NEW TRENDS IN MUSICAL GENRE CLASSIFICATION USING OPTIMUM-PATH FOREST

C. Marques, I. R. Guilherme
UNESP - Univ Estadual Paulista, Rio Claro, SP
Dep. of Statistics, Applied Math. and Computation
marques@caiena.net,ivan@rc.unesp.br

R. Y. M. Nakamura, J. P. Papa
UNESP - Univ Estadual Paulista, Bauru, SP
Department of Computing
{rodrigo.mizobe,papa}@fc.unesp.br

ABSTRACT

Musical genre classification has been paramount in the last years, mainly in large multimedia datasets, in which new songs and genres can be added at every moment by anyone. In this context, we have seen the growing of musical recommendation systems, which can improve the benefits for several applications, such as social networks and collective musical libraries. In this work, we have introduced a recent machine learning technique named Optimum-Path Forest (OPF) for musical genre classification, which has been demonstrated to be similar to the state-of-the-art pattern recognition techniques, but much faster for some applications. Experiments in two public datasets were conducted against Support Vector Machines and a Bayesian classifier to show the validity of our work. In addition, we have executed an experiment using very recent hybrid feature selection techniques based on OPF to speed up feature extraction process.

1. INTRODUCTION

Recently, advances in technology have supported the storage of large amount of data. Therefore, fast information retrieval became a hot challenge. One of the most interesting applications concerns with social network users, which have looked forward to meet people that share common preferences, and also to discover new good music. Thus, an important task in this context is the music classification into different genres aiming a better organization of music datasets, for further recommendation.

Tzanetakis and Cook [22] proposed a work to deal with the problem of musical genre classification using three sets of features representing timbral texture, rhythmic and pitch contents, together with *K*-Nearest Neighbors and Gaussian Mixture Models. Lambrou et al. [11] applied statistical features in temporal domain and three different wavelet transforms for the same task, and Soltau et al. [21] proposed a new architecture called ETM-NN (Explicit Time Modeling with Neural Networks), which employs statistical analysis of a temporal structure.

Xu et al. [24] applied Support Vector Machines (SVMs) to perform a hierarchical classification of different musical genres, and Dellandrea et al. [6] compared SVMs with Neural Networks in the same context. Chan and Vasconcelos [2] proposed a Dynamic Texture Model (DTM) for automatic music segmentation, and Coviello et al. [4] introduced DTM in the context of music tagging. McKay and Fujinaga [13] proposed a novel hybrid system to handle automatic classification of musical genres composed by a Feedfoward Neural Network and *K*-Nearest Neighbors algorithm together with Genetic Algorithms (GA) for feature selection. Finally, Deepa et al. [5] used a brute force method for feature optimization using different feature vectors with SVMs. The idea is to combine the best ones at the final of the process.

In order to combine efficiency for training and effectiveness in the classification task, a novel framework that reduces the pattern recognition problem to an optimum path forest computation (OPF) in the feature space induced by a graph was presented in its unsupervised [20] and supervised versions [14]. The OPF-based classifiers do not interpret the classification task as a hyperplanes optimization problem, but as a combinatorial optimum-path computation from some key samples (prototypes) to the remaining nodes. Each prototype becomes a root from its optimum-path tree and each node is classified according to its strongly connected prototype, that defines a discrete optimal partition (influence region) of the feature space. The OPF framework has some advantages with respect to the aforementioned classifiers: (i) it is free of parameters (supervised version), (ii) does not assume any shape/separability of the feature space and (iii) it runs training phase faster.

In this paper, we propose to introduce the supervised OPF in the context of musical genre classification. As far as we know, we are the first to apply OPF for this task. In re-

gard to feature selection in the context of musical genre classification, it is not usual to find many works on that. Therefore, we would like to shed light over that another main contribution of this paper is to introduce three recently developed feature selection techniques aiming to improve musical genre classification: HS-OPF (Harmony Search with OPF) [18], PSO-OPF (Particle Swarm Optimization with OPF) [17] and GSA-OPF (Gravitational Search Algorithm with OPF) [15]. The experiments are conducted in two rounds: (i) in the former, OPF is compared with SVMs and a Bayesian classifier, and (ii) in the second round we present a comparison between HS-OPF, PSO-OPF and GSA-OPF for feature selection in the context of musical genre classification. The remainder of the paper is organized as follows. The OPF theory is presented in Section 2. The experimental results are discussed in Section 3. Finally, conclusions are stated in Section 4.

2. SUPERVISED OPTIMUM-PATH FOREST

The OPF classifier works by modeling the problem of pattern recognition as a graph partition in a given feature space. The nodes are represented by the feature vectors and the edges connect all pairs of them, defining a full connectedness graph. This kind of representation is straightforward, given that the graph does not need to be explicitly represented, allowing us to save memory. The partition of the graph is carried out by a competition process between some key samples (*prototypes*), which offer optimum paths to the remaining nodes of the graph. Each prototype sample defines its optimum-path tree (OPT), and the collection of all OPTs defines an optimum-path forest, which gives the name to the classifier [14].

The OPF can be seen as a generalization of the well known Dijkstra's algorithm to compute optimum paths from a source node to the remaining ones [7]. The main difference relies on the fact that OPF uses a set of source nodes (prototypes) with any path-cost function. In case of Dijkstra's algorithm, a function that summed the arc-weights along a path was applied. For OPF, we used a function that gives the maximum arc-weight along a path, as explained before. Next section states OPF theory.

2.1 Background Theory

Let $Z = Z_1 \cup Z_2$ be a dataset labeled with a function λ, in which Z_1 and Z_2 are, respectively, a training and test sets such that Z_1 is used to train a given classifier and Z_2 is used to assess its accuracy. Let $S \subseteq Z_1$ a set of prototype samples. Essentially, the OPF classifier creates a discrete optimal partition of the feature space such that any sample $s \in Z_2$ can be classified according to this partition. This partition is an optimum path forest (OPF) computed in $\Re^n$ by the image foresting transform (IFT) algorithm [8].

The OPF algorithm may be used with any *smooth* path-cost function which can group samples with similar properties [8]. Particularly, we used the path-cost function f_{max}, which is computed as follows:

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

in which $d(s,t)$ means the distance between samples s and t, and a path π is defined as a sequence of adjacent samples. In such a way, we have that $f_{max}(\pi)$ computes the maximum distance between adjacent samples in π, when π is not a trivial path.

The OPF algorithm assigns one optimum path $P^*(s)$ from S to every sample $s \in Z_1$, forming an optimum path forest P (a function with no cycles which assigns to each $s \in Z_1 \backslash S$ its predecessor $P(s)$ in $P^*(s)$ or a marker nil when $s \in S$. Let $R(s) \in S$ be the root of $P^*(s)$ which can be reached from $P(s)$. The OPF algorithm computes for each $s \in Z_1$, the cost $C(s)$ of $P^*(s)$, the label $L(s) = \lambda(R(s))$, and the predecessor $P(s)$.

The OPF classifier is composed of two distinct phases: (i) training and (ii) classification. The former step consists, essentially, in finding the prototypes and computing the optimum-path forest, which is the union of all OPTs rooted at each prototype. After that, we take a sample from the test sample, connect it to all samples of the optimum-path forest generated in the training phase and we evaluate which node offered the optimum path to it. Notice that this test sample is not permanently added to the training set, i.e., it is used only once. The next sections describe in details this procedure.

2.1.1 Training

We say that S^* is an optimum set of prototypes when the OPF algorithm minimizes the classification errors for every $s \in Z_1$. S^* can be found by exploiting the theoretical relation between minimum-spanning tree (MST) and optimum-path tree for f_{max} [1]. The training essentially consists in finding S^* and an OPF classifier rooted at S^*.

By computing an MST in the complete graph (Z_1, A), we obtain a connected acyclic graph whose nodes are all samples of Z_1 and the arcs are undirected and weighted by the distances d between adjacent samples. The spanning tree is optimum in the sense that the sum of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to f_{max}. That is, the minimum-spanning tree contains one optimum-path tree for any selected root node. The optimum prototypes are the closest elements of the MST with different labels in Z_1 (i.e., elements that fall in the frontier of the classes). Algorithm 1 implements the training procedure for OPF.

Algorithm 1 – OPF TRAINING ALGORITHM

INPUT:	A λ-labeled training set Z_1 and the pair (v, d) for feature vector and distance computations.
OUTPUT:	Optimum-path forest P_1, cost map C_1, label map L_1, and ordered set Z'_1.
AUXILIARY:	Priority queue Q, set S of prototypes, and cost variable cst.

1. *Set* $Z'_1 \leftarrow \emptyset$ *and compute by MST the prototype set* $S \subset Z_1$.
2. *For each* $s \in Z_1 \backslash S$, *set* $C_1(s) \leftarrow +\infty$.
3. *For each* $s \in S$, *do*
4. $\quad C_1(s) \leftarrow 0$, $P_1(s) \leftarrow nil$, $L_1(s) \leftarrow \lambda(s)$, *insert* s *in* Q.
5. *While* Q *is not empty, do*
6. $\quad$ *Remove from* Q *a sample* s *such that* $C_1(s)$ *is minimum.*
7. $\quad$ *Insert* s *in* Z'_1.
8. $\quad$ *For each* $t \in Z_1$ *such that* $C_1(t) > C_1(s)$, *do*
9. $\quad\quad$ *Compute* $cst \leftarrow \max\{C_1(s), d(s,t)\}$.
10. $\quad\quad$ *If* $cst < C_1(t)$, *then*
11. $\quad\quad\quad$ *If* $C_1(t) \neq +\infty$, *then remove* t *from* Q.
12. $\quad\quad\quad$ $P_1(t) \leftarrow s$, $L_1(t) \leftarrow L_1(s)$, $C_1(t) \leftarrow cst$.
13. $\quad\quad\quad$ *Insert* t *in* Q.
14. *Return a classifier* $[P_1, C_1, L_1, Z'_1]$.

The time complexity for training is $\theta(|Z_1|^2)$, due to the main (Lines 5-13) and inner loops (Lines 8-13) in *Algorithm 1*, which run $\theta(|Z_1|)$ times each.

2.1.2 Classification

For any sample $t \in Z_2$, we consider all arcs connecting t with samples $s \in Z_1$, as though t were part of the training graph. Considering all possible paths from S^* to t, we find the optimum path $P^*(t)$ from S^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in S^*$. This path can be identified incrementally by evaluating the optimum cost $C(t)$ as

$$C(t) = \min\{\max\{C(s), d(s,t)\}\}, \ \forall s \in Z_1. \quad (2)$$

Let the node $s^* \in Z_1$ be the one that satisfies Equation 2 (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ as the class of t. An error occurs when $L(s^*) \neq \lambda(t)$. Algorithm 2 implements this procedure.

Algorithm 2 – OPF CLASSIFICATION ALGORITHM

INPUT:	Classifier $[P_1, C_1, L_1, Z'_1]$, evaluation set Z_2 (or test set Z_3), and the pair (v, d) for feature vector and distance computations.
OUTPUT:	Label L_2 and predecessor P_2 maps defined for Z_2.
AUXILIARY:	Cost variables tmp and $mincost$.

1. *For each* $t \in Z_2$, *do*
2. $\quad i \leftarrow 1$, $mincost \leftarrow max\{C_1(k_i), d(k_i, t)\}$.
3. $\quad L_2(t) \leftarrow L_1(k_i)$ *and* $P_2(t) \leftarrow k_i$.
4. $\quad$ *While* $i < |Z'_1|$ *and* $mincost > C_1(k_{i+1})$, *do*
5. $\quad\quad$ *Compute* $tmp \leftarrow \max\{C_1(k_{i+1}, d(k_{i+1}, t)\}$.
6. $\quad\quad$ *If* $tmp < mincost$, *then*
7. $\quad\quad\quad mincost \leftarrow tmp$.
8. $\quad\quad\quad L_2(t) \leftarrow L(k_{i+1})$ *and* $P_2(t) \leftarrow k_{i+1}$.
9. $\quad\quad i \leftarrow i + 1$.
10. *Return* $[L_2, P_2]$.

In *Algorithm 2*, the main loop (Lines $1-9$) performs the classification of all nodes in Z_2. The inner loop (Lines $4-9$) visits each node $k_{i+1} \in Z'_1$, $i = 1, 2, \ldots, |Z'_1| - 1$ until an optimum path $\pi_{k_{i+1}} \cdot \langle k_{i+1}, t\rangle$ is found.

2.2 Accuracy Computation

The accuracies are measured by taking into account that the classes may have different sizes in Z_2. If there are two classes, for example, with very different sizes and a classifier always assigns the label of the largest class, its accuracy will fall drastically due to the high error rate on the smallest class.

Let $NZ_2(i)$, $i = 1, 2, \ldots, c$, be the number of samples in Z_2 from each class i. We define

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|} \text{ and } e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \ i = 1, \ldots, c \quad (3)$$

where $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. That is, $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2, and $FN(i)$ is the number of samples from class i that were incorrectly classified as being from other classes in Z_2.

The errors $e_{i,1}$ and $e_{i,2}$ are used to define

$$E(i) = e_{i,1} + e_{i,2}, \quad (4)$$

where $E(i)$ is the partial sum error of class i. Finally, the accuracy is written as

$$Acc = \frac{2c - \sum_{i=1}^{c} E(i)}{2c} = 1 - \frac{\sum_{i=1}^{c} E(i)}{2c}. \quad (5)$$

3. EXPERIMENTAL RESULTS

In this section, we described the experiments concerning automatic music genre classification using two public datasets: (i) GTZAN Genre Collection [22] and (ii) Magnatagatune [12]. Table 1 displays the description of the datasets. It is important to notice that we have used a subset of GTZAN dataset.

In regard to music description, for GTZAN dataset we have employed the Marsyas [23] software to extract Mel-Frequency Cepstral Coefficients (MFCC) over sequential windows with size $\approx$ 23ms each. We analyzed 30s of each

Dataset	# Samples	# Features	# Labels
GTZAN	999	33618	10
Magnatagatune	11493	74	15

Table 1. Description of the datasets used in the experiments.

music, obtaining 1293 windows with 26 cepstral coefficients each. Finally, with respect to Magnatagatune dataset, we have used timbre features already extracted and available with that dataset to compose our feature vector with 74 characteristics.

We have conducted two round of experiments: (i) in the former (Section 3.1) we address the robustness of supervised classifiers for musical genre classification, and (ii) in the latter (Section 3.2) we assess the effectiveness of OPF after a feature selection procedure over the original datasets. Notice that the feature selection algorithms are hybrid methodologies based on OPF and three optimization techniques: Harmony Search (HS) [9], Gravitational Search Algorithm (GSA) [19] and Particle Swarm Optimization [10].

The main idea of such algorithms is to use the accuracy over an evaluating set as the fitness function to guide the optimization process. Thus, the feature selection algorithm is designed over training and evaluating sets in order to find suitable subsets of features that lead to good recognition rates over the unseen test set. These hybrid algorithms employ the OPF as the basis classifier [15, 17, 18], since it is very fast and robust, as one can see in the next sections.

3.1 Musical Genre Classification Through Supervised Classification

In this section, we described the experiments conducted to assess the robustness of OPF in the context of musical genre classification. In regard to classifiers, we have compared OPF against SVMs with Radial Basis Function (SVM-RBF) and Bayesian classifier (Bayes). For OPF we adopted the LibOPF [16], and with respect to SVM-RBF we employed SVMTorch [3]. Finally, for Bayesian classifier we used our implementation.

We employed the traditional holdout method with 50% for training and the remaining 50% to compose the test set. The experiments were executed over 10 running with randomly generated training and test sets in order to compute the mean accuracy and training and test times (seconds). Notice that all parameters used in this experiment were empirically chosen, based on our experience. Table 2 displays the recognition rates.

One can see that OPF, Bayes and SVM-RBF achieved similar results for both datasets if one considers the standard deviation. However, in GTZAN dataset OPF was 2.92 and 6.23 times faster than SVMTorch for training and classification, respectively. In regard to Magnatagatune dataset, OPF was 45.80 and 9.39 times faster than SVM-RBF for training and classification, respectively.

Dataset	**Classifier**	**Acc**	**Tr [s]**	**Ts [s]**
GTZAN	OPF	98.61±0.75	9.19	4.40
GTZAN	Bayes	98.54±0.82	1.71	94.31
GTZAN	SVM-RBF	98.72±0.09	26.98	27.23
Magnatagatune	OPF	62.34±0.82	3.55	3.73
Magnatagatune	Bayes	61.58±0.81	2.33	46.53
Magnatagatune	SVM-RBF	63.15±0.03	162.59	35.04

Table 2. Mean accuracy, training (Tr) and testing (Ts) times in seconds.

Although Bayes has been the fastest classifier for training, if one considers the whole execution time, i. e., training and classification, OPF has been the fastest approach.

3.2 Feature selection

In regard to feature selection, we have evaluated three algorithms: PSO-OPF [17], HS-OPF [18] and GSA-OPF [15]. For that, we have used 30% to compose the training set, 20% to the evaluating one and the remaining 50% for the test set. Table 3 displays the parameters used to tune the algorithms. The number of iterations for convergence has been set to 10 for all approaches. The same occurs with the number of initial solutions, i.e., number of particles for PSO-OPF, number of harmonies for HS-OPF and number of masses for GSA-OPF, which has been set to 100. Notice that these values were empirically chosen in order to avoid meta-optimization.

Table 3. PSO-OPF, HS-OPF and GSA-OPF parameters.

PSO-OPF	**HS-OPF**	**GSA-OPF**
$c_1 = 1.4$, $c_2 = 0.6$ $w = 0.7$	$HMCR = 0.7$	$\epsilon = 0.7$, $G_0 = 10$ $k = 100$

Table 4 displays the results. One can see that all techniques have obtained the same results for both datasets. The difference relies on the execution time, in which PSO-OPF and HS-OPF have been executed in a similar period of time, being up to 2 times faster than GSA-OPF. We can see that PSO-OPF has selected 16772 out 33618 features for GTZAN dataset, which means about 100 % of reduction in the number of features. In case of Magnatagatune, PSO-OPF has also allowed 100 % of reduction. It is important to shed light over that this reduction can provide a faster feature extraction procedure, with the compromise of similar and good recognition rates as in the original datasets, i.e., without feature selection.

Dataset	Technique	Acc	Time [s]	# features
GTZAN	PSO-OPF	98.78	300.8540	16772
GTZAN	GSA-OPF	98.78	603.2372	16776
GTZAN	HS-OPF	98.78	305.5086	16776
Magnatagatune	PSO-OPF	62.57	242.3991	37
Magnatagatune	GSA-OPF	62.57	475.2318	44
Magnatagatune	HS-OPF	62.57	244.5305	38

Table 4. Accuracy, time elapsed in seconds and number of selected features.

4. CONCLUSIONS

In this paper, we have addressed the problem of musical genre classification by means of OPF classifier, which has never been applied to this context up to date.

Experiments have been conducted in two rounds: in the former we have compared OPF with SVMs and Bayesian classifier in two public datasets (GTZAN and Magnatagatune), and in the latter we have applied recent OPF-based feature selection techniques in order to speed up the feature extraction process, and also to select the most important subset of features that lead to high recognition rates over an evaluating set.

In regard to the first round of experiments, all classifiers have obtained close and good recognition rates, being OPF faster for training and classification. It is important to highlight that this skill is very interesting in the context of very large multimedia datasets. We would like to stress the importance of user-friendly musical recommendation systems, in which training and classification phases need to be conducted in a feasible manner. In this context, OPF can be suitable for real-time retraining systems, in which new musical genres and songs can be added at any time to the dataset.

In addition, we have conducted an experiment to select the most representative features using algorithms recently developed, which have never been applied to this context to date. We have employed PSO-OPF, HS-OPF and GSA-OPF over GTZAN and Magnatagatune datasets, and the results seemed to be interesting, since one can reduce the number of features of both datasets without compromising the recognition rates. For future works, we intend to employ unsupervised OPF to the same task, as well as to use evolutionary-based feature selection algorithms.

5. ACKNOWLEDGMENT

The authors would like to thank FAPESP grants #2009/16206-1 and #2010/11676-7. We would also like to thank Professor George Tzanetakis from Computer Science Department, University of Victoria - Canada, for GTZAN dataset.

6. REFERENCES

[1] C. Allène, J.-Y. Audibert, M. Couprie, J. Cousty, and R. Keriven. Some links between min-cuts, optimal spanning forests and watersheds. In *Proceedings of the International Symposium on Mathematical Morphology*, volume 1, pages 253–264, São José dos Campos, 2007. Instituto Nacional de Pesquisas Espaciais (INPE).

[2] A. B. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):909–926, 2008.

[3] R. Collobert and S. Bengio. SVMTorch: support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160, 2001.

[4] E. Coviello, L. Barrigton, A. B. Chau, and G. R. G. Lanckeriet. Automatic music tagging with time series models. In *11th Internacional Society for Music Information Retrieval Conference*, 2010.

[5] P. L. Deepa and K. Suresh. Feature optimization for music genre classification based on support vector machine. In *Proceedings of the 10th National Conference on Technological Trends*, pages 315–318, 2009.

[6] E. Dellandrea, H. Harb, and L. Chen. Zipf, neural networks and svm for musical genre classification. In *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology*, pages 57–62, 2005.

[7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[8] A. X. Falcão, J. Stolfi, and R.A. Lotufo. The image foresting transform theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.

[9] Z.W. Geem. *Recent Advances In Harmony Search Algorithm*, volume 270 of *Studies in Computational Intelligence*. Springer, 2010.

[10] J. Kennedy and R. C. Eberhart. *Swarm intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[11] T. Lambrou, P. Kudumakis, R. Speller, M. Sandler, and A. Linney. Classification of audio signals using statistical features on time and wavelet transform domains. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages 3621–3624, may 1998.

[12] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1197–1206, New York, NY, USA, 2009. ACM.

[13] C. McKay and I. Fujinaga. Automatic genre classification using large high-level musical feature sets. In *International Conference on Music Information Retrieval*, pages 525–530, 2004.

[14] J. P. Papa, A. X. Falcão, and Celso T. N. Suzuki. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, 19(2):120–131, 2009.

[15] J. P. Papa, A. F. Pagnin, S. A. Schellini, A. A. Spadotto, R. C. Guido, M. P. Ponti Jr., G. Chiachia, and A. X. Falcão. Feature selection through gravitational search algorithm. In *Proceedings of the 36th International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, 2011. accepted for publication, details in http://wwwp.fc.unesp.br/˜papa/opficassp11.pdf.

[16] J. P. Papa, C. T. N. Suzuki, and A. X. Falcão. *LibOPF: A library for the design of optimum-path forest classifiers*, 2009. Software version 2.0 available at *http://www.ic.unicamp.br/˜afalcao/LibOPF*.

[17] C. C. O Ramos, J. P. Papa, A. N. Souza, and A. X. Falcão. What is the importance of selecting features for non-technical losses identification? In *Proceedings of the IEEE International Symposium on Circuits and Systems*, Rio de Janeiro, Brazil, 2011. accepted for publication, details in http://wwwp.fc.unesp.br/˜papa/opfiscas11.pdf.

[18] C. C. O. Ramos, A. N. Souza, and J. P. Papa. A novel algorithm for feature selection using harmony search and its application for non-technical losses detection. *Computers & Electrical Engineering*, 2011. (submitted).

[19] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi. GSA: A gravitational search algorithm. *Information Sciences*, 179(13):2232–2248, June 2009.

[20] L. M. Rocha, F. A. M. Cappabianco, and A. X. Falcão. Data clustering as an optimum-path forest problem with applications in image analysis. *International Journal of Imaging Systems and Technology*, 19(2):50–68, 2009.

[21] H. Soltau, T. Schultz, M. Westphal, and A. Waibel. Recognition of music types. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1137–1140, may 1998.

[22] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, jul 2002.

[23] George Tzanetakis and Perry Cook. Marsyas: A framework for audio analysis, 2000.

[24] C. Xu, N. C. Madagge, and X. Shao. Automatic music classification and summarization. In *IEEE Transactions on Speech and Audio Processing*, volume 13, pages 441 – 450, 2005.