Australian	0.467 ± 0.117	$\mathbf{0.368} \pm 0.032$	1.017 ± 0.105	6.996 ± 1.370	$\mathbf{0.367} \pm 0.032$
Pima-Indians-Diabetes	0.589 ± 0.128	$\mathbf{0.491} \pm 0.023$	1.687 ± 0.144	9.749 ± 2.987	$\mathbf{0.492} \pm 0.024$
UCI-Ionosphere	0.462 ± 0.215	0.594 ± 0.512	0.979 ± 0.228	4.958 ± 1.311	$\mathbf{0.206} \pm 0.091$
UCI-Leukemia	$\mathbf{0.208} \pm 0.217$	$\mathbf{0.295} \pm 0.213$	0.484 ± 0.148	7.265 ± 5.845	$\mathbf{0.273} \pm 0.226$
UCI-Liver-disorders	0.731 ± 0.176	0.631 ± 0.028	1.999 ± 0.441	12.49 ± 2.323	$\mathbf{0.615} \pm 0.027$
UCI-Madelon	$\mathbf{0.665} \pm 0.007$	0.815 ± 0.128	2.740 ± 0.080	16.32 ± 0.477	0.692 ± 0.002
UCI-Sonar	0.627 ± 0.197	0.667 ± 0.318	0.648 ± 0.153	5.989 ± 1.450	$\mathbf{0.478} \pm 0.095$
UCI-Splice	$\mathbf{0.382} \pm 0.020$	0.426 ± 0.000	1.383 ± 0.145	8.335 ± 0.411	0.408 ± 0.009
Synthetic01	0.712 ± 0.046	0.694 ± 0.001	3.456 ± 0.503	4.635 ± 2.405	$\mathbf{0.669} \pm 0.012$
Synthetic02	0.646 ± 0.025	0.694 ± 0.001	0.623 ± 0.236	0.796 ± 0.582	$\mathbf{0.219} \pm 0.026$

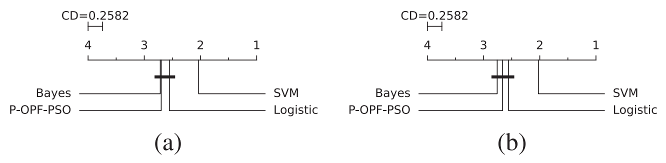

Fig. 4. Comparison concerning P-OPF-PSO against Bayes, Logistic Regression, and SVM concerning the Nemenyi test for: (a) accuracy results and (b) F-measure values.

by P-OPF-NB. The issue concerning P-OPF-PSO for some datasets is related to getting trapped from local optima, i.e., we observed that PSO consistently tried to reach the optimum, but not optimizing both parameters A and B simultaneously (Equation (11)).

In regard to the statistical analysis considering the accuracy results (Figure 4a) and F-measure values (Figure 4b), probabilistic SVM can be considered the most accurate one by Nemenyi test. Furthermore, the statistical test did not point out a CD between the Logistic Regression, P-OPF-PSO, and Bayes, which means they performed similarly.

Table VIII presents the mean computational load (in seconds) for the training and validating phases concerning Bayes, Logistic, P-OPF-PSO, and probabilistic SVM. It is worth noting that the P-OPF and probabilistic SVM have an additional time for training the model due to the calibration of probabilistic predictions according to the Platt scaling. For that purpose, the calibration is performed over a validation set.

Figure 5 shows the statistical analysis considering the computational load for the training time (training and validating) with Nemenyi test. As one can observe, both P-OPF-PSO and SVM are the slowest approaches, since SVM uses the grid-search to fine-tune the hyper-parameters C, γ, followed by the Platt scaling procedure, and P-OPF-PSO is a swarm-based technique, which means an extra computational load to fine-tune parameters A and B. On the average, the fastest approach (Bayesian classifier) required 0.0309 ± 0.0475 seconds for

TABLE VIII
COMPUTATIONAL LOAD (IN SECONDS) CONCERNING BAYES, LOGISTIC, P-OPF-PSO, AND SVM WITH RESPECT TO THE TRAINING TIME (TRAINING AND VALIDATING)

Dataset	Bayes	Logistic	P-OPF-PSO	SVM
UCI-Adult	0.1783 ± 0.0701	72.69 ± 22.85	243.50 ± 56.63	3129.63 ± 3603.51
Statlog-Australian	0.0108 ± 0.0005	0.3121 ± 0.0498	0.6331 ± 0.0075	0.3555 ± 0.1237
Pima-Indians-Diabetes	0.0111 ± 0.0008	0.2457 ± 0.0157	0.6950 ± 0.0066	0.6645 ± 0.3984
UCI-Ionosphere	0.0105 ± 0.0006	0.3110 ± 0.0169	0.3169 ± 0.0033	0.1690 ± 0.0086
UCI-Leukemia	0.0270 ± 0.0023	4.3729 ± 0.5890	0.1175 ± 0.0044	0.3915 ± 0.0228
UCI-Liver-disorders	0.0101 ± 0.0004	0.2261 ± 0.0070	0.3148 ± 0.0100	0.2615 ± 0.1239
UCI-Madelon	0.0425 ± 0.0012	10.46 ± 1.2897	6.1343 ± 0.0343	48.65 ± 67.14
UCI-Sonar	0.0104 ± 0.0005	0.2970 ± 0.0140	0.1947 ± 0.0035	0.1513 ± 0.0048
UCI-Splice	0.0178 ± 0.0011	1.5474 ± 0.3463	3.9445 ± 0.0251	27.56 ± 10.89
Synthetic01	0.0108 ± 0.0007	0.2158 ± 0.0067	0.9663 ± 0.0802	0.4485 ± 0.0766
Synthetic02	0.0108 ± 0.0005	0.2139 ± 0.0055	0.9415 ± 0.0074	0.3187 ± 0.0143

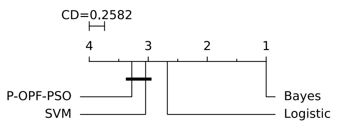

Fig. 5. Comparison concerning P-OPF-PSO, Bayes, Logistic Regression, and SVM with the Nemenyi test considering the computational load for training (training and validating).

training, while probabilistic SVM needed 291.69 ± 897.56 seconds. P-OPF-PSO performed the training and validating phase in 23.43 ± 69.61 seconds, which means it is quite less costly than SVM, mainly due to large datasets employed for comparison purposes (i.e., "UCI-Adult", "UCI-Madelon", and "UCI-Splice"). Note that the P-OPF computational load can be optimized using other meta-heuristic-driven techniques.

B. Experiments for NTL Detection

In this section, we present the experiments results concerning the task of automatic NTL detection. We employed two private datasets from a Brazilian electric utility, being one with 3, 182 profiles of industrial consumers (2, 985 legal and 197 illegal profiles) and the other with 4, 952 profiles of commercial consumers (4, 682 legal and 270 illegal profiles), both represented by eight features:

1) Demand Billed (DB): demand value of the active power considered for billing purposes, in kilowatts (kW);
2) Demand Contracted (DC): the value of the demand for continuous availability requested from the electric utility, in kilowatts (kW);
3) Demand Measured or Maximum Demand (D_{max}): the maximum actual demand for active power, verified by measurement at fifteen-minute intervals during the billing period, in kilowatts (kW);
4) Reactive Energy (RE): energy that flows through the electric and magnetic fields of an AC system, in kilovolt-amperes reactive hours (kVArh);
5) Power Transformer (PT): the power transformer installed for the consumers, in kilovolt-amperes (kVA);
6) Power Factor (PF): the ratio between the consumed active and apparent power in a circuit. The PF indicates the efficiency of a power distribution system;
7) Installed Power (P_{inst}): the sum of the nominal power of all electrical equipment installed and ready to operate at the consumer unit, in kilowatts (kW);

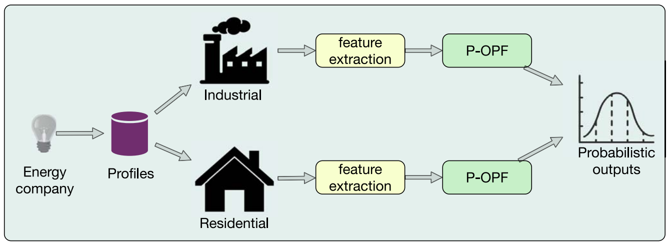

Fig. 6. Pipeline used to validate the proposed approach in the context of NTL identification.

TABLE IX
MEAN ACCURACY RESULTS OF BAYES, LOGISTIC, P-OPF, OPF AND PROBABILISTIC SVM CONCERNING THE TASK OF NTL DETECTION

Dataset	Bayes	Logistic	P-OPF-NB	P-OPF-PSO	OPF	SVM
Commercial	52.28 ± 1.26	50.20 ± 0.47	77.45 ± 3.48	$\mathbf{83.41} \pm 2.34$	$\mathbf{83.41} \pm 2.34$	56.02 ± 7.35
Industrial	52.38 ± 1.30	50.36 ± 0.80	73.68 ± 3.57	80.78 ± 2.71	$\mathbf{80.80} \pm 2.71$	59.22 ± 7.21

TABLE X
MEAN F-MEASURE VALUES OF BAYES, LOGISTIC, P-OPF, OPF AND PROBABILISTIC SVM CONCERNING THE TASK OF NTL DETECTION

Dataset	Bayes	Logistic	P-OPF-NB	P-OPF-PSO	OPF	SVM
Commercial	0.530 ± 0.023	0.493 ± 0.007	0.788 ± 0.028	$\mathbf{0.829} \pm 0.018$	$\mathbf{0.829} \pm 0.018$	0.577 ± 0.096
Industrial	0.529 ± 0.024	0.489 ± 0.017	0.758 ± 0.026	$\mathbf{0.800} \pm 0.019$	$\mathbf{0.801} \pm 0.018$	0.625 ± 0.104

TABLE XI
MEAN LOG LOSS VALUES OF BAYES, LOGISTIC, P-OPF AND PROBABILISTIC SVM CONCERNING THE TASK OF NTL DETECTION

Dataset	Bayes	Logistic	P-OPF-NB	P-OPF-PSO	SVM
Commercial	0.206 ± 0.013	0.214 ± 0.010	0.426 ± 0.062	0.829 ± 0.286	$\mathbf{0.183} \pm 0.033$
Industrial	0.222 ± 0.025	0.237 ± 0.018	0.422 ± 0.085	0.976 ± 0.340	$\mathbf{0.187} \pm 0.036$

8) Load Factor (LF): the ratio between the average demand ($D_{average}$) and maximum demand (D_{max}) of the consumer unit. The LF is an index that shows how the electric energy is used in a rational way.

At every 15 minutes, the electric utility recorded consumption data during one year. After that, such technical data was used to compute the aforementioned monthly features for both datasets.

Figure 6 depicts the pipeline adopted in this work to validate the proposed probabilistic OPF in the context of automatic non-technical losses identification. Given the profiles collected by the energy company accordingly to their category (i.e., industrial or residential), we can perform the feature extraction step that aims at compiling the above eight features from each consumer. Further, these features will be used as input to P-OPF to learn the correct mapping between a particular profile and its situation in the system, i.e., "legal" or "illegal" consumer. The outcome of the pipeline is a probability of that profile being a fraudster.

Tables IX and X present the results concerning the task of NTL detection with respect to the accuracy and F-measure values, respectively. Clearly, one can observe that standard OPF and P-OPF-PSO obtained the best results considering the accuracy and F-measure values, and outperformed the other classifiers considerably. However, with respect to the Log loss values, the SVM probabilistic has been a more confidence technique, as one can observe in Table XI.

From such results, one can draw some interesting conclusions: firstly, probabilistic OPF can provide results that are similar to is standard version, but with the benefit of soft (i.e., probabilistic) outputs. Such skill plays an important role in the context of NTL identification, where the probability of being an illegal consumer is sometimes more interesting than just labeling it as a thief or not. Secondly, probabilistic OPF outperformed SVM by far if we consider the accuracy and F-measure, thus showing to be robust enough to be compared to state-of-the-art techniques.

V. CONCLUSION AND FUTURE WORKS

Probabilistic classification has been a topic of great interest concerning the machine learning community, mainly due to the lack of a more "flexible" information rather than labels only. In this work, we cope with this problem by proposing a probabilistic OPF for NTL detection, namely P-OPF. The results of the proposed P-OPF were compared against standard OPF, probabilistic SVM, the well-known Bayesian classifier, and the Logistic Regression in a number of datasets, achieving suitable results in several of them.

In regard to future works, we aim at extending P-OPF for multi-class classification problems, as well as to consider other optimization techniques to fine-tune the new parameters that help minimizing the cost function.

ACRONYMS

SVM	Support Vector Machines
NTL	Non-Technical Losses
OPF	Optimum-Path Forest
OPT	Optimum-Path Tree
P-OPF	Probabilistic Optimum-Path Forest
PSO	Particle Swarm Optimization
NB	Newton Backtracking optimizer
P-OPF-PSO	P-OPF optimized with PSO
P-OPF-NB	P-OPF optimized with NB

REFERENCES

[1] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. Cambridge, MA, USA: MIT Press, 1999, pp. 61–74.

[2] C. C. O. Ramos, A. N. de Souza, A. X. Falcão, and J. P. Papa, "New insights on nontechnical losses characterization through evolutionary-based feature selection," *IEEE Trans. Power Del.*, vol. 27, no. 1, pp. 140–146, Jan. 2012.

[3] D. R. Pereira *et al.*, "Social-spider optimization-based support vector machines applied for energy theft detection," *Comput. Elect. Eng.*, vol. 49, pp. 25–38, Jan. 2016.

[4] C. C. O. Ramos, D. Rodrigues, A. N. de Souza, and J. P. Papa, "On the study of commercial losses in Brazil: A binary black hole algorithm for theft characterization," *IEEE Trans. Smart Grid*, vol. 9, no. 2, pp. 676–683, Mar. 2018.

[5] J. B. Leite and J. R. S. Mantovani, "Detecting and locating non-technical losses in modern distribution networks," *IEEE Trans. Smart Grid*, vol. 9, no. 2, pp. 1023–1032, Mar. 2018.

[6] J. L. Viegas, P. R. Esteves, R. Melício, V. M. F. Mendes, and S. M. Vieira, "Solutions for detection of non-technical losses in the electricity grid: A review," *Renew. Sustain. Energy Rev.*, vol. 80, pp. 1256–1268, Dec. 2017.

[7] W. Han and Y. Xiao, "A novel detector to detect colluded non-technical loss frauds in smart grid," *Comput. Netw.*, vol. 117, pp. 19–31, Apr. 2017.

[8] C.-H. Lin, S.-J. Chen, C.-L. Kuo, and J.-L. Chen, "Non-cooperative game model applied to an advanced metering infrastructure for non-technical loss screening in micro-distribution systems," *IEEE Trans. Smart Grid*, vol. 5, no. 5, pp. 2468–2469, Sep. 2014.

[9] S. Chatterjee *et al.*, "Detection of non-technical losses using advanced metering infrastructure and deep recurrent neural networks," in *Proc. IEEE Int. Conf. Environ. Elect. Eng. IEEE Ind. Commercial Power Syst. Europe*, 2017, pp. 1–6.

[10] T. Ahmad, "Non-technical loss analysis and prevention using smart meters," *Renew. Sustain. Energy Rev.*, vol. 72, pp. 573–589, May 2017.

[11] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki, "Supervised pattern classification based on optimum-path forest," *Int. J. Imag. Syst. Technol.*, vol. 19, no. 2, pp. 120–131, 2009.

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

[13] J. P. Papa, S. E. N. Fernandes, and A. X. Falcão, "Optimum-path forest based on *k*-connectivity: Theory and applications," *Pattern Recogn. Lett.*, vol. 87, pp. 117–126, Feb. 2016.

[14] C. C. O. Ramos, A. N. de Sousa, J. P. Papa, and A. X. Falcao, "A new approach for nontechnical losses detection based on optimum-path forest," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 181–189, Feb. 2011.

[15] L. A. P. Júnior *et al.*, "Unsupervised non-technical losses identification through optimum-path forest," *Elect. Power Syst. Res.*, vol. 140, pp. 413–423, Nov. 2016.

[16] L. M. Rocha, F. A. M. Cappabianco, and A. X. Falcão, "Data clustering as an optimum-path forest problem with applications in image analysis," *Int. J. Imag. Syst. Technol.*, vol. 19, no. 2, pp. 50–68, 2009.

[17] R. D. Trevizan, A. S. Bretas, and A. Rossoni, "Nontechnical losses detection: A discrete cosine transform and optimum-path forest based approach," in *Proc. North Amer. Power Symp. (NAPS)*, 2015, pp. 1–6.

[18] J. I. Guerrero *et al.*, "Non-technical losses reduction by improving the inspections accuracy in a power utility," *IEEE Trans. Power Syst.*, vol. 33, no. 2, pp. 1209–1218, Mar. 2018.

[19] S. E. N. Fernandes, W. Scheirer, D. D. Cox, and J. P. Papa, "Improving optimum-path forest classification using confidence measures," in *Proc. 20th Iberoamer. Congr. Progr. Pattern Recognit. Image Anal. Comput. Vis. Appl. (CIARP)*, 2015, pp. 619–625.

[20] S. E. N. Fernandes and J. P. Papa, "Improving optimum-path forest learning using bag-of-classifiers and confidence measures," *Pattern Anal. Appl.*, to be published.

[21] W. P. Amorim, A. X. Falcão, J. P. Papa, and M. H. Carvalho, "Improving semi-supervised learning through optimum connectivity," *Pattern Recognit.*, vol. 60, pp. 72–85, Dec. 2016.

[22] J. P. Papa and A. X. Falcão, "A new variant of the optimum-path forest classifier," in *Advances in Visual Computing* (LNCS 5358), G. Bebis *et al.*, Eds. Heidelberg, Germany: Springer, 2008, pp. 935–944.

[23] J. P. Papa and A. X. Falcão, "A learning algorithm for the optimum-path forest classifier," in *Graph-Based Representations in Pattern Recognition* (LNCS 5534), A. Torsello, F. Escolano, and L. Brun, Eds. Heidelberg, Germany: Springer, 2009, pp. 195–204.

[24] C. Allène, J.-Y. Audibert, M. Couprie, and R. Keriven, "Some links between extremum spanning forests, watersheds and min-cuts," *Image Vis. Comput.*, vol. 28, no. 10, pp. 1460–1471, 2010.

[25] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 19–29, Jan. 2004.

[26] H.-T. Lin, C.-J. Lin, and R. C. Weng, "A note on Platt's probabilistic outputs for support vector machines," *Mach. Learn.*, vol. 68, no. 3, pp. 267–276, 2007.

[27] D. Goldberg, "What every computer scientist should know about floating-point arithmetic," *ACM Comput. Surveys*, vol. 23, no. 1, pp. 5–48, 1991.

[28] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*. San Francisco, CA, USA: Morgan Kaufmann, 2001.

[29] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.

[30] P. Nemenyi, *Distribution-Free Multiple Comparisons*, Princeton Univ., Princeton, NJ, USA, 1963.

[31] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[32] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Jan. 2011.

Silas E. N. Fernandes, photograph and biography not available at the time of publication.

Danillo R. Pereira, photograph and biography not available at the time of publication.

Caio C. O. Ramos, photograph and biography not available at the time of publication.

André N. Souza, photograph and biography not available at the time of publication.

Danilo S. Gastaldello, photograph and biography not available at the time of publication.

João P. Papa, photograph and biography not available at the time of publication.