Applied Soft Computing 9 (2009) 632–640

Contents lists available at ScienceDirect

Applied Soft Computing

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

Hyperbox clustering with Ant Colony Optimization (HACO) method and its application to medical risk profile recognition

G.N. Ramos [a,*], Y. Hatakeyama [b], F. Dong [a], K. Hirota [a]

[a] Department of Computational Intelligence and Systems Science, Tokyo Institute of Technology, Yokohama, Japan
[b] Center of Medical Information Science, Medical School, Kochi University, Nankoku city, Japan

ARTICLE INFO

Article history:
Received 24 August 2006
Received in revised form 31 August 2008
Accepted 11 September 2008
Available online 26 September 2008

Keywords:
Clustering
Ant colony
Hyperbox
Optimization
Pattern recognition

ABSTRACT

A clustering method, called HACO (Hyperbox clustering with Ant Colony Optimization), is proposed for classifying unlabeled data using hyperboxes and an ant colony meta-heuristic. It acknowledges the topological information (inherently associated to classification) of the data while looking in a small search space, providing results with high precision in a short time. It is validated using artificial 2D data sets and then applied to a real medical data set, automatically extracting medical risk profiles, a laborious operation for doctors. Clustering results show an improvement of 36% in accuracy and 7 times faster processing time when compared to the usual ant colony optimization approach. It can be further extended to hyperbox shape optimization (fine tune accuracy), automatic parameter setting (improve usability), and applied to diagnosis decision support systems.

1. Introduction

The Ant Colony Optimization (ACO) meta-heuristic [4,5] uses a population of agents (ants) guided by an autocatalytic process directed by a greedy force for discrete combinatorial optimization problems. Previous studies applied optimization techniques to clustering problems, for example [11,12,15,18], aim to minimize a fitness function, usually a distance measure, relating the data to a cluster centroid.

Minimizing the distance between data points and cluster centroids is a logical approach, yet it does not necessarily provide topological information of the data. The mentioned methods have provided reasonable minimum distance results, though they lack the information that is, in many cases, essential for extracting intuitive knowledge from the data. When used in pattern recognition or classification applications, despite the fact that the fitness criterion is satisfied, result accuracy (misclassification) may be an issue.

The Hyperbox clustering with Ant Colony Optimization method (HACO) is proposed for clustering unlabeled data by placing hyperboxes in the feature space optimized by the ACO. It applies an optimization technique combined to a well-known local search algorithm to the clustering problem, acknowledging the topological information of the data, if available.

Hyperboxes are placed in the search space using the ACO meta-heuristic and then clustered using the Nearest-Neighbor (NN) method. The number of hyperboxes to perform the search with is usually smaller than (or, in the worst case, equal to) the number of samples, which means that the search can be done in less iterations, making HACO a fast classifier.

HACO is applied to three computer-generated 2D data sets which have significant topological information for validation as a classification method, considering speed and accuracy. It is then applied to a Human Papillomavirus (HPV) data set in order to identify probable infection profiles that may be used as a basis for preventive medical check-ups. It is compared to two well-established clustering methods and the usual ACO approach and the results show that the HACO method can be more effective than the others.

A brief description of ACO and definition of hyperboxes are presented in Section 2; Section 3 proposes and details the HACO method; clustering experiments and the results are analyzed in Section 4.

2. Description of ant colony optimization and hyperboxes

2.1. A review on ant colony optimization

The ACO meta-heuristic is population-based and can be readily applied to discrete combinatorial optimization problems. It makes an analogy of the way real ant colonies work to optimize combinatorial problems [4,5]. The basic idea is the synergy of

* Corresponding author.
E-mail addresses: ramos@hrt.dis.titech.ac.jp, ramos.at.titech@gmail.com (G.N. Ramos).

1568-4946/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2008.09.004

applying multiple communicating agents to build a solution. Real ants communicate with each other by depositing pheromone on the trail between the food source and the nest [3]. The shorter the trail, the faster the ants will go through it and thus more pheromone will be deposited. Since ants have a high probability of following trails with higher pheromone deposition, the process reinforces itself [4,5].

This is a distinctive feature of ACO: the pheromone matrix works as dynamic memory, indicating how desirable a data object is to the solution [4], and thus mediates how one ant's behavior is determined by the previous ants [3–5]. The values are updated according to the quality of the solutions, so the process "remembers" good solutions and "forgets" bad ones. Similar agent-based applications have been used for data clustering, but such algorithms usually follow the Ant Cemetery approach [12], which provides no global control over the agents. This approach has been combined with Fuzzy C-Means [11] and K-Means [15] algorithms in order to improve the quality of results.

The main characteristics of ACO approach are positive feedback (improves speed of finding good solutions), distributed computation (avoids early convergence) and greedy heuristic (finds reasonable a solution early in the process) [4,5,7]. Due to such characteristics, however, it may be outperformed by specialized algorithms [4,5].

The ACO can be simplified in three basic procedures per iteration [4,5]: build solutions, local optimization (an optional step) and pheromone update. When applied to finding suitable data set partitions for clustering [18], i.e., dividing the data set into distinct classes represented by the clusters, ACO aims to minimize distances between the samples and the centroids.

Due to its inherent characteristics (flexibility and fast convergence), the ACO algorithm is a good approach for partition clustering. In this case, since the objective is to minimize the distance between data objects and the cluster centroids, it attempts to cluster the closest data objects. This is done by assigning clusters to each data object, and then calculating the distances. It may not produce the best results; however, depending on how the data is distributed on the feature space. This approach does not consider the topology of the feature space, which is inherently associated to classification processes [6,17]. For example, consider the data set shown in Fig. 1.

Intuitively, it is clear that the classes are distributed as one long curved-shaped cluster (the points with positive vertical coordinates) and one oval cluster (points with negative vertical coordinates), as in Fig. 1a. The ACO algorithm, however, defines a partition of the data in such way that it is divided into one elongated cluster and one larger cluster, as in Fig. 1b. The fitness of the solution in Fig. 1b is indeed better than the fitness of Fig. 1a; nevertheless it is clear that the solution lacks the topological information.

2.2. Describing hyperboxes

A hyperbox defines a region in an n-dimensional space [17,19,20] and is fully described by two vectors, usually its two extreme points: a_l which is the lower bound and b_l, the upper bound. Assuming an n-dimensional space of real numbers ($\Re^n$) and a hyperbox $H_l = (a_l, b_l)$, where $a_l \leq b_l$, a point y is said to be in H_l if

$$\begin{aligned}
&H = \{H_1, H_2, \ldots, H_l, \ldots, H_C\},\\
&H_l \subset \Re^n,\\
&y = \{y_1, y_2, \ldots, y_j, \ldots, y_n\},\\
&y \in H_l \Rightarrow a_{lj} \leq y_j \leq b_{lj}, \quad a_l, b_l \in \Re^n,
\end{aligned} \tag{1}$$

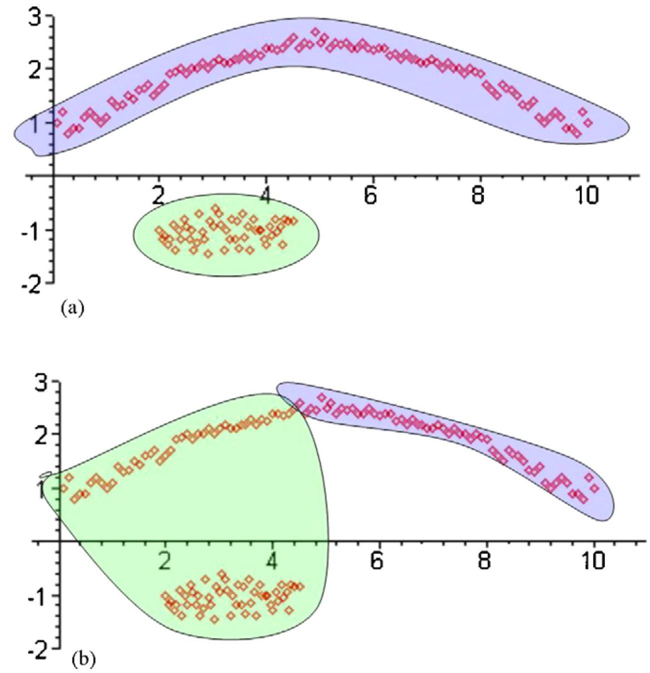

Fig. 1. Data partition examples: (a) intuitive partition; (b) ACO partition.

where C is the total number of hyperboxes and y_j is the jth attribute of y.

Using this definition it is necessary to have two points for a hyperbox; however, the objective of HACO is to group data objects that are near each other, which accounts to some problems on choosing this points. It is, therefore, more convenient to use a slightly different approach, which maintains the useful characteristics of a hyperbox. It can be defined by one point (in HACO, one data object x_i) and an n-dimensional vector D which defines the edge lengths for each attribute, as in $H_l = (x_i, D)$. Therefore, each hyperbox will define a region in the space around such data point, as follows:

$$\begin{aligned}
&X = \{x_1, x_2, \ldots, x_i, \ldots, x_N\} \subset \Re^n,\\
&D = \{D_1, D_2, \ldots, D_j, \ldots, D_n\},\\
&\forall y \in \Re^n,\\
&y \in H_l \Rightarrow x_{ij} - \frac{D_j}{2} \leq y_j \leq x_{ij} + \frac{D_j}{2},
\end{aligned} \tag{2}$$

where X is the set of data points with cardinality N, and D_k is the edge length for the jth hyperbox dimension.

Hyperbox classifiers can give straightforward interpretation for classification rules [17], such as "if $y \in [a_l, b_l]$ then y belongs to the class defined by H_l'', without calculating any distances. Also, if associated with a fuzzy membership function, they can be used as inputs for fuzzy min–max neural networks to be applied in classification [19] or clustering [20]. In these applications, data is assumed to be labeled and part of it is used for training.

It is possible to automatically determine shape patterns by grouping hyperboxes, and then define a class according to the specific characteristics. Since overlapping may occur but data objects are not allowed to belong to different classes (crisp clustering [10]), the proposed method requires that overlapping hyperboxes represent the same class. In other words, hyperboxes representing different classes must be disjoint.

3. HACO: Hyperbox clustering with Ant Colony Optimization

Evolutionary and meta-heuristic methods are globalized search techniques, whereas usual deterministic and stochastic methods are localized ones [4,5], i.e., the latter require good initial conditions to be able the output a good result. To search the possible labels for an optimum value of criteria is computationally prohibitive [4,5]; however, ACO has proved fast convergence [9].

The proposed HACO method aims to find a partition of the data into clusters, a combinatorial problem, since the partition may give insight into some inherent structure of the data or identify a pattern [13]. Despite the similar denomination, HACO does not resemble the Hyper-cube framework for ACO [2], which limits the pheromone values to the [0, 1] interval.

The HACO method has two steps, illustrated in Fig. 2. First ACO is applied to optimally scatter the hyperboxes on the feature space. This population-based approach allows search in more than one solution at a time [4,5]. The second step is to group (or ungroup) the hyperboxes to form clusters that will define the classes.

In the second step, if the number of clusters is known beforehand, the hyperboxes are grouped using a NN method, so the solution considers the topology of the data set to provide better quality results. NN considers proximity to be a key role in intuitive clustering; it assigns each unlabeled object to the cluster of its nearest labeled neighbor object if the distance between them is smaller than a given threshold [10,14].

HACO results provide clusters representing classes, which enable a simple, straightforward classification for the data objects.

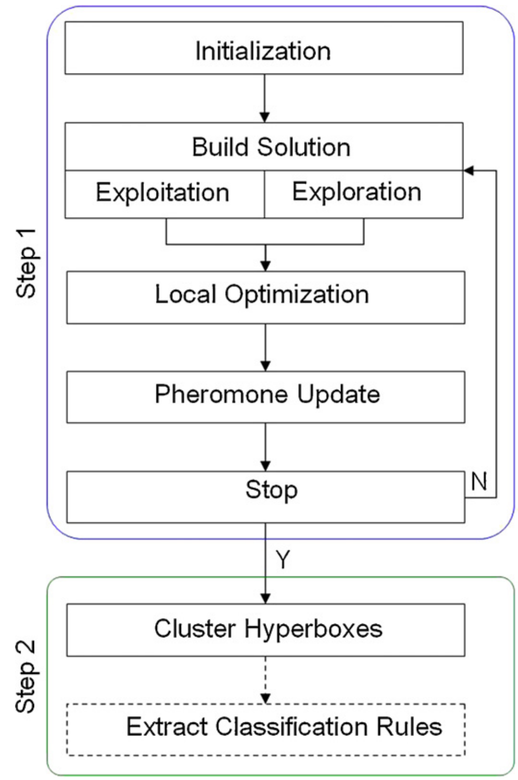

Fig. 2. HACO algorithm diagram.

3.1. Using ACO to place hyperboxes

The proposed method starts by loading the data and determining the number of hyperboxes C to use as

$$C = rd\left(\alpha \cdot \prod_{j=1}^{n} \frac{||\max(x_{ij}) - \min(x_{ij})||}{D_j}\right), \tag{3}$$

where the function rd rounds the resulting number to the nearest integer, α is the search space ratio that defines up to how much of the feature space can be occupied by hyperboxes, D_j is the length of the hyperbox edge in the jth dimension and x_i is the ith data sample.

The parameter α's purpose it to allow the user to have more control over the resulting classifier, since it indicates the ratio of the feature space ratio to be occupied with hyperboxes. It can be any value in the interval [0, 1]. If C is found to be greater than N (either the feature space is too large, the hyperbox dimensions are too small or both), it is reassigned to N.The ACO algorithm, illustrated in step 1 of Fig. 2, will repeat its three basic steps until at least one of the following stop criteria is achieved:

- Number of iterations is equal or more than I.
- The density of the best solution found is 1.

In HACO the quality of a solution is measured as its density, i.e., the relation of the number of data objects that belong to a hyperbox. The density $d_r \in (0, 1]$ of a solution S_r (generated by the rth agent) can be calculated as

$$S_r = \{H_{r1}, H_{r2}, \ldots, H_{rC}\},$$
$$d_r = \frac{1}{N}\sum_{i=1}^{N} f_r(x_i), \tag{4}$$

where

$$x_i \in X,$$
$$l, m \in \{1, 2, \ldots, C\}, \quad l < m,$$
$$f_r(x_i) = \begin{cases} 1, & x_i \in H_{rm}, x_i \notin H_{rl}, \\ 0, & \text{otherwise}. \end{cases} \tag{5}$$

Initialization of the pheromone matrix is done by setting all values to a small positive value $\tau_0 \in \Re$ [4,5], so no partition is preferred over the others, concluding the initialization procedures.

A solution vector S_r is built in every run by each of the R agents, applying the information stored in the pheromone matrix T to assign a hyperbox H_l to a data object x_i. Each solution is subject to the following constraint:

$$\forall\, l, m \in \{1, 2, \ldots, C\}, \quad l \neq m \Rightarrow S_{rl} \neq S_{rm}. \tag{6}$$

This means that, for all solutions, no two hyperboxes are assigned to the same data sample.

The solution can be built in two ways. The first is called *exploitation* [4,5] (probabilistic choice), where the agent chooses the sample with highest pheromone value with a probability $P_E \in [0, 1]$. In case this probability is not satisfied, the agent chooses one of the possible samples with a stochastic distribution p_{il} of the pheromone values, such that

$$p_{il} = \frac{\tau_{il}}{\sum_{i=1}^{N} \tau_{il}}, \tag{7}$$

where τ_{il} is the element of T that associates the pheromone concentration of the ith data point to the lth hyperbox. This process is called *exploration* [4,5].

When exploring, a probability is randomly selected to be compared to the stochastic distribution of the pheromone. This means that the pheromone values for each possible sample is a number in [0, 1], and the sum of all pheromones is 1. Therefore the value for each sample can be considered as the probability of choosing such sample as part of the solution. The sample is chosen by adding the pheromones of each sample until the sum value is greater than the random probability. So the sample is selected randomly among all possible alternatives (possibilistic choice).

The exploitation/exploration of the pheromone matrix accounts for the global search of the optimal solution. Further optimization can be implemented by modifying the solutions built by the agents locally.

Local optimization is done after the agents have built the solution. The system tries to optimize the best solutions by randomly changing the data object assigned to each hyperbox with a local search probability P_{LS}. The local optimization is also subject to Eq. (6) and the modified solution is stored if there is an improvement on the quality, i.e., if its density increases.

Updating the pheromone matrix according to the quality of the solutions is the final, and most important, step in each iteration. HACO updates the pheromone based on the best L solutions generated ($L \leq R$). This resembles the elitist selection strategy used in genetic algorithms, where only the best fit individuals in the population are carried on to the next generation [14,17]. It ensures the agents will tend to build new solutions based on the knowledge gained from previous iterations.

Thus, a solution S' is created from the L solutions with highest densities, and each pheromone value is updated as

$$\tau_{il}(t+1) = \rho \cdot \tau_{il}(t) + \sum_{r=1}^{L} d'_r, \tag{8}$$

where ρ is the trail persistence and d'_r is the density of the rth solution in S'. The trail persistence defines how well the system remembers the knowledge acquired with previous solutions, and $1-\rho$ is called the evaporation rate [4].

The quality of the solution defines how the pheromone matrix will be updated and, therefore, how the system will search for solutions. The main goal of the system is to cover all samples (density is equal to 1), but if that is not possible, the other criterion asserts a maximum runtime. ACO is applied to maximize the density of the solution by placing the hyperboxes where they will contain as many data objects as possible while maintaining the constraints specified in Eq. (6).

3.2. Hyperbox clustering

After running ACO until the stop criteria have been achieved HACO classifies the hyperboxes. Overlapping hyperboxes are joined into one cluster (class) and non-overlapping ones are assigned to their own clusters. This automatically provides the number of clusters (a useful resource for unknown data), though it still is highly dependent on the data set's topology and hyperbox dimensions. Two hyperboxes overlap if

$$\begin{array}{c} H_l, H_m \in H, \quad l \neq m, \\ \exists y \in \Re^n \quad \text{s.t. } y \in H_l, y \in H_m, \end{array} \tag{9}$$

where y is any n-dimensional point. A simple way to test this is by checking if the upper and lower bonds of two hyperboxes intersect. It is important to emphasize that even though hyperboxes may overlap, according to (5) the data objects may only belong to one hyperbox for classification.

In the case the number of clusters K is known a priori, HACO proceeds as follows. If the number of clusters found is greater than the given K, the clusters that are closest to each other are grouped together using the Nearest-Neighbor method [10,14]. If the number of clusters proposed is smaller than the given, HACO assigns a new hyperbox and new associated cluster to the sample with the greatest distance from the existing hyperboxes, repeating this procedure until the given number of clusters is achieved.

3.3. HACO for classification

The goal of data mining is, essentially, to extract knowledge from the data accurately and comprehensibly [16], since the result is usually applied to support human decisions. The hyperboxes assigned to classes provide a straightforward classifier, from which classification rules can be easily extracted [17].

To this end, once the hyperbox clusters are defined, each sample is assigned to a cluster with the following criteria: if a data object x_i is contained in the region defined by a hyperbox H_l, see Eq. (1), there is a straightforward classification (it is assigned to the same cluster as H_l); else it is assigned to the same cluster as the hyperbox closest to it.

The cluster centroids c_r, for the rth solution are calculated as the mean value of all data objects assigned to each cluster. The fitness of the clustering is given by

$$F(S_r) = \sum_{i=1}^{N}\sum_{k=1}^{K} \omega_{ik} \cdot dist(c_{rk}, x_i), \tag{10}$$

where $dist$ is the standard Euclidean distance between kth cluster centroid of c_r and the ith data point, and

$$\omega_{ik} = \begin{cases} 1, & x_i \in A_k, \\ 0, & x_i \notin A_k, \end{cases} \tag{11}$$

where A_k is the class defined by the kth cluster.

4. HACO experiments on data sets

4.1. Experimental setup

Experiments are run under Suse Linux® on a Pentium M® 1.6 GHz with 512 MB of RAM. For each data set the results for the following clustering methods are gathered: Nearest-Neighbor (NN), Fuzzy C-Means (FCM [1]), Ant Colony Optimization (ACO, partition search approach [18]) and Hyperbox Clustering with Ant Colony Optimization (HACO).

The comparison with only FCM and NN has two key points. First, they are well-known algorithms and, therefore, provide a more intuitive base for comparison with the method proposed. Second, they provide different approaches to the clustering problem when compared to evolutionary techniques.

HACO and ACO are probabilistic methods, and FCM is very dependent on the initialization (done randomly in the experiments), so different runs may return different results. For all methods, each run is considered to be the processing of all steps necessary to obtain a final solution that classifies all samples. For statistical validation, each method is applied 10 times to each data set, and the best accuracy results and average runtime are compared.

Running the experiments many times also accounts for a reasonable idea of the runtime required for running each algorithm once, since that may change between attempts due to a number of reasons. This average runtime information is used to compare the performances of HACO and ACO, which have similar approaches to the problem.

NN is a fast and simple method which assigns each unlabeled object to the cluster of its nearest labeled neighbor object if the distance between them is smaller than a given threshold [10,14]. The efficiency of this method depends on the size of the data set and its features, since it calculates the distances for all objects and these distances may be dominated by irrelevant features (curse of

Table 1
Algorithm input parameters.

FCM	ACO	HACO
$I = 1000$	$I = 1000$	$I = 1000$
$\varepsilon = 0.0001$	$R = 50$	$R = 50$
$m = 2$	$P_E = 0.98$	$P_E = 0.98$
	$L = 1$	$L = 1$
	$P_{LS} = 0.01$	$P_{LS} = 0.01$
	$\rho = 0.99$	$\rho = 0.99$

dimensionality [14]). The NN algorithm implemented in this work is slightly modified to accommodate the a priori knowledge of the number of clusters. The threshold is changed according to the number of clusters obtained, increasing if the number is too big and decreasing if it is too small, thus forcing the partition to provide the desired number of clusters.

Fuzzy clustering extends the notion of association between objects and clusters using a membership function that defines with what degree each object belongs to each cluster [1,21]. FCM is a well-known clustering method which runs for at most a number I of iterations or until the difference between the weights in two consecutive iterations is less than an error threshold ε. The shape of the fuzzy sets is controlled by a fuzzification factor $m > 1$ [1], and initial fuzzy weight values (belonging to $[0, 1]$) are assigned randomly. The fitness of the solution given by the FCM algorithm is also calculated by Eq. (10), having ω_{ik} as the membership value.

Each of the applied methods has its own set of parameters, listed in Table 1. They are assigned, according to references, as values that produce good results. NN uses only a distance threshold value as input parameter which is not considered since it is adjusted as needed to fit the desired number of clusters.

Though the HACO accommodates hyperboxes of any shape, there is no logic implemented to reasonably define which shape they should have. Therefore, hyperboxes with the same shape and volume are considered for the experiments, so all elements of the vector D are equal ($D_1 = D_2 = \cdots = D_n = D'$). Since this is still a validation tryout for the HACO, the parameter D' is set to different values to verify its influence on results.

The fuzzification factor for FCM is usually set to 2 [17]. ACO parameters are set according to [11,18] and, for comparison purposes, the parameters for HACO are set likewise.

The number of clusters K desired for each data set is also given as a parameter, according to the data set used in the experiment. For validation, each method is applied to three different computer-generated data sets and then tested with a real set of HPV data.

4.2. Synthetic data set for validation

Each of the three computer-generated data sets, shown in Fig. 3, has a distinct topology to better assert the method's efficiency and, for illustration purposes, only two features.

The first data set, Fig. 3a, consists of 150 objects intuitively forming two clusters, one with an ellipsoid geometry and the other resembling the form of a horseshoe. The data set in Fig. 3b consists of 302 objects forming three clusters, two with ellipsoid geometry and one with an elongated shape. Fig. 3c has 600 objects that form three distinct ellipsoid clusters.

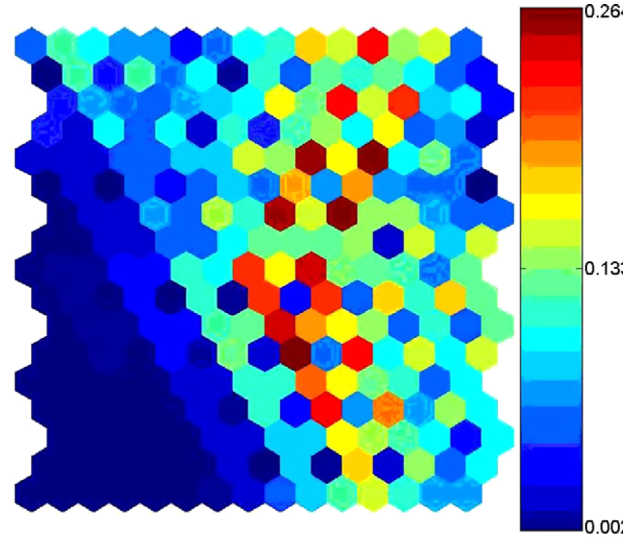

Fig. 4. U-matrix for HPV data. (For interpretation of the references to color in this artwork, the reader is referred to the web version of the article.)

4.3. HPV data—a real world application

The Human Papillomaviruses (HPVs) cause long-term infections that can be detected. There are no effective antiviral drugs yet available, and treatment should be provided because the disease can progress to cancer [8].

The Department of Stomatology, part of the Dentistry School in Unigranrio, has collected 199 samples of oral mucosa and a questionnaire for 42 physical characteristics and habits amongst the local population. The questionnaire provides the features while the mucosa gives the health status (infected or not by HPV) of each individual. At the time of collection all individuals seemed healthy.

The questionnaire contains information such as gender, job, smoking habits, use of mouthwash, and others. It is very broad and was filled without any constraints, meaning each question was answered by the individual's standards, not previously specified values. For example, the smoking frequency assumes values such as "20 cigarettes per day" and "on weekends".

Since some features had no influence on the result (for example, all individuals are of Brazilian nationality) or could not be properly converted into numerical values (profession or place of collection of the mucosa), they were not considered for the experiments. The data set used has 185 samples with 21 features, and is to be divided into two clusters (infected and healthy individuals). Fig. 4

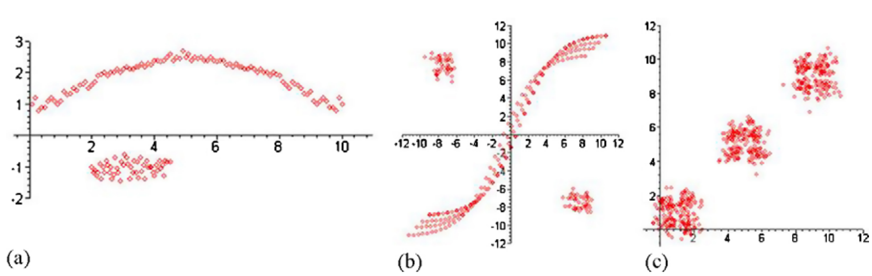

Fig. 3. Computer-generated data sets: (a) 150 samples; (b) 302 samples; (c) 600 samples.

Table 2
Influence of HACO parameter α in data set 2.

α	Number of hyperboxes	Density	Suggested K	Runtime (s)
0.1	47	0.8179	41	39
0.2	95	1	25	22
0.3	143	1	14	0.3
0.4	191	1	14	0.71
0.5	239	1	6	0.94
0.6	287	1	6	1.49
0.7	302	1	9	1.65
0.8	302	1	9	1.66
0.9	302	1	9	1.7
1.0	302	1	9	1.77

illustrates the highly dimensional data through the U-matrix of a self-organizing map.

4.4. HACO experimental results

Considering only data set 2, experiments were run to assess the influence of the parameter α in the results. In this experiment, D' is fixed at 1 and the number of clusters K is not given as input. Details of the results are given in Table 2.

As expected, the number of hyperboxes C, obtained from Eq. (3), grows linearly with α until the threshold (number of samples) is reached.

Also as expected, the density of the solution grows as the number of hyperboxes grows. The suggested number of clusters decreases as the number of hyperboxes, which stabilizes around to suggest around 9 clusters, grows. This is not surprising more hyperboxes of the same size being used means a higher chance of them overlapping. None of the results for suggested number of clusters is the desired 3, suggesting that the hyperbox dimensions also have a strong influence on this matter.

Finally, the runtime decreases drastically with the increasing α, as a consequence of the density stop criterion. This results give insight on how α affects the performance of HACO, and point to a relationship between this parameter, D' and the suggested K. The results are very dependent on the data set in question, but knowing this, and having the number of clusters of the data and the range of its features, used in Eq. (3), it may be possible to automatically define reasonable values for D'.

HACO proposes α to give the user more control over the resulting classifier geometry. Larger values result in more space being taken, which may cause the clusters to be too large (and the classifiers too general). On the other hand, smaller values may result in many smaller clusters (and classifiers that are not general enough). Thus, for the rest of the experiments, the parameter α is set empirically to the intermediate value 0.5. The edges of the hyperboxes, determined by D', will have different values to assess how they affect the performance.

Further evaluation of HACO is done with experiments in all the computer-generated data sets, and the results for each method are shown in Table 3. Time values are considered the mean runtime for all 10 runs of the algorithm; fitness and accuracy values are given by Eqs. (10) and (12), respectively, and are described only for best solution found. Parameters used are the ones given in Table 1, and visual representation of these results is given in Figs. 6–8.

Table 3
Experimental results for synthetic data sets.

	Data	Runtime (s)	Fitness	Accuracy
NN	1	0.02	889.48	100%
	2	0.05	19164.57	100%
	3	0.10	594.07	100%
FCM	1	0.04	913.55	74%
	2	0.10	24457.35	48.68%
	3	0.20	1743.70	100%
ACO	1	6.10	477.47	70.67%
	2	16.20	11089.20	55.96%
	3	33.00	4754.55	72.19%
HACO, D' = 1	1	3	889.48	100%
	2	20.9	19164.57	100%
	3	21.7	594.07	100%
HACO, D' = 2	1	1	889.48	100%
	2	9.9	19164.57	100%
	3	10	594.07	100%
HACO, D' = 3	1	1	489.89	65.33%
	2	6.1	19164.57	100%
	3	5.1	594.07	100%

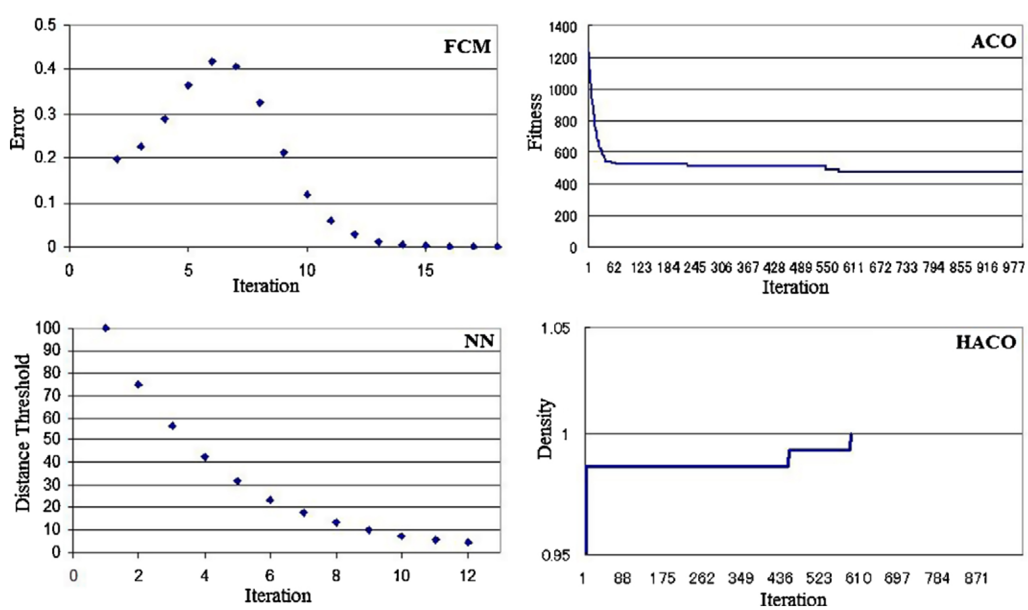

Fig. 5. Evolution of each method.

The accuracy indicates how well each method classifies the data sample [17] and is defined as

$$accuracy = \frac{T_P + T_N}{N}, \tag{12}$$

where T_P is the number of samples correctly classified as positive (belonging to the class), T_N is the number of samples correctly classified as negative (not belonging to the class), and N is the total number of samples.

Both FCM and ACO approaches aim to minimize the distances between data objects and the cluster centroids (their objective function), but the HACO aims to maximize the hyperbox density values (which leads to minimizing the distances). NN joins the samples into clusters, and aims to find a distance threshold that results in the desired number K. The progress of each method is illustrated in Fig. 5.

The maximum hyperbox density is only obtained for data set 1, in other cases the stop criterion is the number of iterations. Notice that once the hyperbox density stop criterion is achieved (around iteration 600), HACO stops searching for other solutions. Using more stop criteria provides faster results and decreases the computational cost of the algorithm.

The fitness found using HACO usually has a higher value than the one found using ACO because HACO uses the topological information when grouping data objects. The ACO can find an optimal solution given a large enough number of iterations and runs, but it produced a bad result for the simplest case of three distinct ellipsoid clusters, shown in Fig. 8c. In a separate experiment, considering only data set 3 and having 200 iterations and as many runs, ACO eventually found much better solutions (though not as accurate as the other methods), but it obviously required a much longer runtime.

Accuracy for FCM results is obtained from classification after a simple defuzzification: each object is assigned to the cluster to which it belongs with highest degree. With crisp clusters the fitness can also be recalculated, assuming values closer to the fitness of other methods.

It is important to notice that FCM assigns fuzzy values while the other two assign crisp values. This means that even though FCM produces similar solutions to the others, Figs. 7a and 8a or Figs. 6c and 7c, the fitness values can be significantly higher.

The accuracy of HACO remains the same for data sets 2 and 3 for all values of D' tested, but it changes for data set 1 with $D' = 3$. In this last case, the size of the hyperboxes is large enough for two of them to overlap and thus be joined in one cluster. The resulting classification provides good fitness but lacks accuracy.

The application of the Nearest-Neighbor method in HACO allows it to consider the topology of the data sets, clearly a key aspect in classification, improving the quality of the result as is obvious in Figs. 6–8.

Notably, FCM quickly provided an excellent result for data set 3; however, this is because the geometry is favorable to this method

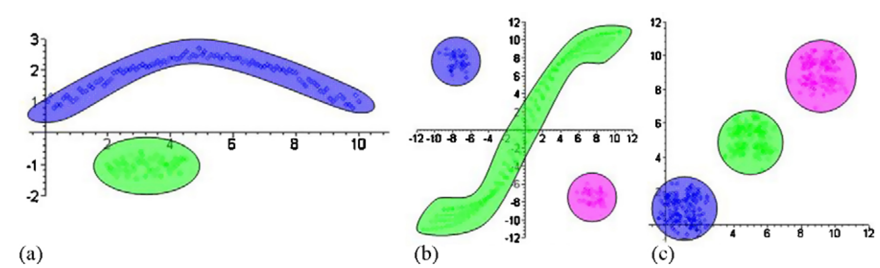

Fig. 6. Experimental results using HACO and using NN: (a) data set 1; (b) data set 2; (c) data set 3. (For interpretation of the references to color in this artwork, the reader is referred to the web version of the article.)

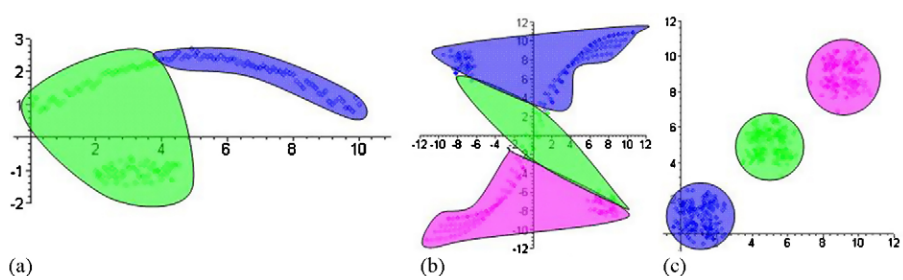

Fig. 7. Experimental results using FCM: (a) data set 1; (b) data set 2; (c) data set 3. (For interpretation of the references to color in this artwork, the reader is referred to the web version of the article.)

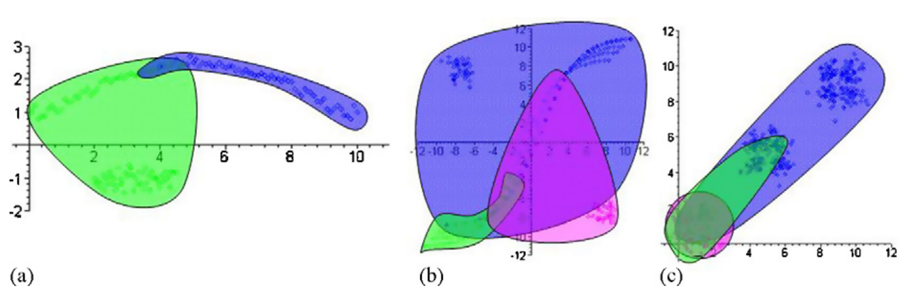

Fig. 8. Experimental results using ACO: (a) data set 1; (b) data set 2; (c) data set 3. (For interpretation of the references to color in this artwork, the reader is referred to the web version of the article.)

Table 4
Experimental results for HPV data set.

	Runtime (s)	Fitness	Accuracy
NN	0.30	40534.31	68.41%
FCM	0.70	21880.28	50.81%
ACO	42.00	12452.35	52.35%
HACO	Runtime (s)	Fitness	Accuracy
$D' = 1$	5.80	25555.14	71.35%
$D' = 2$	2	31918.13	68.11%
$D' = 3$	1	31522.15	67.57%

(only ellipsoid clusters). If an elongated cluster is presented (as in data sets 1 and 2), the results for FCM are not so accurate. These results show that HACO is a suitable method for classification, even if the data has inherent topological information and that it can be applied to real data sets.

Considering the HPV data set, the high number of features makes it difficult to visualize the clusters. Experimental results are shown in Table 4. NN provides fast result, almost as accurate as the HACO. The FCM method also provides fast results, but lacks accuracy. ACO gives slightly better results than FCM, but not as well as NN and with a much longer runtime. The quality of HACO results does not vary much with the hyperbox dimension; however, the computational cost of $D' = 3$ is almost 6 times lower than the one of $D' = 1$.

The NN method is fast and provided very good results for the given test and real data sets. This approach, however, is usually used only for local optimization (usually as pre-processing) and may miss the globally optimum solution [10,14]. The runtime increases with the data set size, so its performance against much larger sets may be different.

FCM is clearly an efficient method; however it does not consider the geometry as an important characteristic of the data set. It also may be outperformed due to poor choice of initial values, in the experiments they were randomly assigned, but gave results similar to the ones in Table 3 when “properly” initialized (initial weight values set according to desired classification, as in Fig. 1a).

The ACO method applied to partition clustering may find good results for the validation data sets, as in Fig. 8a, and presents competitive results when compared to other optimization methods [18]; however it also does not consider the topology of the feature space. Better results may be found by changing some parameters like number of agents or iterations, which will consequently increase the computational cost.

Considering $D' = 1$, HACO uses more hyperboxes and thus has provides a finer approach to the actual shape of the cluster, providing a result over 7 times faster than ACO. For the other values of D', the accuracy decreased; however, this is a reasonable tradeoff due to the gain in processing time.

The results show that the use of hyperboxes in searching provided a faster application. The accuracy (for $D' = 1$) is better than for NN, almost 40% improved over FCM and over 30% better than the accuracy for ACO, showing that HACO was successfully applied to a real world problem, outperforming well-established algorithms (NN and FCM) and the algorithm it is based on (ACO). While this might not be enough to state that HACO is one of the better performing clustering methods available, it should suffice to show that the proposed algorithm provides results which are reasonable enough to be considered and further investigated.

HACO’s major contribution to the ant colony body of knowledge is speed and quality of results, clearly seen from the results for artificial and HPV data. Using hyperboxes, HACO acknowledges the topological information of the data, and can be easily turned into an intuitive classification tool.

The risk profile provided by HACO was shown to an expert in Unigranrio and approved as a tool in aiding HPV infection diagnosis. It is important to emphasize that HACO does not pinpoint the disease in someone, since the exact symptoms/characteristics of HPV infection are not yet defined and vary from place to place. What HACO achieved is a probable risk profile for the specific region where the data was collected. This aids the health professionals by indicating the necessity of further examination of a patient that fits the identified risk profile without the burden of a full check-up.

5. Conclusion

The Hyperbox clustering with Ant Colony Optimization (HACO) method is proposed for data classification. It uses the Ant Colony Optimization (ACO) to search the feature space and hyperboxes to make the space smaller, and a local optimization method to acknowledge the topological information.

The HACO method is validated for data classification with three computer-generated 2D data sets and applied to a Human Papillomavirus (HPV) data set in order to identify probable risk profiles and enable early diagnosis, a crucial factor for preventing the disease to evolve into cancer.

The method is compared to Nearest-Neighbor (NN) and Fuzzy C-Means (FCM), two well-established clustering methods, and to an ACO partition approach for clustering method. The quality of the results given by HACO is competitive against the NN results and significantly better than the FCM and ACO. Results show that data geometry is indeed an important factor and should be considered.

The experimental results obtained for the three computer-generated data sets validate HACO as a clustering method that considers the shape of the input data, suitable for pattern recognition applications. The results for the HPV data set show that HACO is better suited for this data than the other three methods. NN also provided reasonable results, but this method is limited when applied to highly dimensional data sets.

Accuracy values from HACO are significantly better than FCM and ACO (over 30% more accurate), and the runtime improvement when compared to the latter is substantial (over 7 times faster). Class features can be readily obtained from the HACO HPV results, and the cluster centroid for each class provides a probable risk profile for infection in the location of sample collection. Comparison of a risk profile and an individual profile may indicate need for further examination and, consequently, early diagnosis of infection.

HACO showed promising results for clustering (and straightforward extraction of classification rules), indicating that further development in this research should be considered. Better assessment of the performance can be achieved by comparison with newer and/or similar approaches, such as GA, SA, PSO, Ant Cemetery, Extended FCM, and other hybrid evolutionary approaches.

Future perspectives also include automatic parameter tuning from the input data set, specially the hyperbox edge length D, in order to minimize user burden and improve the quality of the results. Further development can also be pursued in optimizing the hyperbox dimensions, to fine tune results, and in verifying how HACO performs when applied to larger data sets or to similar medical data, aiming to use this method as part of a computer aided diagnosis system. HACO may also be used with any type of data, and can be readily applied, for example, in software quality assessment or telecommunication customer’s profile identification.

Acknowledgments

The authors would like to thank Prof. Witold Pedrycz for insightful comments, Dr. Doralina Rabello and Prof. Andréa Moleri of Unigranrio University for the data and expert opinion on the subject. The authors also thank the anonymous reviewers for their useful suggestions.

References

[1] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenun, 1981.
[2] C. Blum, M. Dorigo, The hyper-cube framework for ant colony optimization, IEEE Transactions on Systems, Man and Cybernetics - Part B 34 (2) (2004) 1161–1172.
[3] J.-L. Deneubourg, S. Aron, S. Goss, The self-organizing exploratory pattern of the argentine ant, Journal of Insect Behavior 3 (2) (1990) 159–168.
[4] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man and Cybernetics - Part B 26 (1) (1996) 29–41.
[5] M. Dorigo, T. Stützle, Ant Colony Optimization, The MIT Press, 2004.
[6] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley–Interscience, 2000.
[7] L.M. Gambardella, M. Dorigo, An ant colony system hybridized with a new local search for the sequential ordering problem, INFORMS Journal on Computing 12 (3) (2000) 237–255.
[8] R.J. Greenblatt, Human papillomaviruses: diseases, diagnosis, and a possible vaccine, Clinical Microbiology Newsletter 27 (18) (2005) 139–145.
[9] W.J. Gutjahr, A graph-based ant system and its convergence, Future Generation Computer Systems 16 (8) (2000) 873–888.
[10] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Computing Surveys 31 (3) (1999) 264–323.
[11] P.M. Kanade, L.O. Hall, Fuzzy ants and clustering, IEEE Transactions on Systems, Man and Cybernetics - Part A 37 (5) (2007) 758–769.
[12] M. Martin, B. Chopard, P. Albuquerque, Formation of an ant cemetery: swarm intelligence or statistical accident? Future Generation Computer Systems 18 (7) (2002) 951–959.
[13] P. Michaud, Clustering techniques, Future Generation Computer Systems 13 (2–3) (1997) 135–147.
[14] T. Mitchell, Machine Learning, McGraw-Hill, 1997.
[15] N. Monmarché, M. Slimane, G. Venturini, AntClass: discovery of clusters in numeric data by an hybridization of an ant colony with k-means algorithm, Internal Report No. 213, Laboratoire d'Informatique de l'Universite, 1999.
[16] R.S. Parpinelli, H.S. Lopes, A.A. Freitas, Data mining with an ant colony optimization algorithm, IEEE Transactions on Evolutionary Computing 6 (4) (2002) 321–332.
[17] W. Pedrycz, G. Succi, Genetic granular classifiers in modeling software quality, The Journal of Systems and Software 76 (3) (2005) 277–285.
[18] P.S. Shelokar, V.K. Jayaraman, B.D. Kulkarni, An ant colony approach for clustering, Analytica Chimica Acta 509 (2) (2004) 187–195.
[19] P.K. Simpson, Fuzzy min–max neural networks. Part 1. Classification, IEEE Transactions on Neural Networks 3 (5) (1992) 776–786.
[20] P.K. Simpson, Fuzzy min–max neural networks. Part 2. Clustering, IEEE Transactions on Fuzzy Systems 1 (1) (1993) 32–45.
[21] R.R. Yager, L.A. Zadeh, Fuzzy Sets, Neural Networks, and Soft Computing, John Wiley & Sons, Inc., 1994.