Computers and Electrical Engineering 37 (2011) 886–894

Contents lists available at SciVerse ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

A novel algorithm for feature selection using Harmony Search and its application for non-technical losses detection [☆]

Caio C.O. Ramos [a,*], André N. Souza [a,b], Giovani Chiachia [c], Alexandre X. Falcão [c], João P. Papa [d]

[a] *Department of Electrical Engineering, University of São Paulo, São Paulo, Brazil*
[b] *Department of Electrical Engineering, UNESP – Univ Estadual Paulista, Bauru, Brazil*
[c] *Institute of Computing, University of Campinas, Campinas, Brazil*
[d] *Department of Computing, UNESP – Univ Estadual Paulista, Bauru, Brazil*

ARTICLE INFO

Article history:
Received 26 January 2011
Received in revised form 24 September 2011
Accepted 26 September 2011
Available online 22 October 2011

ABSTRACT

Finding an optimal subset of features that maximizes classification accuracy is still an open problem. In this paper, we exploit the speed of the Harmony Search algorithm and the Optimum-Path Forest classifier in order to propose a new fast and accurate approach for feature selection. Comparisons to some other pattern recognition and feature selection techniques showed that the proposed hybrid algorithm for feature selection outperformed them. The experiments were carried out in the context of identifying non-technical losses in power distribution systems.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

In recent years, the problem of detecting non-technical losses in distribution systems, which refers to the delivered but not billed energy, has been paramount [1,2]. Theft and adulteration of power meters, with the purpose to modify the measurement of the energy consumption, are the main causes that lead to non-technical losses in power companies. Since that to perform periodic inspections to minimize such frauds may be very expensive, it is a hard task to calculate or measure the amount of losses.

Aimed at reducing fraud and energy theft, several electric power companies have been concerned that the illegal connections should be better profiled. The minimization of such losses may guarantee investments in energy quality programs, as well as it may enable a reduction in its price to the consumer.

Recently, some advances in this area were made with the application of artificial intelligence techniques in order to automatically identify non-technical losses. Nagi et al. [3], for instance, used Support Vector Machines (SVMs) aiming to accomplish this task. A hybrid approach between Genetic Algorithms (GA) and SVMs for detecting non-technical losses was also addressed in [4]. Nizar et al. [5] applied data mining-based techniques for non-technical loss analysis, and Monedero et al. [6] proposed to use Artificial Neural Networks (ANNs) together with statistical analysis for fraud detection in electrical consumption. Ramos et al. [7] introduced the Optimum-Path Forest (OPF) classifier in this context, and also showed its advantages over the aforementioned approaches.

Although a great part of the pattern recognition community has studied new machine learning techniques in order to benefit a wide range of applications, another part has designed algorithms for feature selection aimed at dimensionality reductions in the feature space. By selecting the most discriminating features, one may improve both the classifier accuracy and efficiency, which is very important in general and even more important for large datasets.

[☆] Reviews processed and approved for publication to the Editor-in-Chief Dr. Manu Malek.
[*] Corresponding author.
E-mail addresses: caioramos@gmail.com (C.C.O. Ramos), papa@fc.unesp.br (J.P. Papa).

0045-7906/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compeleceng.2011.09.013

Given that feature selection can be modeled as an optimization problem, several works have addressed this task by using evolutionary techniques, which can find near-optimal solutions with a reasonable computational effort. Firpi and Goodman [8] proposed a Particle Swarm Optimization (PSO) algorithm defined in the binary space to select the most representative features, validating their approach for remote sensing applications. The employment of GA-based algorithms for this purpose has also been investigated [9]. Recently, Diao and Shen [10] presented an approach based on Harmony Search to handle the same problem.

In this paper, we exploit the OPF speed to propose a new hybrid feature selection algorithm based on Harmony Search and OPF (HS–OPF), and we validate it in the context of automatic recognition of non-technical losses in power distribution systems. The aim is to characterize and to detect possible illegal consumers by means of a fast methodology to perform this task. As far as we know, we are the first to combine OPF with HS in order to select most representative features. The experiments are conducted in two rounds: at first we compare OPF to several other supervised classifiers, and secondly we compare HS–OPF to a PSO-based algorithm, which also uses OPF for feature selection (PSO–OPF) [11]. Both HS–OPF and PSO–OPF are applied in the original feature space, as well as in two other features spaces derived from the original one by Principal Component Analysis (PCA) and Kernel Principal Component Analysis (KPCA).

The remainder of this paper is organized as follows. Section 2 briefly introduces OPF theory, and Sections 3 and 4 discuss, respectively, the evolutionary intelligence background and the proposed algorithm for feature selection based on HS and OPF. Section 5 presents the experiments and, finally, Section 6 states conclusions and future works.

2. Optimum-Path Forest classifier

Given a training set with samples from distinct classes, we wish to design a pattern classifier which can assign the true class label to any new sample. Each sample is represented by a set of features and a distance function measures their dissimilarity in the feature space. In the OPF context, the training samples are then interpreted as the nodes of a graph, whose arcs are defined by a given adjacency relation and weighted by the distance function. It is expected that samples from a same class/cluster are connected by a path of nearby samples. Therefore, the degree of connectedness for any given path is measured by a connectivity (path-value) function, which exploits the distances along the path. In supervised learning, the true label of the training samples is known and so it is exploited to identify key samples (prototypes) in each class. Optimum paths are computed from the prototypes for each training sample, so that each prototype becomes a root for an Optimum-Path tree composed by its most strongly connected samples. The labels of these samples are assumed to be the same as their root. In unsupervised learning, each cluster is represented by an Optimum-Path tree rooted at a single prototype but we do not know the class label for the training samples. Therefore, we expect that each cluster contains only samples of a same class and some other information about the application is needed to complete classification. The basic idea is then to specify an adjacency relation and a path-value function, compute prototypes and reduce the problem into an Optimum-Path Forest computation in the underlying graph. The training forest becomes a classifier which can assign the label of its most strongly connected root to any new sample. Essentially, this methodology extends a previous approach, called Image Foresting Transform [12], for designing image processing operators from the image domain to the feature space.

Papa et al. [13] presented a first method for supervised classification using a complete graph (implicit representation) and maximum arc weight along a path as a connectivity function. The prototypes were chosen as samples that share an arc between distinct classes in a minimum spanning tree of the training set. This OPF classifier has been widely used in several applications, such as remote sensing, pathology detection by means of biomedical signal recognition, emotion recognition through speech processing, automatic vowel classification, biometrics, petroleum well drilling monitoring, medical image segmentation, and robust object tracking. Although we have different variations of the Optimum-Path Forest-based classifier [14,15], in this work we focus on the initial version of OPF, i.e., the one that employs the complete graph, mainly because of its simplicity and speed.

3. Evolutionary intelligence background

In this section we present some theoretical background about the evolutionary intelligence techniques used in this work.

3.1. Harmony Search

Harmony Search (HS) is an evolutionary algorithm inspired in the improvisation process of music players [16]. HS is simple in concept, has few parameters, and it is easy to implement, with theoretical background for a stochastic derivative. The main idea is to use the same process adopted by musicians to create new songs to obtain a near-optimal solution for some optimization process. Basically, any possible solution is modeled as a harmony and each parameter to be optimized can be seen as a musical note. The best harmony (solution) is chosen as the one that maximizes some optimization criteria. The algorithm is composed by few steps, as described below:

- Step 1: Initialize the optimization problem and algorithm parameters.
- Step 2: Initialize a Harmony Memory (HM).

- Step 3: Improvise a new harmony from HM.
- Step 4: Update the HM if the new harmony is better than the worst harmony in the HM, include the new harmony in HM, and remove the worst one from HM, and
- Step 5: If the stopping criterion is not satisfied, go to Step 3.

Below, we discuss each one of the aforementioned steps.

3.1.1. The optimization problem and algorithm parameters

In order to describe how HS works, an optimization problem is specified in Step 1 as follows:

$$\max f(x) \quad \text{subjectto } x^j \in X_j, \forall j = 1,2,\ldots,N, \tag{1}$$

where $f(x)$ is the objective function, x^j and X_j, mean, respectively, the design variable j and its set of possible values, and N is the number of design variables. Notice that $X_j \in \{0,1\}$ in the case of feature selection, and this extends to all variables $j = \{1,2,\ldots,N\}$. Thus, in this case, we can generalize X_j to X.

The HS parameters required to solve the optimization problem (Eq. 1) are also specified in this step. They are: the Harmony Memory Size (HMS), the Harmony Memory Considering Rate (HMCR), the Pitch Adjusting Rate (PAR), and the stopping criterion. HMCR and PAR are parameters used to improve the solution vector, i.e., they can help the algorithm to find globally and locally improved solutions in the Harmony Search process (Step 3).

3.1.2. Harmony Memory (HM)

In Step 2, the HM matrix (Eq. 2) is initialized with randomly generated solution vectors with their respective values for the objective function:

$$\text{HM} = \left[\begin{array}{cccc|c} x_1^1 & x_1^2 & \ldots & x_1^N & f(x_1) \\ x_2^1 & x_2^2 & \ldots & x_2^N & f(x_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{\text{HMS}}^1 & x_{\text{HMS}}^2 & \ldots & x_{\text{HMS}}^N & f(x_{\text{HMS}}) \end{array}\right], \tag{2}$$

where x_i^j denotes the decision variable j from harmony i.

3.1.3. Generating a new harmony from HM

In Step 3, a new harmony vector $\hat{x} = (\hat{x}^1, \hat{x}^2, \ldots, \hat{x}^N)$ is generated from the HM based on memory considerations, pitch adjustments, and randomization (music improvisation). It is also possible to choose the new value using the HMCR parameter, which varies between 0 and 1 as follows:

$$\hat{x}^j \leftarrow \begin{cases} \hat{x}^j \in \left\{x_1^j, x_2^j, \ldots, x_{\text{HMS}}^j\right\} & \text{with probability HMCR,} \\ \hat{x}^j \in X & \text{with probability (1-HMCR).} \end{cases} \tag{3}$$

The HMCR is the probability of choosing one value from the historic values stored in the HM, and (1-HMCR) is the probability of randomly choosing one feasible value not limited to those stored in the HM.

Further, every component j of the new harmony vector $\hat{x}$ is examined to determine whether it should be pitch-adjusted:

$$\text{Pitching adjusting decision for } \hat{x}^j \leftarrow \begin{cases} \textbf{Yes} \text{ with probability PAR,} \\ \textbf{No} \text{ with probability (1-PAR).} \end{cases} \tag{4}$$

The pitch adjustment for each instrument is often used to improve solutions and to escape from local optima. This mechanism concerns shifting the neighboring values of some decision variable in the harmony. If the pitch adjustment decision for the decision variable $\hat{x}^j$ is Yes, $\hat{x}^j$ is replaced as follows:

$$\hat{x}^j \leftarrow \hat{x}^j + rb, \tag{5}$$

where b is an arbitrary distance bandwidth for the continuous design variable, and r is a uniform distribution between 0 and 1. In case of binary optimization problems, this computation is not used.

3.1.4. Update HM

In Step 4, if the new harmony vector is better than the worst harmony in the HM, the latter is replaced by this new harmony.

3.1.5. Stopping criterion

In Step 5, the HS algorithm finishes when it satisfies the stopping criterion. Otherwise, Steps 3 and 4 are repeated in order to improvise a new harmony again.

3.2. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an algorithm modeled on swarm intelligence that finds a solution in a search space based on social behavior dynamics [17]. Each possible solution to the problem is modeled as a particle in the swarm that imitates its neighborhood based on the values of the fitness function found so far.

Other definitions consider PSO as a stochastic and population-based search algorithm, in which social behavior learning allows each possible solution (particle) to "fly" into this space (swarm) looking for other particles that have better characteristics, i.e., the ones that maximize a fitness function. Each particle has a memory that stores its best local solution (local maxima) and the best global solution (global maxima). Thus, taking this information into account, each particle has the ability to imitate the others that give to it the best local and global maxima. This process simulates social interaction between humans looking for the same objective or bird flocks looking for food, for instance. This socio-cognitive mechanism can be summarized into three main principles [17]: (i) evaluating, (ii) comparing and (iii) imitating. Each particle can evaluate others in its neighborhood through some fitness function, can compare it with its own value and, finally, can decide whether it is a good choice to imitate them.

The entire are also known. After defining the swarm space $\Re^N$, in which each particle $p_i = (\vec{x}_i, \vec{v}_i) \in \Re^N$ has two main features: (i) position ($\vec{x}_i$) and (ii) velocity ($\vec{v}_i$). The local (best current position $\widehat{x}_i$) and global solution $\widehat{s}$ are also known. After defining the swarm size, i.e., the number of particles, each one of them is initialized with random values for both velocity and position. Each individual is then evaluated with respect to some fitness function and its local maximum is updated. At the end, the global maximum is updated with the particle that achieved the best position in the swarm. This process is repeated until some convergence criterion is reached. The updated position and velocity equations of particle p_i, in the simplest form that governs the PSO, are, respectively, given by

$$\vec{v}_i = w\vec{v}_i + c_1 r_1(\widehat{x}_i - \vec{x}_i) + c_2 r_2(\widehat{s} - \vec{x}_i) \tag{6}$$

and

$$\vec{x}_i = \vec{x}_i + \vec{v}_i, \tag{7}$$

where w is the inertia weight that controls the power of the interactions between the particles, and $r_1, r_2 \in [0,1]$ are random variables that give the idea of stochasticity to the PSO method. Constants c_1 and c_2 are used to guide particles into good directions.

3.2.1. Binary PSO

Kennedy and Eberhart [18] proposed a binary version of PSO, i.e., BPSO, in which the PSO algorithm was modified in order to deal with binary optimization problems. Actually, the only modification was made in the particle's position updating equation, since continuous values are not allowed here. Thus, Eq. 7 can be rewritten as:

$$x_i^j = \begin{cases} 1 & \text{if } \psi < \frac{1}{1+e^{-v_i^j}} \\ 0 & \text{otherwise} \end{cases}, \tag{8}$$

where ψ is a random factor in the [0,1] interval and x_i^j and v_i^j denote, respectively, the values of position and velocity of particle i at position j, $j = 1,2,\ldots,N$.

Further, Firpi and Goodman [19] extended BPSO to the context of feature selection, since the idea is quite simple: to model each possible solution as a particle in the swarm. The entire swarm dynamics is guided by some classifier's recognition rate as the fitness function. Therefore, the subset of features that maximizes the recognition rate will be the one chosen by BPSO. Basically, one can use a training and an evaluating set in order to conduct any evolutionary-based algorithm for feature selection: firstly, the classifier is trained in the training set, and then it is evaluated in the evaluating set by classifying its samples. Thus, the accuracy on this set is used as the fitness function to guide the optimization algorithm.

4. Proposed HS–OPF approach for feature selection

The HS algorithm is an interesting approach for several applications mainly because of its simplicity and low computational cost. Similarly, the Optimum-Path Forest classifier is very fast for training from the patterns and, in addition, has been demonstrated to be accurate [13]. This way, we believe that the combination of HS and OPF may provide a fast and robust solution for feature selection, specially in large datasets, where the number of features to be selected is usually high, as well as the number of samples.

The proposed hybrid algorithm takes OPF as the fitness function to guide the HS algorithm. Basically, for each harmony in the HM, the OPF classifier is trained with the corresponding subset of features in a training set, and its accuracy is assessed over an evaluating set. This procedure is presented in Algorithm 1.

Algorithm 1. Feature Selection Through Harmony Search

Input: λ-labeled training Z_1 and evaluating set Z_2, Harmony Memory Size *HMS*, number of features *N*, Harmony Memory Considering Rate *HMCR* and number of iterations *T* for convergence.
Output: Harmony vector H^* of size *N* that gives the highest classification accuracy over Z_2.

Auxiliary: Harmony Memory HM of size $HMS \times (N+1)$, new harmony vector *H* of size $N+1$ and variable *tmp*.

```
For $i \leftarrow 1$ to $HMS$, do
    For $j \leftarrow 1$ to $N$, do
        $HM_i j \leftarrow$ Random{0,1}.
    Create $Z'_1$ and $Z'_2$ from $Z_1$ and $Z_2$, respectively, such
    that both contain features $HM_{i,j} = 1, j = 1,2,\ldots,N$.
    Train OPF over $Z'_1$, evaluate it over $Z'_2$ and stores the
    accuracy in $HM_{i,N+1}$.
For $k \leftarrow 1$ to $T$, do
    $tmp \leftarrow \lfloor HMCR * N \rfloor$.
    For $j \leftarrow 1$ to $tmp$, do
        $H_j \leftarrow n(HM_{ij}), i = 1,2,\ldots,HMS$.
    For $j \leftarrow tmp + 1$ to $N$, do
        $H_j \leftarrow Random\{0,1\}, i = 1,2,\ldots,HMS$.
    Find $i$ such that $HM_{i,N+1}$ is minimum.
    If $(H_{N+1} > HM_{i,N+1})$, then
        For $j \leftarrow 1$ to $N+1$, do
            $HM_{i,j} \leftarrow H_j$.
    Create $Z'_1$ and $Z'_2$ from $Z_1$ and $Z_2$, respectively, such
    that both contain features $HM_{i,j} = 1, j = 1,2,\ldots,N$.
    Train OPF over $Z'_1$, evaluate its over $Z'_2$ and stores the
    accuracy in $HM_{i,N+1}$.
Find $i$ such that $HM_{i,N+1}$ is maximum.
For $j \leftarrow 1$ to N, do
    $H^*_j \leftarrow HM_{i,j}$.
Returns $[H^*]$
```

The initial loop in Lines 1–7 is responsible for Harmony Memory initialization. In Line 3, each position of the Harmony Memory $HM_{i,j}$ receives a binary value indicating whether that feature will be selected or not. Lines 4–5 create new training and evaluating sets only with the previously selected features, i.e., $HM_{i,j} = 1$, $i = 1,2,\ldots,HMS$ and $j = 1,2,\ldots,N$. In Lines 6–7, the OPF classifier is trained and evaluated with these new sets and its accuracy over the evaluating set is stored in $HM_{i,N+1}$. Line 9 calculates the number of features that will be selected from the Harmony Memory to compose the new harmony *H*. The loop in Lines 10–11 selects values from the Harmony Memory to build the new harmony vector *H*. The function $n(HM_{i,j})$ returns the more recurrent value for the feature *j*. Further, the next loop in Lines 12–13 ends the memory consideration step by allocating random values $x \in \{0,1\}$ to the remaining positions of *H*. Recall that Lines 9–13 refer to Eq. 3.

Line 14 finds the harmony index *i* in $HM_{i,N+1}$ that achieved the lowest classification accuracy over some Z'_2, and replaces the whole harmony HM_i with the new *H* obtained if $HM_{i,N+1}$ is lower than the one given by H_{N+1} (Lines 15–17). The Lines 18–21 are responsible for updating the Harmony Memory fitness values. Lines 18–19 create new training and evaluating sets only with the features so $HM_{i,j} = 1$, $i = 1,2,\ldots,HMS$ and $j = 1,2,\ldots,N$. In Lines 20–21, the OPF classifier is trained and evaluated on these new sets, and its accuracy over the evaluating set is stored in $HM_{i,N+1}$. Finally, Line 22 finds the harmony in the Harmony Memory that gives the highest OPF classification accuracy over some Z'_2 and stores it in the best harmony vector H^* in Lines 23–24. In this algorithm, the pitch adjustment step was not applied because the range of values for decision variables is composed only by binary values.

5. Experiments

In this section we describe the dataset used in the experiments, as well as the features considered for each consumer profile and the experiments itself.

5.1. Dataset

We used two labeled datasets obtained from a Brazilian electric power company, B_i and B_c. The former is a dataset composed of 3486 samples related to industrial consumers, and the latter contains 5645 samples of commercial consumers. Both the industrial and the commercial profiles are represented by the eight following features:

- Demand Billed (DB): demand value of the active power considered for billing purposes, in kilowatts (kW).
- Demand Contracted (DC): the value of the demand for continuous availability requested from the energy company, which must be paid whether electric power is used by the consumer or not, in kilowatts (kW).
- Demand Measured or Maximum Demand (D_{max}): the maximum actual demand for active power, verified by measurement at 15-min intervals during the billing period, in kilowatts (kW).
- Reactive Energy (RE): energy that flows through the electric and magnetic fields of an AC system, in kilovolt-amperes reactive hours (kVArh).
- Power Transformer (PT): the power transformer installed for consumers, in kilovolt-amperes (kVA).
- Power Factor (PF): the ratio between the consumed active and the apparent power in a circuit. The PF indicates the efficiency of a power distribution system.
- 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).
- Load Factor (LF): the ratio between the average demand ($D_{average}$) and the maximum demand (D_{max}) of the consumer unit. The *LF* is an index that shows how electric power is used in a rational way.

5.2. Experimental results

We performed two rounds of experiments. In the former (Section 5.2.1), we evaluated the robustness of the most actively pursued pattern recognition techniques for identifying non-technical losses, say that OPF, SVM–RBF (SVM with Radial Basis Function – RBF as the kernel function), SVM-noKernel (SVM without kernel mapping), ANN–MLP (Artificial Neural Networks with Multilayer Perceptron – MLP), SOM (Kohonen Self Organizing Maps) and the *k*-Nearest Neighbors (*k*-NN) algorithm. For both datasets, we employed 25% of the samples for training the classifiers and 50% for testing them. In the latter round of experiments (Section 5.2.2), we applied the same training set as before to train OPF with PSO for feature selection (PSO–OPF) and OPF with HS (HS–OPF). The remaining samples not used in the previous experiments (25% of the whole dataset) were now applied to compose the evaluation set, over which the OPF accuracy is taken as the fitness value for both PSO and HS.

Considering that both feature selection and feature extraction aim at reducing the dimensionality of the original space in a discriminative way, we also considered the employment of PCA and KPCA to this purpose as additional baselines.

For SVM–RBF, we used the latest version of the LibSVM package [20] with RBF kernel, parameter optimization and the one-versus-one strategy for the multi-class problem. With respect to SVM-noKernel, we used the LibLINEAR package [21] with parameter optimization. For OPF we used the LibOPF [22], which is a library for designing Optimum-Path Forest-based classifiers, and for ANN–MLP we used the Fast Artificial Neural Network Library (FANN) [23], which was trained using the backpropagation algorithm. The network configuration was $i{:}h_1{:}h_2{:}o$, where $i = 8$ (number of features), $h_1 = h_2 = 8$ and $o = 2$ (number of classes) is the number of neurons in the input, hidden and output layers, respectively. The ANN–MLP was trained with a backpropagation algorithm, and its architecture was chosen empirically. For the SOM network we used a lattice with 100×100 neurons with 10 iterations for the algorithm convergence. Regarding the *k*-NN algorithm we used our own implementation, in which the *k* value chosen was the one that maximized the accuracy over the training set. Finally, with respect to the computational resources, the experiments were carried out on a PC equipped with a Pentium® Core i5 processor, with 4 Gb of RAM and having Linux Ubuntu 10.04 LTS as the operational system.

5.2.1. Robustness evaluation

Table 1 shows the results for B_c and B_i datasets in the format $x[y]$, where x and y mean, respectively, the accuracy rate and the training time (in seconds).

Table 1
Accuracy rate and training time for B_c and B_i datasets.

Classifier	B_c	B_i
OPF	55.88%[0.106]	64.65%[0.043]
SVM–RBF	51.46%[82.64]	53.53%[34.02]
SVM-noKernel	51.27%[49.15]	45.38%[21.46]
ANN–MLP	50.00%[89.16]	50.00%[58.91]
SOM	52.15%[95.56]	60.10%[53.03]
k-NN	56.20%[26.21]	65.02%[4.848]

One can see that OPF and *k*-NN achieved similar results for the B_c dataset, outperforming the remaining classifiers. However, OPF was 247.26 times faster than *k*-NN for training. Regarding the B_i dataset, OPF was 112.74 faster than *k*-NN for training. Although one may argue that training time is not crucial, it is important to shed light that some applications require fast training phases. In the one addressed here, for each candidate solution (particle or harmony, for instance), its fitness value has to be calculated, which refers to the classifier accuracy over the evaluation set. Thus, for each possible solution, the classifier has to be trained and evaluated over the respective sets.

5.2.2. Evaluation of the proposed algorithm for feature selection

In this section we compared the same OPF as in the previous section, i.e., without feature selection, to PSO–OPF and HS–OPF. Additionally, we also compared these results to the ones obtained with the employment of the PCA and the KPCA feature extraction techniques over the joined training and evaluation sets. In the KPCA, we employed a Gaussian kernel with $\sigma = 1$.

For both PCA and KPCA, we first present the results in the usual way of retaining the γ features related to the greater eigenvalues, denoted here, respectively, as PCA_γ–OPF and $KPCA_\gamma$–OPF. Then, we present the results that were obtained by letting HS and PSO decide which extracted features are better to use. While assessing the potential of these techniques (i.e., PCA and KPCA), we could also have some insight about whether they are worth applying in these datasets regardless the criteria to reduce the dimensionality. These last approaches are denoted as PSO–PCA_8–OPF, HS–PCA_8–OPF, PSO–$KPCA_8$–OPF, and HS–$KPCA_8$–OPF.

Table 2 displays the parameters used to tune PSO and HS algorithms. These values were empirically chosen in order to avoid meta-optimization.

The results for the B_c and the B_i datasets are shown in Table 3. One can note an impressive improvement in accuracy by the HS–OPF and PSO–OPF methods. For dataset B_c, both methods selected four features as most discriminative, as follow: DC, $D_{\max}$, PF and LF. For dataset B_i, HS–OPF and PSO–OPF also agreed about the features selected. In this case, the features were DC, $D_{\max}$, P_{inst}, LF. Regarding the features selected for B_c, the only difference is that P_{inst} was chosen instead of PF.

Although PSO–OPF and HS–OPF achieved similar accuracy in dataset B_c, the former executed in 556.22 s and the proposed approach needed only 60.50 s. Likewise, PSO–OPF needed 579.93 s to be executed in dataset B_i while HS–OPF executed in 59.88 s. Hence, HS–OPF was 9.19 times faster than PSO–OPF in the first dataset and 9.68 times faster in the second one. This characteristic may be very important in datasets with high dimensions and a large number of samples.

It is also possible to observe that PCA and KPCA slightly improved the accuracy of OPF in both datasets when retaining the features in the usual way (i.e., by the eigenvalues) and no consistent difference in performance was noticed between them. When applying PSO and HS in the feature spaces derived by PCA and KPCA, it becomes clear that the original space contains

Table 2
PSO and HS parameters.

HS	PSO
# of iterations = 10	# of iterations = 10
HMS = 300	# of particles = 300
HMCR = 0.65	$w = 1.2$, $c_1 = 2.8$, $c_2 = 0.9$

Table 3
Accuracy rates for datasets B_c and B_i.

Classifier	Accuracy on B_c (%)	Accuracy on B_i (%)
OPF	55.88	64.65
PSO–OPF	**92.17**	**96.50**
HS–OPF	**92.60**	**96.50**
PCA_2–OPF	56.71	64.15
PCA_3–OPF	56.62	66.41
PCA_4–OPF	57.71	67.32
PCA_5–OPF	57.69	66.34
PCA_6–OPF	57.78	66.31
PCA_7–OPF	57.78	66.31
PCA_8–OPF	57.78	66.31
PSO–PCA_8–OPF	79.46	62.82
HS–PCA_8–OPF	78.83	60.07
$KPCA_2$–OPF	56.58	61.32
$KPCA_3$–OPF	57.12	64.18
$KPCA_4$–OPF	56.36	64.55
$KPCA_5$–OPF	56.33	58.77
$KPCA_6$–OPF	55.39	58.10
$KPCA_7$–OPF	54.14	59.03
$KPCA_8$–OPF	55.59	58.76
PSO–$KPCA_8$–OPF	87.38	69.64
HS–$KPCA_8$–OPF	90.61	69.41

features that confuse not only the classification task, but also these feature extraction techniques. The fact that the results were almost always better when not considering the features by their associated eigenvalues emphasizes this idea.

6. Conclusions

Non-technical losses in power distribution systems are related to the consumed but no billed energy and it represents a problem to the electric power companies, since these losses are strongly related with frauds and energy theft. Although several works have addressed this point, a few of them addressed the problem of selecting the most relevant features in order to identify possible illegal consumers.

As the problem of selecting a subset of features that maximizes some classifier's accuracy can be modeled as an optimization procedure, several techniques have been applied to handle this task. In this paper, we proposed a novel and fast hybrid algorithm to accomplish the task of feature selection by using Harmony Search and Optimum-Path Forest (HS–OPF). The main idea is to use the OPF accuracy over an evaluating set as the fitness value to guide the Harmony Search algorithm on finding the best combination of features. Despite the fact that some setups of feature selection become prohibitive due to their computational cost, this is not the case of HS–OPF, which is made by the combination of two very efficient approaches.

We validated the proposed approach in two labeled datasets obtained from a Brazilian electric power company, one composed by commercial consumers and the other composed by industrial consumers.

The experiments were carried out in two rounds. In the former, we used all available features and compared OPF to SVM–RBF, SVM-noKernel, ANN–MLP, SOM and *k*-NN. In the latter, we compared HS–OPF to the former OPF, to PSO–OPF and to 18 variations of classifiers built with the employment of the PCA and the KPCA feature extraction techniques. PSO–OPF differed from HS–OPF only by the search algorithm.

In the first round of experiments, OPF and *k*-NN were the most accurate classifiers, with OPF being much faster for training. With respect to the second round of experiments, HS–OPF achieved similar results to the ones obtained by PSO–OPF, but HS–OPF was about 9 times faster to identify the most relevant features. Some other interesting findings were observed by analyzing the results obtained with PCA and KPCA. The importance of selecting a discriminant subset of features was reinforced by the new and fast method.

Acknowledgment

The authors are grateful to Capes and FAPESP Grants 2009/16206-1 and 2010/00994-8.

References

[1] Nizar A, Dong Z, Wang Y. Power utility nontechnical loss analysis with extreme learning machine method. IEEE Trans Power Syst 2008;23:946–55.
[2] Nagi J, Mohammad AM, Yap KS, Tiong SK, Ahmed SK. Nontechnical loss detection for metered consumers in power utility using support vector machines. IEEE Trans Power Deliv 2010;25:1162–71.
[3] Nagi J, Mohammad AM, Yap KS, Tiong SK, Ahmed SK. Non-technical loss analysis for detection of electricity theft using support vector machines. In: Proceedings of the 2nd IEEE international power and energy conference; 2008, p. 907–12.
[4] Nagi J, Yap KS, Tiong SK, Ahmed SK, Mohammad AM. Detection of abnormalities and electricity theft using genetic support vector machines. In: IEEE TENCON Region 10 Conference; 2008, p. 1–6.
[5] Nizar AH, Dong ZY, Zhao JH, Zhang P. A data mining based ntl analysis method. In: Proceedings of the IEEE power engineering society general meeting; 2007, p. 1–8.
[6] Monedero I, Biscarri F, León C, Biscarri J, Millán R. Midas: detection of non-technical losses in electrical consumption using neural networks and statistical techniques. In: Proceedings of the international conference on computational science and applications, vol. 3984. Heidelberg: Lecture Notes in Computer Science, Springer Berlin; 2006.
[7] Ramos CCO, Souza AN, Papa JP, Falcão AX. A new approach for nontechnical losses detection based on optimum-path forest. IEEE Trans Power Syst 2011;26:181–9.
[8] Firpi HA, Goodman E. Swarmed feature selection. In: Proceedings of the 33rd applied imagery pattern recognition workshop; 2004, p. 112–8.
[9] Yang J, Honavar V. Feature subset selection using a genetic algorithm. IEEE Intell Syst Appl 1998;13(2):44–9.
[10] Diao R, Shen Q. Two new approaches to feature selection with harmony search. In: Proceedings of the 19th international conference on fuzzy systems; 2010, p. 3161–7.
[11] Ramos CCO, Souza AN, Chiachia G, Falc ao AX, Papa JP. What is the importance of selecting features for non-technical losses identification? In: IEEE international symposium on circuits and systems, Rio de Janeiro, Brazil; 2011, p. 1045–8.
[12] Falcão AX, Stolfi J, Lotufo RA. The image foresting transform theory, algorithms, and applications. IEEE Trans Pattern Anal Mach Intell 2004;26(1):19–29.
[13] Papa JP, Falcão AX, Suzuki CTN. Supervised pattern classification based on optimum-path forest. Int J Imaging Syst Technol 2009;19(2):120–31.
[14] Papa JP, Falcão AX. A new variant of the optimum-path forest classifier. In: Proceedings of the 4th international symposium on advances in visual computing. Heidelberg: Springer-Verlag, Berlin; 2008, p. 935–44.
[15] Rocha LM, Cappabianco FAM, Falcão AX. Data clustering as an optimum-path forest problem with applications in image analysis. Int J Imaging Syst Technol 2009;19(2):50–68.
[16] Geem ZW. Novel derivative of harmony search algorithm for discrete design variables. Appl Math Comput 2008;199:223–30.
[17] Kennedy J, Eberhart R. Swarm intelligence. Morgan Kaufman; 2001.
[18] Kennedy J, Eberhart RC. A discrete binary version of the particle swarm algorithm. In: IEEE International Conference on Systems, Man, and Cybernetics, vol. 5; 1997, p. 4104–8.
[19] Firpi HA, Goodman E. Swarmed feature selection. In: Proceedings of the 33rd applied imagery pattern recognition workshop, IEEE Computer Society, Washington, DC, USA, 2004, p. 112–8.
[20] Chang CC, Lin CJ. LIBSVM: A Library for Support Vector Machines. Available at url <http://www.csie.ntu.edu.tw/~cjlin/libsvm> (2001).
[21] Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ. LIBLINEAR: a library for large linear classification. J Mach Learn Res 2008;9:1871–4.

[22] Papa JP, Suzuki CTN, Falcão AX. LibOPF: a library for the design of optimum-path forest classifiers, software version 2.0. Available from: <http://www.ic.unicamp.br/afalcao/LibOPF>, 2009.
[23] Nissen S. Implementation of a Fast Artificial Neural Network Library (FANN), department of Computer Science University of Copenhagen (DIKU). Software available at <http://leenissen.dk/fann/>, 2003.

Caio C.O. Ramos received his B.Sc. in Electrical Engineering (2006) from UNESP – Univ Estadual Paulista, SP, Brazil. In 2010, he received his M.Sc. in Electrical Engineering from the same university, and his interests include intelligent systems, transformers, fraud detection in electrical systems and energy system protection. Currently, he is working on his doctoral degree in Electrical Engineering from the Polytechnical School of the University of São Paulo, SP, Brazil.

André N. de Souza received his B.Sc. in Electrical Engineering (1991) from the Mackenzie University, SP, Brazil. In 1995 and 1999, respectively, he received his M.Sc. and Ph.D. in Electrical Engineering at the Polytechnical School from the University of São Paulo, SP, Brazil. He has been Associate Professor at the Electrical Engineering Department, UNESP – Univ Estadual Paulista, since 2005. His interests include intelligent systems, transformers, fraud detection in electrical systems, atmospheric discharges and power quality.

Giovani Chiachia received his B.Sc. in Information Systems from the São Paulo State University, SP, Brazil. In 2009, he received his M.Sc. in Computer Science also from São Paulo State University, SP, Brazil. Since then he is a Ph.D. Candidate in Computer Science at University of Campinas, SP, Brazil. His research interests include black-box optimization, pattern recognition, and visual computing.

Alexandre X. Falcão received the B.Sc. degree in Electrical Engineering from the University of Pernambuco, PE, Brazil, in 1998 and the M.Sc. and the Ph.D. degrees in Electrical Engineering from the University of Campinas, SP, Brazil, in 1993 and 1996, respectively. He has worked in image processing and analysis since 1991. He has been Professor at the Institute of Computing, University of Campinas, since 1998, and his research interests include image segmentation and analysis, volume visualization, content-based image retrieval, mathematical morphology, digital TV, medical imaging applications and pattern recognition.

João P. Papa received his B.Sc. in Information Systems from the UNESP – Univ Estadual Paulista, SP, Brazil. In 2005, he received his M.Sc. in Computer Science from the Federal University of São Carlos, SP, Brazil. In 2008, he received his Ph.D. in Computer Science from the University of Campinas, SP, Brazil. During 2008–2009, he had worked as post-doctorate researcher at the same institute. He has been Professor at the Computer Science Department, UNESP – Univ Estadual Paulista, since 2009, and his research interests include machine learning, pattern recognition and image processing.