A New Evolutionary-Based Clustering Framework for Image Databases

Alessia Amelio and Clara Pizzuti

Institute for High Performance Computing and Networking,
National Research Council of Italy, CNR-ICAR,
Via P. Bucci 41C, 87036 Rende (CS), Italy
{amelio,pizzuti}@icar.cnr.it

Abstract. A new framework to cluster images based on Genetic Algorithms (GAs) is proposed. The image database is represented as a weighted graph where nodes correspond to images and an edge between two images exists if they are sufficiently similar. The edge weight expresses the level of similarity of the feature vectors, describing color and texture content, associated with images. The image graph is then clustered by applying a genetic algorithm that divides it in groups of nodes connected by many edges with high weight, by employing as fitness function the concept of weighted modularity. Results on a well-known image database show that the genetic approach is able to find a partitioning in groups of effectively similar images.

Keywords: Genetic Algorithms, image clustering, graph partitioning, content based image retrieval, database summarization.

1 Introduction

Content-based image retrieval (CBIR) is an active research field whose aim is to search for digital images in large image databases. The term *content* means that a CBIR system analyzes an image with respect to an abstract representation derived from the image, such as color, texture, shape. Initial CBIR systems were based on a search-by-query strategy, i.e. a user gives a query image and the system exhaustively compares this image with those contained in the database to obtain the most similar. In the last few years, however, image repositories have dramatically increased in size and contain a huge number of images, thus approaches to speed up retrieval are necessary and desirable. Grouping images into categories and extracting salient characteristics for each group to define a cluster representative, is a methodology proposed by many researchers to reduce computing time requirements. In fact, since the query image is compared with the cluster representatives, and generally, the number of obtained clusters is much lower than the number of images contained in the database, the response times can dramatically decrease.

Several methods have been proposed for clustering images. Approaches can roughly be classified into three main categories [3] : paire-wise-distance-based, optimization of a quality measure that assesses clustering result, and statistical modeling. Hierarchical clustering and spectral graph partitioning are representatives of the former category.

A. Elmoataz et al. (Eds.): ICISP 2014, LNCS 8509, pp. 322–331, 2014.

These methods can represent images with complex formulations, however they need to compute the distance between each pair of images. The very popular K-means clustering method and its variations belong to the second category since they optimize the distance of an image to a centroid vector, deemed the cluster representative. To determine the best centroid is not an easy task, furthermore the number of clusters must be given as input parameter. Statistical approaches model a cluster as a distribution and the database of images as mixture of these distributions.

In this paper, a new approch to cluster images, based on graph partitioning through Genetic Algorithms (GAs) [6], named *GA-IC* (Genetic Algorithms Image Clustering), is proposed. The image database is represented as a weighted graph where nodes correspond to images and an edge between two images exists if they are similar. The edge weight expresses the level of similarity of the feature vectors associated with images. A feature vector characterizes an image by capturing color and texture content. The graph of images is then clustered by applying a genetic algorithm that divides the graph in groups of nodes connected by many edges with high weight, by employing as fitness function the concept of weighted modularity [5]. It is worth to note that, although many evolutionary-based clustering approaches have been proposed, here we introduce a genetic algorithm to deal with the image database clustering problem, which is not always well solvable by traditional clustering techniques.

The paper is organized as follows. In the next section the problem of image clustering is defined, together with a description of the adopted feature selection method and similarity measure. Section 3 describes the algorithm, the employed fitness function, the genetic representation and operators. Section 4 presents the experiments, along with the evaluation measure used to assess the quality of the results. Finally, section 5 summarizes the approach and discusses future extensions.

2 Graph-Based Image Clustering

An image database DB can be represented as a weighted graph $G = (V, E, w)$, where V is the set of the nodes, E is the set of edges in the graph, and $w : E \rightarrow \mathcal{R}$ is a function that assigns a value to graph edges. Each node corresponds to an image in the image database, and an edge (i, j) connects two images i and j, provided that these two images are *sufficiently similar*. The weight $w(i, j)$ associated with an edge (i, j) expresses the similarity value between images i and j. Let W be the adjacency weight matrix of the graph G. Thus W_{ij} contains the weight $w(i, j)$ if the nodes i and j are similar, zero otherwise. Image clustering can thus be realized by partitioning the graph G in groups of densely connected nodes where edges have high weights. As pointed out in [3], any CBIR method has to deal with two crucial problems: the mathematical description of an image and the similarity measure used to compute how much similar two images are. In the next section we describe which features have been adopted to represent an image, and the similarity measure we employed to compute how much alike two images are.

2.1 Feature Description

The extraction and choice of which features represent at best an image is a long debated problem. In this paper we used features based on co-occurrence matrices and color centiles proposed in [7] and experimentally validated by Bianconi et al. [2] on a test suite of images. This is adopted to consider not only the grayscale textural but also the color channel content of the images. Each image is represented by a feature vector of numerical values describing both color and texture image content. The feature vector consists of fourteen different values: five are monochrome rotationally-invariant co-occurrence features, and nine are RGB centiles features. The five co-occurrence features represent the texture image content and the nine RGB centiles features codify the color image content [2]. The co-occurrence matrix P is created from the gray scale version of the image. It is composed of 256 rows and columns, which is the number of gray levels. Given an offset vector $d = (d_x, d_y)$, a single element $P(i, j)$ of P corresponds to the number of occurrences of the gray level values i and j, whose distance to each other is d inside the image plan. To guarantee a co-occurrence matrix which is invariant to rotation, the average of the number of occurrences computed at different offset vectors for each position $P(i, j)$ is considered. After that, the co-occurrence matrix P is normalized such that the overall sum of its elements is 1. Starting from the normalized P, a set of five features is computed: energy (E), contrast (Con), correlation (Cor), homogeneity (Hom) and entropy (H). The *energy* E is the sum of the squared elements in P: $E = \sum_{i,j} P(i,j)^2$. The *contrast* Con measures the intensity difference between a pixel and its neighbor in the image: $Con = \sum_{i,j} |i-j|^2 P(i,j)$. The *correlation* Cor expresses how a pixel is correlated to its neighbor in the image: $Cor = \sum_{i,j} \frac{(i-\mu_i)(j-\mu_j)P(i,j)}{\sigma_i \sigma_j}$, where ($\mu_i$, σ_i) are respectively the average and the standard deviation in the i (row) direction and (μ_j, σ_j) are respectively the average and the standard deviation in the j (column) direction inside P. The *homogeneity* Hom is the closeness of the distribution of elements in P to the diagonal: $Hom = \sum_{i,j} \frac{P(i,j)}{1+|i-j|}$. The *entropy* H measures the randomness of P: $H = -\sum_i \sum_j P(i,j) log_2 P(i,j)$. In order to obtain the RGB centiles features, the red (R), green (G) and blue (B) image histograms are computed. Each histogram contains the frequency of the image pixels at different intensity values in the given red, green or blue channel. After that, a cumulative channel histogram is derived from each channel histogram. In the cumulative histogram, each bin is the sum of all lower bins of the channel histogram. Centiles are computed from the normalized cumulative channel histograms $C_i(v)$, where i represents the channel, by detecting the intensity values v that split the cumulative channel histogram vertically into different parts. For example, the 20% centile of green channel corresponds to the intensity value of the green channel such that the 20% of all the pixels in the image is darker than this value. This intensity value is considered as a feature value. Three centile values are obtained from each cumulative channel histogram [7].

2.2 Similarity Computation

In order to determine the graph weights, we first need to compute the distance between two images. To this end, we used the L_1 norm because, as pointed out in [3], it is fast and one of the most popular measures adopted in image retrieval. Thus, given the feature vectors $I = [i_1, i_2, ..., i_n]$ and $J = [j_1, j_2, ..., j_n]$ associated with images i and

j, the distance between i and j is computed as: $d_{i,j} = \sum_{k=1}^{n} |i_k - j_k|$. The similarity $w(i,j)$ between images i and j is then obtained by applying the formula proposed by Gdalyahu et al. in [4]: $w(i,j) = e^{-(d_{i,j}^2/a^2)}$, where a is a local scale parameter defined by Gdalyahu et al. as the average distance to the second nearest neighbor. However, we propose to compute this scale parameter by using the distance of each image to its first nearest neighbor. The concept of first nearest neighbors, and more generally of *h-nearest* neighbors of an image i, analogously to [1] in the context of image segmentation, is introduced as follows.

$$d_{min}^h = \{d^1, \ldots, d^h \mid d^1 \leq, \ldots, \leq d^h\} \quad nn_i^h = \{j \mid d_{i,j} \in d_{min}^h\} \tag{1}$$

Given a generic image i in the graph, let d_{min}^h be the first h lowest distance values between image i and all the other images in the database. The h nearest neighbors of i, denoted by nn_i^h, are defined as the set of those images having minimum distance from i, i.e. maximum similarity with i. It is worth to note that the cardinality of nn_i^h can be greater than h since there can be more that one image j such that $d_{i,j} \in d_{min}^h$. The local scale parameter a is then computed. To obtain a sparse representation of the images contained in the dataset, we connect two images i and j only if j is among the h-nearest neighbors of i,

$$w_{i,j} = \begin{cases} e^{-\frac{d_{i,j}^2}{a^2}} & \text{if } j \in nn_i^h, i \neq j \\ 0 & \text{otherwise.} \end{cases} \qquad a = \frac{\sum_i d_i^1}{\mid D \mid} \tag{2}$$

Thus, if the distance between two images i and j is high, the corresponding weight $w(i,j)$ between these images will be low. On the other hand, if the value of distance is sufficiently weak, this happens when images are similar to each other in color and texture, the weight between the two images will be very high.

(a) African (b) Beaches (c) Horses (d) African (e) Beaches (f) Horses

Fig. 1. Six example images from three semantic classes: African people and villages, Beaches, and Horses

As an example, consider the image database DB composed of the six images in Figure 1. Each image is transformed into a 14-dimension feature vector. For example, the first image feature representation is I_1 =[0.001 0.001 0.99 0.37 0.76 0.25 0.42 0.61 0.17 0.33 0.47 0.16 0.28 0.43]. The L_1 distance between each couple of image features is computed and then the distance matrix D is calculated, as reported in Figure 2 (a). Fixed a h-neighborhood of 3, the nearest neighbors with the first 3 lowest distance values are computed for each image (Figure 2 (b)). Then, the a parameter is derived as the average distance of each image to its first nearest neighbor, as computed in the D

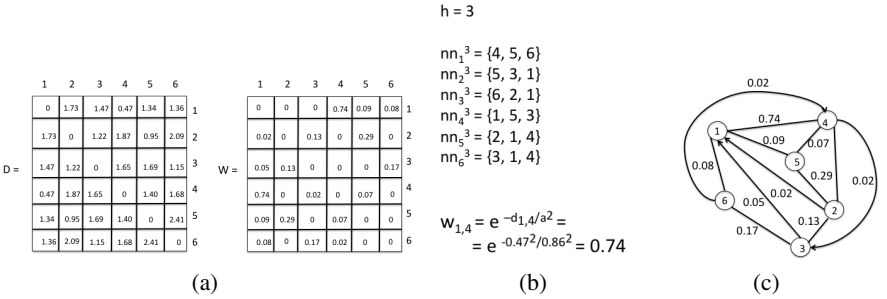

Fig. 2. Example of graph-based representation of an image database

matrix, i.e. $a = (0.47 + 0.95 + 1.15 + 0.47 + 0.95 + 1.15)/6 = 0.8563$. The similarity values are calculated as in Formula 2 from the corresponding distance values of the 3-nearest neighbors for each image row i in D. Finally, the similarity matrix W (Figure 2 (a)), is derived, and the corresponding graph in Figure 2 (c), where each node is an image in DB, is built from W, i.e. W is the weighted adjacency matrix of this graph.

3 Algorithm

In this section a description of the algorithm *GA-IC* is reported, along with the representation we used and the variation operators we adopted. The genetic algorithm uses the locus-based adjacency representation proposed in [8]. In this graph-based representation an individual of the population consists of n genes $g_1, \ldots, g_n$ and each gene can assume values in the range $\{1, \ldots, n\}$. Genes represent nodes of the graph $G = (V, E, w)$, and a value j assigned to the ith gene is interpreted as a link between images i and j. The initialization process assigns to each node one of its neighboring images at random. The kind of crossover operator we adopted is uniform crossover. The mutation operator randomly assigns to a node i, chosen in a random way, one of its neighbors. To better understand the genetic operators, consider the graph reported in Figure 2(c) built from the six images depicted in Figure 1. Figure 3(a) shows an example of individual initialized at random in which node 1 is connected with node 6, node 2 with node 3, and so on. This initialization corresponds to the division of the six images in 3 clusters composed by {1,6}, {2,3}, and {4,5}. Uniform crossover (Figure 3(b)) considers a random binary mask and, from two parents, generates a child having the value of the first parent if the mask is 1, while the value of the second parent if the mask is zero. Mutation (Figure 3(b)) changes the first parent of the crossover operator by substituting the neighbor of node 6 from 3 to 1, thus generating two new clusters composed by {1,3,4,6} and {2,5}. The fitness function we adopted to obtain groups of images densely connected with edges having high weights, and sparse low weighted edges between groups, is the *modularity* introduced by Girvan and Newman [5] and suitably modified for weighted graphs.

The weighted modularity is defined as $Q = \frac{1}{r}\sum_{ij}(W_{ij} - \frac{k_i k_j}{r})\delta(C_i, C_j)$, where $W_{i,j}$ is the weight of the edge (i, j), r is the total weight of all edges, k_i is the total weight

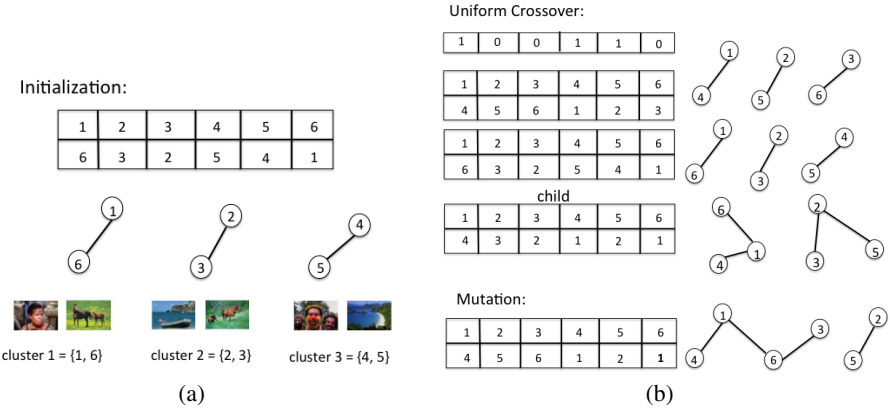

Fig. 3. Example of genetic operators: (a) Initialization, (b) crossover and mutation

of edges adjacent to node i, C_i is the cluster to which i belongs to, $\delta(C_i, C_j)$ is 1 if the images i and j belong to the same cluster, 0 otherwise. The algorithm thus creates a random initial population by connecting each image to one of its nearest neighbor similar images, and, for a fixed number of generations, applies crossover and mutation operators, evaluates the fitness of each individual, and generates a new population. At the end of the evolutionary process, the individual with the best fitness provides the graph partitioning, where each partition is a cluster of similar images.

4 Experimentation

In this section we present the results of *GA-IC* on the Wang image database and compare the performance of our algorithm with other four methods. The *GA-IC* algorithm has been written in MATLAB 7.14 R2012a, using the Genetic Algorithms and Direct Search Toolbox 2. In order to set parameter values, a trial and error procedure has been employed and then the parameter values giving good results for the benchmark images have been selected. Thus we set crossover rate to 0.9, mutation rate to 0.5, elite reproduction 10% of the population size, tournament selection function. The population size was 700, the number of generations 200. The value h of nearest neighbors has been fixed to 9, after a proper tuning of values in the range $[1, \ldots, 100]$.

The Wang image database has been created at the Pennsylvania State University. It is a manually selected subset of the Corel stock photo database, well known in Computer Vision, available at the website http://wang.ist.psu.-edu/docs/related/. The dataset is composed of 1000 color test images, divided into ten semantic classes, each composed of 100 images. The ten image classes are: African people and villages, Beaches, Buildings, Buses, Dinosaurs, Elephants, Flowers, Horses, Mountains and glaciers, Food. Images are of size 256×384 or 384×256 in jpeg format. Example figures of each class are shown in Figure 4.

(a) African (b) Beaches (c) Buildings (d) Buses (e) Dinosaur

(f) Elephants (g) Flowers (h) Horses (i) Mountains (j) Food

Fig. 4. Example images from the Wang dataset, one for each semantic class: African people and villages (a), Beaches (b), Buildings (c), Buses (d), Dinosaurs (e), Elephants (f), Flowers (g), Horses (h), Mountains and glaciers (i), Food (j)

4.1 Evaluation Measures

Image clustering is a retrieval problem, thus the popular evaluation measures of *precision*, *recall*, and *f-measure* are adopted to test the performance of the methods and to compare the results of *GA-IC* with other clustering approaches.

Precision P is the ratio of the number of retrieved relevant images to the total number of retrieved images: $P = \frac{Tot.\ \#\ retrieved\ relevant\ images}{Tot.\ \#\ retrieved\ images}$. Recall R is the ratio of the total number of retrieved relevant images to the total number of relevant images: $R = \frac{Tot.\ \#\ retrieved\ relevant\ images}{Tot.\ \#\ relevant\ images}$. The *F-Measure* F is the harmonic mean of precision and recall: $F = 2\frac{P \times R}{P+R}$. Since we deal with a multi-class problem, i.e. the dataset of images contains a number $k \geq 2$ of different image categories, and we can obtain any number of clusters, we need to specify what we mean by relevant images for each category. To this end we create a confusion matrix CM where the rows $\{g_1, \ldots, g_{k_{true}}\}$ correspond to the k_{true} ground-truth image groups contained in the dataset, and the columns $\{c_1, \ldots, c_{k_{pred}}\}$ to the k_{pred} clusters obtained by the algorithm. $CM(i,j)$ counts the number of images of cluster j appearing in the ground-truth group i. We assume that the predicted cluster c_j corresponding to the ground-truth group g_i is that containing the maximum number of images belonging to g_i.

$$c_j = arg\ max_j CM(i,j) \quad P_i = \frac{tp_i}{tp_i + fp_i} \quad R_i = \frac{tp_i}{tp_i + fn_i} \tag{3}$$

The above measures P_i and R_i are then computed for each ground class g_i, where tp_i is the number of true positive images, i. e. those images contained in the predicted cluster c_j corresponding to g_i, contained also in g_i, and fp_i is the number of false positive images, i.e. those images contained in c_j but not appearing in g_i. fn_i is the number of false negative images, i.e. those images of class g_i not predicted by c_j.

4.2 Experimental Results

In order to compare the results of *GA-IC* with other clustering paradigms, we run the well known K-Means method, a classical average linkage hierarchical clustering, the

Table 1. Precision, Recall and F-measure computed for each ground class of the Wang dataset for the image clustering approaches: *GA-IC*, Block Truncation Algorithm (BTC), K-Means, average linkage hierarchical clustering and Local Scaling clustering with co-occurrence descriptors and color centiles as image features. *nClusters* is the number of clusters obtained by the algorithms, *classes* is the name of the specific ground class in the Wang dataset. Values in bold correspond to cases in which *GA-IC* outperforms the other techniques.

	nClusters	classes	Precision	Recall	F-measure
GA-IC	13	African people and villages	0.3870 (0.0560)	0.4490 (0.0849)	0.4082 (0.0362)
		Beaches	0.1693 (0.0108)	0.3480 (0.0377)	0.2272 (0.0133)
		Buildings	0.2127 (0.0087)	0.4380 (0.0518)	0.2855 (0.0131)
		Buses	0.5688 (0.0048)	0.5450 (0.0135)	0.5566 (0.0091)
		Dinosaurs	0.3750 (0.0940)	0.4130 (0.0067)	0.3862 (0.0595)
		Elephants	0.3375 (0.0367)	0.3700 (0.0271)	0.3509 (0.0200)
		Flowers	0.4427 (0.0453)	0.4870 (0.0503)	0.4611 (0.0360)
		Horses	0.6614 (0.0272)	0.5770 (0.0841)	0.6141 (0.0609)
		Mountains and glaciers	0.1943 (0.0591)	0.3320 (0.0736)	0.2417 (0.0604)
		Food	0.2456 (0.1873)	0.3380 (0.0476)	0.2567 (0.0626)
BTC	10	African people and villages	**0.3358**	**0.4400**	**0.3809**
		Beaches	0.4242	0.4200	0.4221
		Buildings	**0.0792**	**0.0800**	**0.0796**
		Buses	**0.4483**	**0.5200**	**0.4815**
		Dinosaurs	0.9706	0.9900	0.9802
		Elephants	0.4483	0.3900	0.4171
		Flowers	0.9412	0.8000	0.8649
		Horses	**0.5882**	**0.5000**	**0.5405**
		Mountains and glaciers	0.4321	0.3500	0.3867
		Food	**0.2232**	**0.2500**	**0.2358**
K-means	10	African people and villages	**0.3621 (0.0536)**	**0.3660 (0.0386)**	**0.3604 (0.0273)**
		Beaches	0.1730 (0.0327)	**0.2330 (0.0427)**	**0.1978 (0.0345)**
		Buildings	0.3164 (0.0622)	**0.3780 (0.0312)**	0.3422 (0.0461)
		Buses	0.5885 (0.0220)	**0.5220 (0.1146)**	**0.5440 (0.0704)**
		Dinosaurs	**0.3737 (0.0800)**	**0.3770 (0.0696)**	**0.3752 (0.0750)**
		Elephants	**0.3358 (0.0342)**	**0.3520 (0.0469)**	**0.3427 (0.0373)**
		Flowers	0.4496 (0.0155)	**0.4590 (0.0694)**	**0.4505 (0.0337)**
		Horses	**0.5353 (0.1587)**	**0.3840 (0.0412)**	**0.4395 (0.0876)**
		Mountains and glaciers	0.2611 (0.0128)	**0.2790 (0.0277)**	0.2690 (0.0144)
		Food	**0.2373 (0.0428)**	**0.3160 (0.0420)**	0.2699 (0.0378)
Hierarchical	10	African people and villages	**0.2632**	**0.4000**	**0.3175**
		Beaches	**0.1346**	0.5100	**0.2129**
		Buildings	**0.1372**	0.5200	**0.2171**
		Buses	0.5800	**0.5800**	0.5800
		Dinosaurs	0.4100	**0.4100**	0.4100
		Elephants	**0.3208**	**0.3400**	**0.3301**
		Flowers	0.4623	0.4900	0.4757
		Horses	**0.1240**	**0.4700**	**0.1962**
		Mountains and glaciers	0.2632	0.4000	0.3175
		Food	**0.1451**	0.5500	**0.2296**
LS clustering	10	African people and villages	**0.2205 (0.0061)**	**0.4370 (0.0048)**	**0.2931 (0.0064)**
		Beaches	0.1795 (0.0036)	0.3560 (0.0135)	0.2386 (0.0061)
		Buildings	**0.1795 (0.0029)**	**0.3560 (0.0126)**	**0.2386 (0.0054)**
		Buses	0.5882 (0.0000)	**0.3000 (0.0000)**	**0.3974 (0.0000)**
		Dinosaurs	**0.1677 (0.0032)**	**0.2580 (0.0063)**	**0.2033 (0.0043)**
		Elephants	**0.3251 (0.0019)**	0.3710 (0.0032)	**0.3466 (0.0024)**
		Flowers	0.4557 (0.0013)	0.5200 (0.0000)	0.4858 (0.0007)
		Horses	0.6741 (0.0067)	**0.3680 (0.0042)**	**0.4761 (0.0032)**
		Mountains and glaciers	0.2476 (0.0019)	0.4910 (0.0110)	0.3292 (0.0037)
		Food	0.2623 (0.0035)	0.4230 (0.0067)	0.3238 (0.0035)

clustering algorithm based on local scaling proposed in [10], and the *Block Truncation Algorithm (BTC)* proposed by Silakari et al. [9].

The local scaling clustering algorithm [10] is a spectral method, based on the computation of an affinity matrix W, where a value at position (i, j) inside W represents the similarity between the points i and j. The feature vectors associated with images are the same of those used in our approach, i.e. co-occurrence descriptors and color centiles. Moreover, affinity computation is performed similarly to our formula (2). However, while we evaluate the affinity values only between the data points and their h-nearest neighbors, in [10] the affinity values are computed for each pair of data points. Specifically, in formula (2) the a parameter between two data points i and j is calculated as the average distance of all the data points to their corresponding first nearest neighbors and it is always the same. In [10], given an affinity value $W(i, j)$, this a parameter is computed for the specific data point i as $\sigma_i \sigma_j$, where σ_i (σ_j) is the distance of i (j) from its k-nearest neighbor.

BTC applies the k-means algorithm to images represented by features obtained by using the concepts of color moment and block truncation coding. In particular, the color distribution in each image is considered as a probability distribution, whose moments (mean, standard deviation and skewness) can be used as color features. Based on this concept, the algorithm calculates the Red, Green and Blue components from the original input image. Mean, standard deviation and skewness are computed for each component, and then each component is split in the color component of all pixels in the image which are above and below the corresponding mean. In such a way, for each of the three colors, six informations are computed to obtain a final feature vector of 18 components. The k-means algorithm is then applied to group the image feature vectors into clusters. The authors showed that on the Wang dataset the algorithm performs better, in terms of retrieval precision/recall, than the k-means algorithm with only color moments.

The results obtained by *GA-IC* and the other contestant methods are reported in Table 1. For each method we report the number of clusters found by each algorithm, and, for every class image, the precision, recall and f-measure values. Note that for the comparison methods the number of clusters must be given as input parameter. *GA-IC*, instead, automatically computes this value by evolving the population. *GA-IC* has been executed ten times, thus Table 1 reports the values averaged over these 10 runs with standard deviation in parenthesis. Values in bold on the rows of the four methods we compare with, correspond to those cases in which *GA-IC* outperforms the other techniques. From the table we can observe that the values of precision, recall, and f-measure obtained by *GA-IC* are better than those found by BTC for 5 out of 10 classes. As regards K-means, *GA-IC* obtains always the highest recall values and better precision for 5 out of 10 cases, and 7 out of 10 cases for f-measure. Furthermore, the hierarchical approach outperforms *GA-IC* only on four classes as regards precision and f-measure values, and on 5 classes for recall values. Finally, the clustering algorithm based on local scaling is superior to *GA-IC* on six classes as regards precision, on five classes as regards recall but only on four classes for f-measure. It is worth to note that the number of clusters obtained by *GA-IC* does not differ too much from the true number of ground truth groups. These results show that the genetic algorithm is a very promising approach to cluster images in groups of homogeneous images.

Furthermore, the modularity value of *GA-IC* has been computed on the graph representing the image database, averaged on ten different solutions found from the algorithm. A value of 0.7586 reveals that the clusters obtained from the algorithm well capture the image graph structure, demonstrating the high quality of the found clustering.

5 Conclusions

The paper proposed an approach to image clustering based on graph partitioning with genetic algorithms. The image database is represented by a graph where images correspond to nodes, and two images are connected provided that they are sufficiently similar. The similarity concept we introduced relies on the nearest neighbor images of a given image. The lower the distance between two images, the higher the weight of the edge connecting them. Experiments on a standard image dataset show very promising results, comparable with other state-of-the-art approaches. Future work will aim at investigating different content based features to improve evaluation indexes. Furthermore, an image database summarization approach will be provided by detecting the centroids from each retrieved image cluster.

Acknowledgements. This work has been partially supported by the project *MERIT : MEdical Research in Italy*, funded by MIUR.

References

1. Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. In: da Vitoria Lobo, N., Kasparis, T., Roli, F., Kwok, J.T., Georgiopoulos, M., Anagnostopoulos, G.C., Loog, M. (eds.) S+SSPR 2008. LNCS, vol. 5342, pp. 287–297. Springer, Heidelberg (2008)
2. Bianconi, F., Harvey, R., Southam, P., Fernandez, A.: Theoretical and experimental comparison of different approaches for color texture classification. Journal of Electronic Imaging 20(4), 043006–043006-17 (2011)
3. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Survey 40(2), Article 5 (2008)
4. Gdalyahu, Y., Weinshall, D., Werman, M.: Self-organization in vision: Stochastic clustering for image segmentation, perceptual grouping, and image database organization. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(10), 1053–1074 (2001)
5. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. In: Proc. National. Academy of Science. USA 99, pp. 7821–7826 (2002)
6. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston (1989)
7. Niskanen, M., Silvén, O., Kauppinen, H.: Color and Texture Based Wood Inspection With Non-Supervised Clustering. In: Proc. of the 12th Scandinavian Conference on Image Analysis, pp. 336–342 (2001)
8. Park, Y.J., Song, M.S.: A genetic algorithm for clustering problems. In: Proc. of 3rd Annual Conference on Genetic Algorithms, pp. 2–9 (1989)
9. Silakari, S., Motwani, M., Maheshwari, M.: Self-organization in vision: Stochastic clustering for image segmentation, perceptual grouping, and image database organization. International Journal of Computer Science 4(2), 31–35 (2009)
10. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: NIPS (2004)