A Markov Random Field Model for Combining Optimum-Path Forest Classifiers Using Decision Graphs and Game Strategy Approach

Moacir P. Ponti-Jr.¹, João Paulo Papa², and Alexandre L.M. Levada³

¹ Institute of Mathematical and Computer Sciences
University of São Paulo (ICMC/USP) at São Carlos, SP, Brazil
moacir@icmc.usp.br

<http://www.icmc.usp.br/~moacir>

² Department of Computing, UNESP — Univ Estadual Paulista at Bauru, SP, Brazil
papa@fc.unesp.br

<http://www.fc.unesp.br/~papa>

³ Computing Department, Federal University of São Carlos (DC/UFSCar) at
São Carlos, SP, Brazil
alexandre@dc.ufscar.br

<http://www.dc.ufscar.br/~alexandre>

Abstract. The research on multiple classifiers systems includes the creation of an ensemble of classifiers and the proper combination of the decisions. In order to combine the decisions given by classifiers, methods related to fixed rules and decision templates are often used. Therefore, the influence and relationship between classifier decisions are often not considered in the combination schemes. In this paper we propose a framework to combine classifiers using a decision graph under a random field model and a game strategy approach to obtain the final decision. The results of combining Optimum-Path Forest (OPF) classifiers using the proposed model are reported, obtaining good performance in experiments using simulated and real data sets. The results encourage the combination of OPF ensembles and the framework to design multiple classifier systems.

1 Introduction

The research on multiple classifiers systems comprises the creation of an ensemble of classifiers and also the combination of the decisions. The classifier ensembles are often produced through techniques such as bagging [1], boosting [4] and random subspace methods [6], producing classifiers using different subsets of samples and features. The combination (or fusion) of all decisions is often addressed using fixed rules [8] and also more complex methods.

If the classifiers to be combined provide only class labels as output, the majority voting is the approach commonly used to combine them. The limits of such schemes were investigated and the diversity aspect is currently under discussion, with the study of patterns of failure and success, and “good” and “bad” diversity [2].

In this study, we propose a framework for the combination of classifiers by creating a decision graph under a Markov Random Field (MRF) model. In order to compute the final decision we explored a majority voting scheme and a game strategy approach (GSA). This should not be confused with fusion of graphs or combination of structural pattern recognition classifiers. We rather developed a framework to combine classifiers using a random field model constructed from a graph of classifier decisions, as a way to capture the dependency between their outputs. Although another study used game theory to generalize rules for support-vector classifiers combination [5], the GSA combination algorithm proposed here is different, designed so that each classifier is seen as a player and each classifier decision (class label) is seen as a strategy.

This paper aims to combine ensembles of Optimum-Path Forest (OPF) classifiers [12], although the proposed model is general and can be used with a variety of other classifiers. The OPF technique models the feature space as a graph, using optimum-path algorithms to perform training and classification, and it outputs only class labels. The proposed combination is performed by training classifiers with distributed disjoint subsets and it is designed to improve accuracy and reduce running time. To improve speed, the method was developed so that it could be processed using parallel or distributed processors. Therefore, this paper presents contributions both on the study of a new model for combination of classifiers and on the development of a combination algorithm for the OPF classifier.

The paper is organized as follows. Section 2 introduces the OPF classifier used as basis for the ensembles. Section 3 describes the proposed combination framework, including the graph-based MRF model, the GSA combination algorithm and how the ensembles are created. Section 4 and 5 describe the experiments, results and discussion, respectively. Finally, the conclusions are presented in Section 6.

2 Optimum-Path Forest Classifier (OPF)

Papa et al. [12] introduced the idea of designing pattern classifiers based on optimum-path forest. The training samples are interpreted as the vertices of a graph, whose edges are defined by a given adjacency relation and weighted by some distance function. It is expected that samples from a same class 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. Since the true label of the training samples is known, key samples (prototypes) are identified in each class. Optimum paths are computed from the prototypes to each training sample, such that each prototype becomes the root of an optimum-path tree (OPT) composed by its most strongly connected samples. The labels of these samples are assumed to be the same of their root.

The training phase of OPF consists, basically, in finding prototypes and execute OPF algorithm to determine the OPTs rooted at them. Further, the test

phase essentially evaluates, for each test sample, which training vertex offered the optimum-path to its. The classification of each new sample t from test set O is performed based on the distance $d(s, t)$ between t and each training vertex $s \in T$ and on the evaluation of the following equation:

$$C_2(t) = \min\{\max[C_1(s), d(s, t)]\}, \forall s \in T. \quad (1)$$

Let $s^* \in T$ be the vertex s that satisfies this equation. It essentially considers all possible paths from S in the training graph T extended to t by an edge (s, t) , and label t with the class of s^* .

The OPF classifier has been demonstrated to have similar results to the ones obtained by Support Vector Machines, but running training much faster.

2.1 Learning Algorithm

Large datasets usually present redundancy, so it should be possible to estimate a reduced training set with the most relevant patterns for classification. The use of a training and an evaluation set has allowed OPF to learn relevant training samples from the classification errors in the evaluating, by swapping misclassified samples of the evaluating set and non-prototype samples of the training one during a few iterations. In this learning strategy, the training set remains with the same size and the classifier instance with the highest accuracy is selected to be tested in the unseen test set.

3 Random Field Model for Combination of Classifier Decisions

A classifier that outputs only class labels, for example OPF, allows only abstract-level methods for combination. In such cases, the most common approach is to apply a majority voting rule to obtain the final decision.

In this context, we propose a model similar to a Bayesian Network, using a directed graph of decisions for each object to be classified, where the vertices represent each classifier decision (class label), and the edges denote how each classifier influences the decision of the other ones. At first, this graph can assume any topology. To simplify the model, we are going to avoid cycles and use a tree with inverted directions in this paper. An example of such model is depicted in Figure 1(a), where the final decision is obtained through the propagation of the decisions through the graph, from the source vertices to the sink vertex.

This framework allows the definition of a more general model for combination of decisions. It can be specially useful when there is knowledge about how the decisions must influence each other. One can also use it at random, choosing different levels of vertices and edges.

One of the advantages of this framework is that it can be modeled through a Markov Random Field (MRF). Consider, for example, the situation depicted in Figure 1(a). In this case, we want to model the final decision, provided here by

Fig. 1. Examples a graph modelling the decision of classifiers, where the final decision is obtained at the “sink” (S) vertex: (a) 10 classifiers and arcs with different weights, (b) star graph for 9 classifiers

the base classifier “S”, in terms of the partial decisions given by other classifiers (competitors). In other words, we want to analyze how individual elements (classifiers) modify their behavior to conform to the behavior of other individuals in their vicinity. This is the typical scenario found in MRF models, used to study collective effects based on consequences of local interactions.

Since here each classifier gives as output a hard label (i.e., a discrete number that indicates the class of that sample) a suitable choice is the Potts MRF model.

3.1 The Potts MRF Model

The dependency structure between the classifiers output can be well modeled by a Potts MRF pairwise interaction. This model is both isotropic and stationary, which means that the spatial dependency parameter is the same for all directions and does not change as we move from one vertex to another. According to the Hammersley and Clifford theorem, a MRF model can be equivalently defined by a joint Gibbs distribution (global model) or by a set of local conditional density functions (LCDF). Due to mathematical and computational tractability, we will adopt a local description of the probability model by means of a set of LCDF’s. According to [14], the Potts model LCDF for a single observation is given by:

$$p(v_i = m | \eta_i, \beta) = \frac{\exp\{\beta U_i(m)\}}{\sum_{\ell=1}^M \exp\{\beta U_i(\ell)\}}, \quad (2)$$

where m is the label of the current vertex, $U_i(\ell)$ is the number of neighbors of the i -th vertex having label equal to ℓ , $\beta \in \mathbb{R}$ is the spatial dependency parameter that controls how strong is the influence of the neighboring vertices, and $\ell \in G$, with $G = \{1, 2, \dots, M\}$, where M represents the number of classes of the decision problem.

3.2 A Game Strategy Approach to Compute the Final Decision

It has been shown that MAP-MRF problems — where MAP stands for Maximum a Posteriori — do not allow closed-form solutions and, in order to approximate

the MAP estimator, combinatorial optimization algorithms are required to iteratively improve an initial solution [9]. In this study, we propose a combination scheme (GSAC) based on the Game Strategy Approach (GSA) algorithm [15], a non-cooperative game-theoretic algorithm that has been proved to approximate the MAP-MRF estimator by, given an initial solution, converging to the Nash Equilibrium.

In a n -person game, $I = \{1, 2, \dots, n\}$ denotes the set of players. Here, each player is a classifier, represented by a vertex in the graph. The idea is to build a decision graph where each vertex is given by a decision of an OPF classifier. The iterative process begins at the source vertices, propagating the labels to the sink, where the final decision will take place.

Each player has a set of pure strategies S_i . In this case, the set S_i is the same for all players, being defined as $S_i = \{1, 2, \dots, C\}$, where C is the number of classes. The game process consists of, at a given instant, each player choosing a strategy $s_i \in S_i$ in a way that a play $\mathbf{s} = (s_1, s_2, \dots, s_n)$ is yielded, and a payoff $H_i(\mathbf{s})$ is assigned to each player.

Algorithm 1 – GSAC: COMBINATION OF CLASSIFIER DECISIONS

INPUT: A labeled cooperation graph $G = (V, E)$ with $|E| = |V| - 1$ (tree) and $|V| = D$ where D is the number of classifiers, and $\alpha \in [0, 1]$

OUTPUT: Set of labels after final decision, L

AUXILIARY: The base classifier is the sink vertex and the labels of each vertex is the output of the respective classifier.

1. For $k = 1$ to D , do
2. Choose the strategy $l'_i \neq l_i^{(k)}$ that maximizes the payoff, i.e., $H_i(l_i^{(k)} || l'_i) = \max_{l'_i} H(l_i^{(k)} || l'_i)$
3. If $H_i(l_i^{(k)} || l'_i) \leq H_i(l_i^{(k)} || l_i^{(k)})$
4. $l_i^{(k+1)} = l_i^{(k)}$
5. Else
6. $l_i^{(k+1)} = l'_i$ with probability α
7. $l_i^{(k+1)} = l_i^{(k)}$ with probability $1 - \alpha$
8. Stop if the labeled graph is a Nash point, or repeat the iteration using the current play.
9. Return the final decision (the result of the combination of the classifiers).

Two fundamental hypothesis are assumed in GSA: first, the payoff of a player depends only on its own strategy and the strategies of its neighbors, and second, it is supposed that each player knows all possible strategies and the payoff given by each one of them. As GSA is based on non-cooperative game theory, each player selects independently his own strategy to maximize the local payoff. The solutions of a non-cooperative game are the *Nash points*, a condition achieved when none of the players can improve his expected payoff by unilaterally changing his strategy. In mathematical terms, a play $\mathbf{t}^* = (t_1^*, t_2^*, \dots, t_n^*)$ satisfies the Nash Equilibrium if [11]:

$$\forall i : H_i(\mathbf{t}^*) = \max_{s_i \in S_i} H_i(\mathbf{t}^* || \mathbf{t}) \quad (3)$$

where $\mathbf{t}^* || \mathbf{t}$ is the play obtained by replacing $\mathbf{t}^*$ by $\mathbf{t}$. It has been shown that Nash points always exist in non-cooperative games with pure or mixed strategies [11]. The GSA fundamentals are based on a major result derived by [15] which

states that given a initial play, the GSA algorithm converges to a *Nash point* in a finite number of steps.

The proposed GSAC algorithm for combining classifiers is given by Algorithm 1. Some considerations must be provided. First, the local payoff is calculated as the local energy, that is, $H_i(l_i) = \beta U(l_i)$, which is directly proportional to the probability value $p(v_i = l_i | \eta_i, \beta)$. Another issue is regarding the $\alpha \in [0, 1]$ parameter existing in GSAC, which controls the probability of acceptance of new strategies. If $\alpha = 1$, we always accept a better strategy (the algorithm becomes deterministic), otherwise even if there is an improved strategy, we might not accept it in that iteration (non-deterministic behavior).

3.3 Building an OPF Classifier Ensemble

To create an ensemble of OPF classifiers, we used disjoint training subsets [13]. Given a fixed number D of subsets, the algorithm chooses random samples, without replacement, from the original training set T until D subsets are created. The samples are taken so that each subset will contain approximately the same number of objects per class. Each subset is then used to train a classifier. The procedure is described in Algorithm 2, where *OPF_learn* and *OPF_classify* corresponds to OPF learning and classification procedures, respectively.

As described in Section 2, the OPF has a fast learning algorithm to rebuild training set using an evaluation set in order to improve accuracy of each classifier, and, therefore, is expected to improve accuracy of the final decision. It has a behavior similar to a boosting algorithm. Moreover, the OPF training algorithm has computational complexity of $\Theta(N^2)$, in which N denotes the training set size, and therefore, it is expected to run faster for k training sets of size X then for a larger one with size $N = k \times X$.

Algorithm 2 – OPFCD: OPF COMBINATION OF DISTRIBUTED DISJOINT SETS

INPUT: Training data set T of size N with correct labels $\omega_j \in \Omega$, $j = \{1, \dots, C\}$ for C classes, the evaluation set V , the set of objects to be classified O (test set), the number of disjoint subsets D , the number of samples of each class $P = \{p_1, \dots, p_C\}$, and the OPF algorithm *OPF_learn* and *OPF_classify*

OUTPUT: Set of labels after final decision, L

AUXILIARY: The number of objects on each subset M , training subsets K_i , classifiers E_i , and objects labeled by each classifier I_i , where $i = \{1, \dots, D\}$

1. $M \leftarrow \lfloor N/D \rfloor$
2. For each subset i , ($\forall i = 1..D$), do
3. For each class j , ($\forall j = 1..C$), do
4. $\hookrightarrow$ Select randomly $(p_j/D) \times M$ samples of class j from T without replacement and store them in K_i
5. $E_i \leftarrow \text{OPF_learn}(K_i, V)$
6. $I_i \leftarrow \text{OPF_classify}(O, E_i)$
7. $L \leftarrow \text{Vote}(I_i)$
8. Return L

4 Experiments

4.1 Data

The experiments were carried out using four simulated and five real data sets. The simulated data sets were built so that the classes were partially overlapped.

Table 1. Simulated data sets characteristics, where C is the number of classes and F is the number of features

NAME	SIZE	C	F	TYPE
B2-2D	1,000	2	2	BANANA-SHAPE
B2-3D	2,000	4	2	BANANA-SHAPE
G4-2D	1,000	2	4	GAUSSIAN
G4-3D	2,000	4	4	GAUSSIAN

Table 2. Real data sets characteristics, where C is the number of classes and F is the number of features

NAME	SIZE	C	F
NTL	8,067	2	4
COREL	1,000	10	150
WINE	178	3	13
KDD-1999	380,271	3	9
ACTIVITY	164,860	11	6

The Gaussian data sets have classes with different covariance matrices. The banana-shape data set with 3 features was generated with a higher variance when compared to the one with 2 features. The project web page¹ contains the code to generate the simulated data using PRTools² and an implementation of GSAC. Table 1 shows the characteristics of simulated data sets and Table 2 of real data sets.

NTL is a dataset of an electric power company for identification of legal and illegal profiles of industrial costumers. *COREL* is a subset of an image database including 10 classes with SIFT features [10]. *Wine* is a small data set with results of chemical analysis of wines of three different cultivars [3]. *KDD-1999* models a network intrusion detector, capable of distinguishing intrusions or attacks, and normal connections [3]. *Activity* is the “Localization Data for Person Activity Data Set” to identify what is the current activity of a person based on sensors [7].

4.2 Settings and Implementation

We conducted the experiments as follows: the data sets were partitioned in three sets, 10% for training, 5% for evaluating and 85% for testing.

There are three exceptions in the settings due to the number of samples per class and the size of some real data sets. The exceptions are: i) *COREL* data set with 15% for training and 5% for evaluating, ii) *Wine* data set with 30% for training and 15% for evaluating, and iii) *KDD-1999* data set with 1% for training and 0.5% for evaluating.

¹ <http://sites.google.com/site/projectensembles/>

² <http://www.prtools.org/>

The whole training set was used to train the OPF “single classifier”, and the subsets obtained from the training set were used to train several classifiers and build the ensembles. We performed experiments using from 3 to 9 training set partitions (disjoint training subsets) obtained by the OPFcd (Algorithm 2).

In these series of experiments, the decision graph was modeled using a directed star graph, since it is the simplest possible model and can show the robustness of our proposed schema for classifier combination. Figure 1(b) shows an example of the model for an ensemble of 9 classifiers. On each experiment, the sink vertex was randomly chosen from the ensemble and all the other classifiers were connected to the sink node to create the decision graph. No other relations were specified. The GSAC algorithm (Algorithm 1) was then applied to obtain the final decision. We also report the results of a simple majority vote applied to the classifier ensemble in order to compare with the proposed method.

The parameters for the GSAC algorithm was experimentally obtained and defined as $\beta = 0.8$ (spatial dependency parameter) and $\alpha = 0.9$ (probability of acceptance of new strategies).

All experiments were repeated 10 times on different training sets and partitions. The average and standard deviation results are presented. In order to verify the significant differences between groups of results, we performed a two-tailed t-test for samples with unequal variances.

5 Results and Discussion

The main results are shown in Table 3. It displays the classification errors for the data sets using the naive OPF in the first column, and the errors for ensembles of 3, 5, 7, 9 and 42 OPF classifiers using Vote and GSA algorithms to compute the final decision. The results for the ensembles with 4, 6 and 8 classifiers were omitted due to space limitations, since the results were similar to the displayed ones. The ensemble with 42 classifiers was created using the outputs of all classifiers produced by the number of partitions from 3 to 9.

The combination approach using disjoint data sets improved the results for all data sets except for the *Activity*. It was due to a high decrease in performance as less samples are used for training in this data set — a bad scenario for combination of classifiers. However, the combination of all 42 decisions improved the results of all data sets. An improvement on the results was observed for both simulated and real problems in which the single classifier obtained higher error rates, for example *Wine* and *G4-2D*, and also for those with a lower error, for example *KDD-1999* and *B2-2D*. By using an ensemble of 42 OPF classifiers it was possible to achieve an accuracy of 99.95% for the *KDD-1999* data set.

The GSAC results were similar to the Vote method. It is possibly due to the characteristics of the GSAC algorithm under a star graph model, in which the final decision will be dominated by the mode of the classes in the neighborhood. It works as a majority vote rule using a random approach for tie-breaking when the parameters are defined as $\alpha = 1$ and $\beta = 1$. The proposed method is, however, more flexible, allowing the design of different graph topologies and an adjustment on the parameters α and β , as performed in this study.

Although the differences between the proposed and competing method were not significant for all data sets, the GSAC was able to improve the performance using different sizes of classifier ensembles, and, therefore, different number of training set partitions. It also showed a slightly better performance when using the graph with all 42 decisions. We expect that more complex systems can take advantage of the GSAC characteristics.

Table 3. Classification errors for naive OPF and combination of classifiers using Vote and GSA. Results in boldface represents significant improvement for a $p < 0.05$. * the ensemble with 42 classifiers represents the decisions of all classifiers trained with number of partitions from 3 to 9.

DATA SET / COMB.		NUMBER OF CLASSIFIERS IN THE ENSEMBLE				
NAIVE	OPF METHOD	3	5	7	9	42*
B2-2D	VOTE	5.5±1.0	5.5±1.0	6.0±0.7	4.4±0.7	1.4±0.4
7.1±0.9	GSA	5.6±0.9	5.4±0.8	5.6±0.9	4.4±0.9	1.1±0.5
B2-3D	VOTE	21.2±1.5	17.1±1.7	18.4±2.4	16.4±2.1	5.6±1.1
21.6±1.5	GSA	20.6±1.7	17.0±1.5	18.3±2.4	16.4±2.2	4.5±0.7
G4-2D	VOTE	25.3±3.5	21.5±2.7	23.7±3.7	19.5±1.8	6.3±0.3
26.4±2.5	GSA	23.6±2.2	21.6±2.8	22.0±3.1	19.5±1.9	6.0±0.2
G4-3D	VOTE	14.4±1.3	14.2±1.1	13.5±0.9	12.3±1.0	9.6±1.2
14.2±1.0	GSA	14.2±1.5	13.0±0.7	12.6±1.0	12.3±0.9	9.5±1.1
NTL	VOTE	8.2±1.4	8.4±1.2	9.1±2.9	9.5±3.5	5.4±0.2
10.1±0.4	GSA	7.7±1.1	8.2±1.2	8.9±1.7	10.2±3.3	5.3±0.1
COREL	VOTE	16.4±1.0	15.9±0.8	15.5±1.6	16.3±2.0	12.3±0.3
18.1±0.6	GSA	16.4±1.2	15.1±0.8	15.0±1.2	15.3±1.3	12.2±0.5
WINE	VOTE	28.6±2.0	28.0±1.5	29.2±2.0	31.1±2.0	17.7±3.6
30.4±1.8	GSA	27.8±3.0	27.1±1.6	27.0±1.7	30.1±2.6	15.6±3.3
KDD-1999	VOTE	0.16±0.02	0.25±0.03	0.24±0.04	0.14±0.02	0.04±0.01
0.21±0.05	GSA	0.17±0.02	0.21±0.02	0.19±0.02	0.15±0.03	0.05±0.01
ACTIVITY	VOTE	30.9±2.0	31.1±1.6	31.4±0.7	31.8±1.7	22.1±1.6
30.0±1.2	GSA	30.8±1.9	30.9±1.2	31.4±1.1	29.8±1.4	21.3±1.9

6 Conclusions

Using a simple star graph to model the classifier system, the results showed to decrease the classification error. It was interesting to observe such results even with the combination of few classifiers obtained with the OPFcd method, encouraging the use of OPF classifiers to build ensembles. Since the OPF training algorithm is $\Theta(N^2)$ as discussed in Section 3.3, the combination of OPF ensembles is also a faster procedure when compared with the single OPF, specially when it is carried out with few classifiers, as proposed in this paper using disjoint training subsets.

The behavior of the proposed combination framework was similar to the behavior of known methods for combination of ensemble of classifiers. We believe it is a consequence of the simple star graph model used in the experiments, and that it can change when a more complex model with several levels of decisions

are used to model the multiple classifier system. The proposed model allow the design of complex combination schemes, allowing the use of many known methods for MRF and optimization to solve the classification problem. Therefore it has the potential to overcome difficult problems when there is knowledge about how the classifiers decisions should influence each other. A more detailed analysis of how the decision graphs can be designed for different machine learning problems is a point left for future studies.

References

1. Breiman, L.: Bagging predictors. *Machine Learning Journal* 2(24), 123–140 (1996)
2. Brown, G., Kuncheva, L.I.: “Good” and “Bad” diversity in majority vote ensembles. In: El Gayar, N., Kittler, J., Roli, F. (eds.) *MCS 2010. LNCS*, vol. 5997, pp. 124–133. Springer, Heidelberg (2010)
3. Frank, A., Asuncion, A.: UCI machine learning repository (2010), <http://archive.ics.uci.edu/ml>
4. Freund, T.: Boosting: a weak learning algorithm by majority. *Information and Computation* 121(2), 256–285 (1995)
5. Georgiou, H., Mavroforakis, M., Theodoridis, S.: A game-theoretic approach to weighted majority voting for combining SVM classifiers. In: Kollias, S.D., Stafylopatis, A., Duch, W., Oja, E. (eds.) *ICANN 2006. LNCS*, vol. 4131, pp. 284–292. Springer, Heidelberg (2006)
6. Ho, T.: The random subspace method for constructing decision forests. *IEEE Trans. Pattern Analysis and Machine Intelligence* 20(8), 832–844 (1998)
7. Kaluza, B., Mirchevska, V., Dovgan, E., Lustrek, M., Gams, M.: An agent-based approach to care in independent living. In: *Int. Joint Conf. on Ambient Intelligence (AML 2010)*, Malaga, Spain (2010)
8. Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. *IEEE Trans. Pattern Analysis and Machine Intelligence* 20(3), 226–239 (1998)
9. Levada, A.L.M., Mascarenhas, N.D.A., Tannús, A.: A novel MAP-MRF approach for multispectral image contextual classification using combination of suboptimal iterative algorithms. *Pattern Recognition Letters* 31(13), 1795–1808 (2010)
10. Li, J., Wang, J.Z.: Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Trans. Pattern Analysis and Machine Intelligence* 25(9), 1075–1088 (2003)
11. Nash, J.F.: Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences* 36(1), 48–49 (1950)
12. Papa, J., Falcão, A.X., Suzuki, C.T.N.: Supervised pattern classification based on optimum-path forest. *Int. J. Imaging Systems and Technology* 19(2), 120–131 (2009)
13. Ponti Jr., M.P., Papa, J.P.: Improving accuracy and speed of optimum-path forest classifier using combination of disjoint training subsets. In: Sansone, C. (ed.) *MCS 2011. LNCS*, vol. 6713, pp. 237–248. Springer, Heidelberg (2011)
14. Yamazaki, T., Gingras, D.: Image classification using spectral and spatial information based on mrf models. *IEEE Trans. on Image Processing* 4(9), 1333–1339 (1995)
15. Yu, S., Berthod, M.: A game strategy approach for image labelling. *Computer Vision and Image Understanding* 61(1), 32–37 (1995)