Multimodal Pattern Recognition Through Particle Swarm Optimization

Fábio A. Faria
Institute of Computing
University of Campinas
Campinas - Brazil
fabio.faria@students.ic.unicamp.br

João P. Papa
Department of Computing
São Paulo State University
Bauru - Brazil
papa@fc.unesp.br

Ricardo S. Torres
Institute of Computing
University of Campinas
Campinas - Brazil
rtorres@ic.unicamp.br

Alexandre X. Falcão
Institute of Computing
University of Campinas
Campinas - Brazil
afalcao@ic.unicamp.br

***Abstract*—In this paper we introduce the idea of descriptor combination by Particle Swarm Optimization and its applications for classification purposes using a recently pattern recognition technique called Optimum-Path Forest (OPF), which interprets the samples of the dataset as the nodes of a given graph, and each arc is weighted by the distance between the corresponding nodes. The method combines different color descriptors for classification in the COREL dataset to further weight the OPF graph arcs by these combined similarities. We show that our proposed approach can provides better accuracies than the individual ones obtained by each descriptor.**

***Index Terms*—Optimum-Path Forest, Descriptor Combination, Particle Swarm Optimization, Pattern Recognition**

I. Introduction

Multimodal pattern recognition has become a very promising research area, mainly due to its power of describing datasets. However, it is usual to find researches that validate their feature extraction algorithms in a restrict domain of datasets. Shape features extractors [1], for instance, are commonly evaluated in the MPEG-7 Shape Dataset [2], which is composed by binary objects for shape analysis. Regarding color features, the most commonly used dataset is COREL [3]. Color histograms [4] and Color Coherence Vectors [5], for instance, are the most actively used feature extraction algorithms and are constantly validated on COREL dataset. Several texture analysis approaches have been carried out in Brodatz Dataset [6]. Montoya et al. [7], for instance, evaluated their new wavelet-based approach for scale and rotation texture invariant recognition in this dataset.

Given that humans visually recognizes objects using different features, they tend to distinguish images based on their shape, color and texture properties. In this context, some works have been investigating the problem of combining and selecting features. However, the main problem of combining features of different types (such as color, texture, and shape), and even features of the same type, relies on the fact that some features can not be treated in the same metric space. The BAS shape feature extractor [1], for instance, is composed by the computation of three statistical moments of the k-curvature at each shape point. This means that the features extracted from BAS for each image are not correlated to each other. Hence a common distance metric, such as the Euclidean distance, can not be used for computing BAS distances, and so more complex distance functions need to be used [8]. In the opposite way, color histograms represent the image as a whole, and the Euclidean distance can be used to computed the distance between two different color histograms.

Recently, a novel pattern recognition technique called Optimum-Path Forest (OPF) [9] was developed, and has been demonstrated to be superior than Artificial Neural Networks (ANN) and similar to Support Vector Machines (SVM), but much faster. The OPF models the pattern recognition as a graph partitioning problem, in which each class is represented by an Optimum-Path Tree (OPT), and a sample that belongs to a given OPT is more strongly connected to its root than any other root in the forest. Given that the arcs of this graph are weighted by the distance between their corresponding nodes (feature vectors), we propose in this work to combine different descriptors (i.e., different similarity values) by Particle Swarm Optimization (PSO) [10]. PSO is a evolutionary approach that has been extensively used for several optimization problems. The main idea is to use this combined distance to weight the OPF graph edges. In such a way, each arc weight will represent a similarity value between two samples that takes into account different features extraction algorithms. As far as we know, we are the first to propose a descriptor combination using Particle Optimization and other main contribution of our paper is that we are the pioneer into applying this approach for the OPF algorithm.

The remainder of this paper is organized as follows. Sections II and III present, respectively, the OPF and PSO methods. Section IV discuss the experimental results and Section V states conclusions and future works.

II. Optimum-path forest classifier

Let Z_1 and Z_2 be the training and and test sets with $|Z_1|$ and $|Z_2|$ samples such as points or image elements (e.g., pixels, voxels, shapes or texture information). Let $\lambda(s)$ be the function that assigns the correct label i, $i = 1, 2, \ldots, c$, from class i to any sample $s \in Z_1 \cup Z_2$. Z_1 is a labeled set used to the design of the classifier and Z_2 is used to assess the performance of classifier and it is kept unseen during the project.

Let $S \subset Z_1$ be a set of prototypes of all classes (i.e., key samples that best represent the classes). Let v be an

algorithm which extracts n attributes (color, shape or texture properties) from any sample $s \in Z_1 \cup Z_2$ and returns a vector $\vec{v}(s) \in \Re^n$. The distance $d(s,t)$ between two samples (images in this paper), s and t, is the one between their feature vectors $\vec{v}(s)$ and $\vec{v}(t)$. One can use any valid metric (e.g., Euclidean) or a more elaborated distance algorithm. In this context, we introduce the idea of *descriptor* [11], which can be defined as a pair $D = (\vec{v}, d)$, in which d is a distance function between samples. A descriptor claims that every feature vector needs its related distance.

Our problem consists of using S, (v, d) and Z_1 to project an optimal classifier which can predict the correct label $\lambda(s)$ of any sample $s \in Z_2$. 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 [12].

A. Training

Let (Z_1, A) be a complete graph whose nodes are the training samples and any pair of samples defines an arc in $A = Z_1 \times Z_1$. The arcs do not need to be stored and so the graph does not need to be explicitly represented. A path is a sequence of distinct samples $\pi_t = \langle s_1, s_2, \ldots, t\rangle$ with terminus at a sample t. A path is said *trivial* if $\pi_t = \langle t\rangle$. We assign to each path π_t a cost $f(\pi_t)$ given by a connectivity function f. A path π_t is said optimum if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t. We also denote by $\pi_s \cdot \langle s,t\rangle$ the concatenation of a path π_s and an arc (s,t).

We will address the connectivity function $f_{\max}$, given by:

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

such that $f_{\max}(\pi_s \cdot \langle s,t\rangle)$ computes the maximum distance between adjacent samples along the path $\pi_s \cdot \langle s,t\rangle$. We wish to assign to every sample $t \in Z_1$ an optimum path $P^*(t)$ from the set $S \subset Z_1$ of prototypes, whose minimum cost $C(t)$ is:

$$C(t) = \min_{\forall \pi_t \in (Z_1, A)} \{f_{\max}(\pi_t)\}. \quad (2)$$

The minimization of C is computed by Algorithm 1 (described afterwards), called OPF algorithm, which is an extension of the general image foresting transform (IFT) algorithm [12] from the image domain to the feature space, here specialized for $f_{\max}$. This process assigns one optimum path from S to each training sample t in a non-decreasing order of minimum cost, such that the graph is partitioned into an optimum-path forest P (a function with no cycles which assigns to each $t \in Z_1 \backslash S$ its predecessor $P(t)$ in $P^*(t)$ or a marker nil when $t \in S$. The root $R(t) \in S$ of $P^*(t)$ can be obtained from $P(t)$ by following the predecessors backwards along the path, but its label is propagated during the algorithm by setting $L(t) \leftarrow \lambda(R(t))$.

In Algorithm 1, Lines $1-3$ initialize maps and insert prototypes in Q. The main loop computes an optimum path from S to every sample s in a non-decreasing order of minimum cost (Lines $4-11$). At each iteration, a path of minimum cost $C(s)$ is obtained in P when we remove its last node s from Q (Line 5). Ties are broken in Q using first-in-first-out policy. That is, when two optimum paths reach an ambiguous sample s with the same minimum cost, s is assigned to the first path that reached it. Note that $C(t) > C(s)$ in Line 6 is false when t has been removed from Q and, therefore, $C(t) \neq +\infty$ in Line 9 is true only when $t \in Q$. Lines $8-11$ evaluate if the path that reaches an adjacent node t through s has a lower cost than the current path with terminus t and update the position of t in Q, $C(t)$, $L(t)$ and $P(t)$ accordingly.

Algorithm 1: – OPF ALGORITHM

INPUT: *A training set Z_1, λ-labeled prototypes $S \subset Z_1$ and the pair (v, d) for feature vector and distance computations.*
OUTPUT: *Optimum-path forest P, cost map C and label map L.*
AUXILIARY: *Priority queue Q implemented as a binary heap and cost variable cst.*

1. For each $s \in Z_1 \backslash S$, set $C(s) \leftarrow +\infty$.
2. For each $s \in S$, do
3. $\quad C(s) \leftarrow 0$, $P(s) \leftarrow nil$, $L(s) \leftarrow \lambda(s)$, and insert s in Q.
4. While Q is not empty, do
5. $\quad$ Remove from Q a sample s such that $C(s)$ is minimum.
6. $\quad$ For each $t \in Z_1$ such that $t \neq s$ and $C(t) > C(s)$, do
7. $\quad\quad$ Compute $cst \leftarrow \max\{C(s), d(s,t)\}$.
8. $\quad\quad$ If $cst < C(t)$, then
9. $\quad\quad\quad$ If $C(t) \neq +\infty$, then remove t from Q.
10. $\quad\quad\quad$ $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$ and $C(t) \leftarrow cst$.
11. $\quad\quad\quad$ Insert t in Q.

We say that S^* is an optimum set of prototypes when Algorithm 1 minimizes the classification errors in Z_1. S^* can be found by exploiting the theoretical relation between minimum-spanning tree (MST) and optimum-path tree for $f_{\max}$ [9]. By computing a MST in the complete graph (Z_1, A), we obtain a connected acyclic graph whose nodes are all samples of Z_1 and arcs are undirected and weighted by the distances d between adjacent samples. The MST 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, and 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. By removing the arcs between different classes, their adjacent samples become prototypes in S^* and Algorithm 1 can compute an optimum-path forest in Z_1.

B. 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:

$$C(t) = \min\{\max\{C(s), d(s,t)\}\}, \ \forall s \in Z_1. \quad (3)$$

Let node $s^* \in Z_1$ be the one that satisfies Equation 3 (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 $L(s^*) \neq \lambda(t)$.

III. Particle Swarm Optimization

Recently, several applications have used evolutionary techniques as heuristic methods to find optimal or quasi-optimal solutions. A particular attention has been devoted to Particle Swarm Optimization (PSO), due to its simplicity and effectiveness. Basically, PSO is an algorithm modeled on swarm intelligence that finds a solution in a search space based on the social behavior dynamics [10]. Each possible solution of the problem is modeled as a particle in the swarm that imitates its neighbor based on a fitness function.

Other definitions consider PSO as a stochastic and population-based search algorithm, in which the 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). Taking into account this information, each particle has the ability to imitate the other ones that give to it the best local and global maxima. This process simulates the 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 [10]: (i) evaluating, (ii) comparing and (iii) imitating. Each particle can evaluate other ones into 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 swarm is modeled as multidimensional space $\Re^m$, in which each particle $p_i = (\vec{x_i}, \vec{v_i}) \in \Re^m$ 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 (Figure 1). After defining the swarm size, i.e., the number of particles, each one of them is initialized with random values of 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 into the swarm. This process is repeated until some convergence criterion be reached. The updated position and velocity equations of the particle p_i in the simplest form that govern the PSO are, respectively, given by

$$\vec{v_i} = w\vec{v_i} + c_1 r_1(\widehat{x}_i - \vec{x_i}) + c_2 r_2(\widehat{s} - \vec{x_i}) \quad (4)$$

and

$$x_i = x_i + v_i, \quad (5)$$

where w is the inertia weight that controls the interaction power between 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. The whole PSO algorithm is given

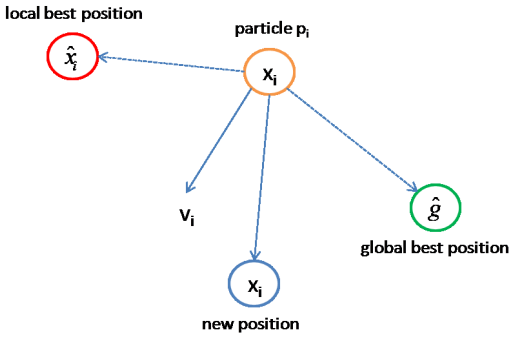

Fig. 1. PSO geometrical interpretation.

below. For the swarm size and number of iterations, we used $n = 100$ and $T = 10$, respectively. Regarding the inertia weight we used $w = 0.5$.

Algorithm 2: – PSO Algorithm

Input: *Inertia weight value w, PSO swarm size n, fitness function f, dimension m and number of iterations T and c_1, c_2 values.*
Output: *Global best position $\widehat{s}$.*
Auxiliary: *Fitness vector $\vec{a}$ with size n and variables r_1, r_2, c_1, c_2, k, tmp, $maxfit$, $globalfit$ and $maxindex$.*

```
1.  For each particle p_i (∀i = 1,...,n), do.
2.      For each dimension j (∀j = 1,...,m), do.
3.          x_{i,j} ← uniform random value
4.          v_{i,j} ← 0
5.      a_i ← −∞
6.  globalfit ← −∞
7.  For each iteration t (t = 1,...,T), do.
8.      For each particle p_i (∀i = 1,...,n), do.
9.          tmp ← f(p_i)
10.         If (tmp > a_i), then.
11.             a_i ← tmp
12.             For each dimension j (∀j = 1,...,m), do.
13.                 x̂_{i,j} ← x_{i,j}
14.     [maxfit, maxindex] ← max(a)
15.     If(maxfit > globalfit), then.
16.         globalfit ← maxfit
17.         For each dimension j (∀j = 1,...,m), do.
18.             ŝ_j ← x_{maxindex,j}
19.     For each particle p_i (∀i = 1,...,n), do.
20.         For each dimension j (∀j = 1,...,m), do.
21.             r_1 ← uniform random value
22.             r_2 ← uniform random value
23.             k ← c_2 * r_2 * (ŝ_{i,j} − x_{i,j})
24.             v_{i,j} ← w * v_{i,j} + c_1 * r_1 * (x̂_{i,j} − x_{i,j}) + k
25.             x_{i,j} ← x_{i,j} + v_{i,j}
```

The loop (Lines $1-5$) initializes the particle parameters (Lines $2-4$) and its local fitness value (Line 5). The swarm global fitness is initialized in Line 6. The main loop (Lines $7-25$) is the core of PSO algorithm: the first inner loop (Lines $8-13$) calculates, for each particle, its fitness function (Line 9) and evaluates whether this value is better than the best one recorded by the particle (Lines $10-13$). In the positive case, the fitness value of particle p_i (Line 11) and its best position, i.e., local maxima, are updated in Line 12. Line 14 retrieves the best (maximum) fitness value and its position and stores them in the variables $maxfit$ and $maxindex$, respectively. If the obtained best fitness value is greater than the global fitness,

the last one is updated with former value (Lines $17-18$). The global best particle is updated with the same position of the particle that gave the maximum fitness value in the actual iteration (Line 16). Finally, the second inner loop (Lines $19-25$) is responsible for updating each particle's position and velocity with Equations 4 and 5 (Lines $23-25$).

IV. Experimental results

In this section we describe the experiments regarding the descriptor combination by PSO, as well its application for the Optimum-Path Forest classifier. We used the COREL dataset [3] with the following color descriptors: BIC [13], GCH [4] and LUCOLOR. We also used the LAS texture descriptor [14]. The Corel dataset used here was extracted from a database containing 20.000 images from the GALLERY Magic - Stock Photo Library2, and our subset is composed by 3.906 images, in which we have 85 classes of images and the number of images per class varies from 7 to 98.

For each descriptor, we computed its similarity matrices (i.e., the distance between all pairs of samples), which were used as input to the OPF algorithm for classification purposes. We divided the dataset into three sets: 25% for training, 25% for evaluating and 50% for testing. The former sets (training and evaluating) were used to guide the PSO algorithm, i.e., the OPF was trained in the training set and validated in the evaluating one. The accuracy obtained in the latter was used as the fitness function for PSO. In such a way, the value $f(p_i)$ in Line 9 of Algorithm 2 corresponds to the OPF accuracy over the evaluating set. In other words, the PSO task was to find the descriptor combination that maximizes the OPF accuracy over the evaluating set.

Mathematically speaking, let's say that D^* is the best composite descriptor that we could obtain. Our descriptor combination can be represented as a linear combination of the aforementioned descriptors, given by

$$D^* = \lambda_1 D_{BIC} + \lambda_2 D_{LAS} + \lambda_3 D_{GCH} + \lambda_4 D_{LUCOLOR}. \quad (6)$$

In such a way, we used the PSO methodology to find $\lambda^* = \{\lambda_1, \lambda_2, \lambda_3, \lambda_4\}$ that maximizes the OPF accuracy over the evaluating set. After that, we use λ^* to transform the original dataset to this new one and apply OPF for classifying samples from the test set. The whole process is repeated 10 times in order to compute the mean accuracies values. Table I displays the results for each descriptor individually, and also for several combinations of them.

We can see that any descriptor combination by PSO and its validation using OPF can outperform the individual accuracy of each one. The best results were found by combining all descriptors (bolded line in the above table).

TABLE I
Color descriptors performance.

Descriptor	Accuracy	Classifier
BIC	72.69%	OPF
LAS	60.15%	OPF
GCH	65.49%	OPF
LUCOLOR	62.77%	OPF
LAS+GCH+LUCOLOR	66.31%	OPF+PSO
BIC+LAS	73.38%	OPF+PSO
LAS+GCH	68.61%	OPF+PSO
BIC+LAS+GCH+LUCOLOR	**73.46%**	**OPF+PSO**

V. Conclusions

We proposed in this work the descriptor combination instead of simple features selection/combination. Our proposed schema was carried out by the Particle Swarm Optimization and validated in the Optimum-Path Forest algorithm in the COREL image dataset. We used here 3 color and 1 texture descriptors, which had their similarity matrices combined in order to maximize the OPF accuracy over an evaluating set. As our proposed methodology can be described by a linear combination of variables, the PSO was used to find them for further dataset transformation. Our results demonstrated that the proposed methodology outperformed the individual accuracy of each descriptor. Thus, it can be efficiently used for classification purposes.

For future works, we intend to combine descriptors from different contexts, i.e., shape, color and texture. Another idea is to validate our approach for Content-Based Image Retrieval.

References

[1] N. Arica and F. T. Y. Vural, "BAS: A Perceptual Shape Descriptor Based on the Beam Angle Statistics," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1627–1639, June 2003.

[2] MPEG-7, "Mpeg-7: The generic multimedia content description standard, part 1," *IEEE MultiMedia*, vol. 09, no. 2, pp. 78–87, 2002.

[3] Corel Corporation, "Corel stock photo images," -, http://www.corel.com.

[4] M. Swain and D. Ballard, "Color Indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991.

[5] G. Pass, Zabih R, and J. Miller, "Comparing images using color coherence vectors," in *MULTIMEDIA'96: Proceedings of the 4th ACM International Conference on Multimedia*, 1996, pp. 65–73.

[6] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*, Dover, New York, 1966.

[7] J.A. Montoya-Zegarra, J.P. Papa, N.J. Leite, R.S. Torres, and A.X. Falcão, "Learning how to extract rotation-invariant and scale-invariant features from texture images," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, pp. 1–16, 2008.

[8] Y. P. Wang and T. Pavlidis, "Optimal correspondence of string subsequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 11, pp. 1080–1087, 1990.

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

[10] J. Kennedy and R.C. Eberhart, *Swarm Intelligence*, Morgan Kaufman, 2001.

[11] Ricardo da Silva Torres, A.X. Falcão, M.A. Gonçalves, J.P. Papa, B. Zhang, W. Fan, and E.A. Fox, "A genetic programming framework for content-based image retrieval," *Pattern Recognition*, vol. 42, no. 2, pp. 283–292, 2009.

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

[13] R.O. Stehling, M.A. Nascimento, and A.X. Falcão, "A compact and efficient image retrieval approach based on border/interior pixel classification," in *Proc. of the 2002 ACM CIKM International Conference on Information and Knowledge Management*, 2002, pp. 102–109.

[14] B. Tao and B.W. Dickinson, "Texture recognition and image retrieval using gradient indexing," *Journal of Visual Communication and Image Representation*, vol. 11, no. 3, pp. 327–342, September 2000.