Dynamic Learning of Indexing Concepts for Home Image Retrieval

Stéphane Bissol, Philippe Mulhem, Yves Chiaramella

DYNAMIC LEARNING OF INDEXING CONCEPT FOR HOME IMAGE RETRIEVAL

Bissol Stéphane[1&2], Mulhem Philippe[1&2], Yves Chiaramella[3]

[1]National University of Singapore
School of Computing
2 Science Drive 2
Singapore 117543
dcsbs@nus.edu.sg

[2]IPAL-CNRS
21 Heng Mui Keng Terrace
Singapore 199613
mulhem@i2r.a-star.edu.sg

[3]CLIPS-IMAG
385 rue de la Bibliothèque
38041 Grenoble Cédex 9
chiara@imag.fr

ABSTRACT

This paper presents a component of a content based image retrieval system dedicated to let a user defines the indexing terms used later during retrieval. A user inputs a indexing term name, image examples and counter-examples of the term, and the system learns a model of the concept as well as a similarity measure for this term. The similarity measure is based on weights reflecting the importance of each low-level feature extracted from the images. The system computes these weights using a genetic algorithm. Rating a particular similarity measure is done by clustering the examples and counter-examples using these weights and computing the quality of the obtained clusters. Experiments are conducted and results are presented on a set of 600 images.

1. INTRODUCTION

The large spreading and decrease in prices of digital camera, scanners and storage media has led to the proliferation of digital images. A typical owner of such devices easily possesses thousands [1] of these digital pictures and this number is perpetually increasing, and the problem of being flooded by this amount of images arises when users want to retrieve specific photographs. By default, consumer photographs are most of the time not, or poorly, indexed, leading to arduous browsing and retrieval. Content-based image retrieval (CBIR) systems [2] intend to provide solutions for browsing and retrieval of visual data. Such systems are dedicated to automatically (or semi-automatically) index images and make possible their retrieval via a search engine. CBIR systems usually impose an arbitrary and fixed retrieval scheme to the user, such as giving an example image for querying or terms from a predefined dictionary. Our proposal enables a user to make the system learn new indexing terms dynamically (e.g. "Indoor", "Underwater picture" or "Portrait"), allowing personalized retrieval.

The system first extracts bw level features from the images in the base, as usual CBIR system do. Then the user can create new indexing terms. The approach taken here consists in defining these terms using examples and counter examples images provided by the user. In the context of real consumer applications, we know from existing work on text retrieval that: 1) users ask short queries [12], and 2) users use an average of 13 documents to expand their query using relevance feedback [13], and in our case the example/counter-example sets are like relevance feedback input from a user perspective. So we know that the input from the user will not be large and we have to provide solutions for this. To each dynamically defined term corresponds a model and a custom similarity metrics. Indexing of new images is achieved by first extracting the low level features, then comparing them to the models created by the user, according to the learnt similarity metrics. Once the user has made the system learn a new term, it becomes an indexing feature for all the pictures like any other predefined indexing feature, and may be used for further learning of new indexing terms.

This paper is organized as follows. After introducing some related works in section 2, we present in section 3 an overview of our proposal before entering the details of the low-level features used (section 4), genetic algorithm (section 5) and clustering process used (section 6). Experiments and results are described in section 7. We eventually conclude in section 8 and outline some future works and improvements.

2. RELATED WORK

Most of the CBIR systems are based on the same framework [3]: they contain a feature extraction module, an indexing module (sometimes using machine learning techniques) and a retrieval module. Such systems differ on which features are extracted, what features are used for indexing the images (and possibly what learning is used for indexing), what a query input is provided by the system and what similarity measures are used.

QBIC [4] abstracts the images according to three features: color, shape, and texture. The color feature is represented by color histograms. QBIC uses an predefined color similarity matrix to determine the similarity between two color histograms in RGB space. Several heuristic geometric features such as area, circularity, eccentricity, and moment invariants compose the shape feature. The texture feature is described by three characteristics: coarseness, contrast, and directionality. As the shape feature, the similarity between two texture features is based on weighted Euclidean distances. There is no learning in the system as indexing only involves mapping the image into a feature vector. The possibility is given to the user to retrieve images using color, shape, texture features or by drawing a sketch. The choice of the features used in a specific context is made a priori during the creation of the database, like in the *State Hermitage Museum* [The State Hermitage Museum, http://www.hermitagemuseum.org] where only colors are considered. Other CBIR systems like VisualSEEk [5] or Netra [14] are also based on low level predefined features (color/texture/spatial relationships between objects) and predefined matching to retrieve images. CIRES [6], from the University of Texas, extend the works above by proposing a notion of image structure obtained via perceptual hierarchical grouping. The histogram intersection measure is used for color comparison. A weighted linear combination of the distances in the product space of structure, color and texture is used for retrieval. In CIRES, all the weights of the linear combination are predefined. To improve the retrieval, a system like Simplicity from Stanford University [7] first classifies images in several classes such as photography, drawing, textured, none textured. These categories are learnt from a training set and a matching metric is associated with each category. In Simplicity, the set of features for a given image category is determined empirically based on the perception of the developers.

In most of these systems, a user must either query the system using an example image or sketch, or provide predefined keywords (like in [15]) The problem we see in example query is the possible mismatch between the user and the system regarding the definition of similarity. In the user mind, similarity may be based on colors, texture, an object contained in the image or the background, location where the picture was taken. In the case of a keywords query, there is no confusion on what the user wants but, on the other hand, the query space is limited to a combination of a predefined indexing vocabulary. This vocabulary doesn't necessarily reflect the user needs. So our work is much more similar to the one of Minka in Four Eyes [16], where a user can input terms to image regions and features. In our case of home consumer application however, as we mentioned in the introduction, we have to propose new ways to overcome the need of big sets of learning samples using genetic algorithms.

3. SYSTEM OVERVIEW

We propose here a system in which a user defines her/his own indexing concepts which will be learnt by the system and be used to index all images in the database.

To do so, our system performs a concept-specific features selection by optimizing a similarity measure with a genetic algorithm (GA). We use Genetic algorithms because they are a way to tackle the large amount of possible similarity measures that can be generated on image features, in a way to find out a good (but not always the best, however) similarity measure that fit the description of the new indexing term. To measure the quality of the similarity measure, the genetic algorithm clusters some image examples and counter-examples and evaluates the resulting clusters. If examples and counter examples are well separated according to one similarity measure, this measure will be given a good 'mark'. This approach differs from the one known as 'Genetic Clustering' [8] in which the genetic algorithm optimizes the distribution of the elements in the clusters. In our system, the GA optimizes the similarity measure used to form the clusters. This work is more closely related to [9] in which a GA is used to select the weights of the features used to compare two Chinese character images.

Let us consider a user who has a term in mind; we assert that giving one image example of it is not enough to characterize this concept. Because of the large amount of information contained in a picture, to identify the user's focus requires more than one example. We design a CBIR system in which the user can make the system learn concepts from a few examples.

Figure 1 gives an overview of our system. The initial indexes (representations of the images) of images are obtained via low-level features extraction, and are extended after user input of new indexing terms. The user interacts with the system by providing examples and counter examples of a concept she/he wishes the system to learn. The learning process then builds a model of this concept. The index database is updated to take into consideration this new indexing term.

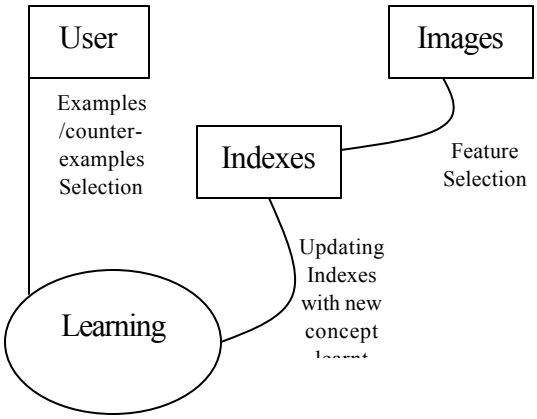

Figure 1. Overview of the system

The goal of the learning process in Figure 1 is to learn an appropriate similarity measure for a given indexing term. For instance, the indexing term *Underwater* should have a corresponding similarity measure which emphasizes the color features, particularly the features corresponding to the blue color. Whereas the indexing term *Night shot* might focus more on the intensity features.

The Figure 2 describes in more detail the learning process of Figure 1. The genetic algorithm generates similarity function candidates (represented by chromosomes) and assesses them by using a clustering algorithm. The goal of the clustering algorithm then is to group the examples and counter examples given by the user for the indexing term. Once the clustering process is achieved, the quality of its result is computed and used as the fitness measure for the chromosomes. The genetic algorithm creates new chromosomes until the required quality of clusters is obtained. The genetic algorithms then stops and: 1) the weights for the similarity settings corresponding to the best chromosome are saved as representative of the similarity function for the indexing term, and 2) the examples and counter-example images are also kept as representative for the indexing term.

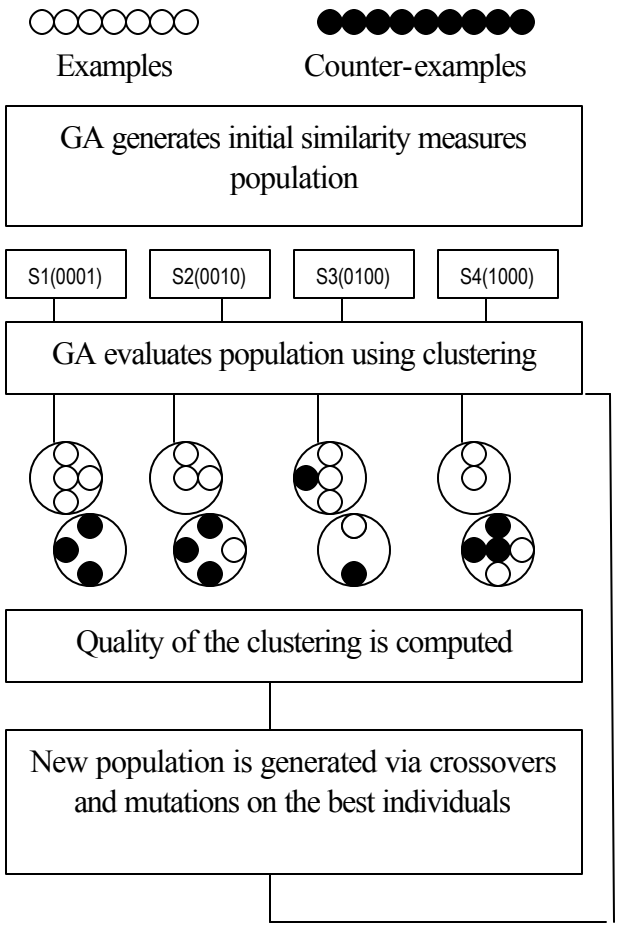

Figure 2. The learning process

4. EXTRACTING LOW-LEVEL FEATURES

The low-level features that are extracted currently in our system are mainly color-based and are describe only the whole image and not image regions. We consider however that the features used here are able to show the interest of our approach. Here are the extracted features:

- Normalized and quantified histograms extracted from the HSV color space (1 histogram per component). The HSV (Hue-Saturation-Value) space is used because of its similarities to the way humans tend to perceive color.
- *Connected* hue histogram. The special hue histogram goal is to focus on homogeneous regions of the image. The hue values are quantized in n groups and we build an n bins special histogram that stores the co-occurrence\ number of 4-connected pixels having the same quantized hue. As a result of this algorithm, larger bin values will correspond to the hue values appearing in homogeneous regions. This histogram is also normalized to 1.
- Bin number corresponding to the first 2 maximum values in connected hue histogram. These 2 values

are expected to describe the main regions in the image, like a large and uniform sky region or a close-up of a face.

- Average luminosity (computed from the value histogram).
- Mean saturation (computed from the saturation histogram).
- A histogram of the 7 Hue invariants computed from the image moments. Hue invariants are computed from the central and spatial moments of the image. Hue moments describe features like mass center of the image, orientations, and rough shapes. These features are proven to be invariant to scale, rotation and reflection.

5. THE GENETIC ALGORITHM

We remind that, in our system, the user inputs a new indexing term and gives examples of it, as well as counter examples. Based on these elements, the system finds out *why* the user did group the images this way. More specifically, this problem is translated in our case into finding the *ad hoc* weights for the similarity measure between images that best distinguishes between the examples and counter-examples provided.

Genetic algorithms are commonly used for optimization purposes, especially in case of NP-hard optimization problem which can't be solve by trying all potential solutions and where there's no efficient heuristic. As we explained earlier, the definition of a good similarity function among all the possible choices cannot be achieved through extensive process of the search space (the problem of optimizing a n parameters function is a NP-hard problem), that is why genetic algorithm are used here. An additional reason for us to consider the use of GA is that, in our context where users interact with the system, we have to provide fast responses from the system, so that in case of problem (bad learning for instance), the user does not hesitate to rerun the learning process with other images for instance.

We use here a standard genetic algorithm, as described in [10], to optimize the similarity measure between images used for the clustering. This similarity measure S has the form:

$$S(v1, v2) = \sum_{i=1}^{N} \alpha_i . \mathrm{dist}(v1_i, v2_i) \qquad (1)$$

Where N is the number of features (cf. section 4), v1 and v2 are the 2 features vectors corresponding to the description of two images, α_i are the weights assigned to the features i.

A genetic algorithm is based on the use of chromosomes that describe the search space. In our case, a chromosome is a configuration of weights defining a similarity measure (cf. equation (1)). Each chromosome has the following form:

$$Chromosome = (\alpha_1, \alpha_2, ..., \alpha_N)$$

Where each gene α_i codes a weight in the set {0,1}.

A genetic algorithm is also defined by a fitness function, indicating which chromosome is the more resistant. In our case, the fitness of the individual chromosomes of the population is the quality of a clustering performed on the examples and counter-examples using the similarity measure coded by the chromosome. We assume that the quality of a similarity measure depends on how well a clustering algorithm using this measure will separate examples and counter-examples in distinct clusters.

A genetic algorithm is also described by the size of its population (i.e. the number of chromosomes) and the way the initial chromosomes are defined. The initial chromosomes are usually defined randomly. In our case, we use as many chromosomes as the number of features, and they are not selected randomly. Let us consider N features, we force the N initial chromosomes to be chosen as: the chromosome i (1=i=N) will have all its genes initialized to 0, except the gene i, set to 1. This is based on the assumption that the similarity measure should be as simple as possible. Even if several similarity measures exist leading to a correct classification, the one involving the minimum number of features is considered as better. Indeed, if more features are taken into account it reduces the generalization ability of the learning.

6. CLUSTERING

A clustering algorithm is used to assess the quality of each chromosome representing a similarity function. We use a supervised clustering algorithm to group the examples and counter examples given by the user to learn a new indexing term. The quality of the obtained set of clusters is used to evaluate the fitness of the chromosomes in the GA. Unlike unsupervised clustering, we know what should be the optimal clusters composed of examples and counter-examples images representatives. Namely, a perfect set of clusters is one in which every cluster contains either only examples or only counter examples. An important aspect that all clustering algorithms have is the similarity measure used. The output of the clustering depends on this measure, on the number of clusters to be formed and to a less significant extent, on how the algorithm operates.

The clustering algorithm used here is the well known K-mean. It is known to yield to good results but performing poorly in terms of complexity, in the case of very large data set [11]. As the data set is provided manually by the user, we

expect it to have a low cardinality, as described in the introduction, i.e. less than 15 examples and counter-examples.

Given a set of clusters, we define the measure of its quality as:

$$Q = - \sum_{nbclusters} L(y, p)$$

With

$$L(y, p) = \begin{cases} 0 & \text{if } y = p \\ 1 & \text{otherwise} \end{cases}$$

L is the *zero*-one loss function [17] in which y is the label after clustering and p is the expected label.

The best quality is obtained when no example is in the same cluster as a counter example, and vice versa, leading Q being equal to 0. The worst case occurs if clusters contain an equivalent number of examples and counter example, and then Q is equal to zero minus the number of examples plus the number of counter-examples.

The K-mean algorithm needs two parameters: the first one is the number of clusters wanted, and the second is the selection process of the seeds as cluster centers. Instead of selecting the seeds randomly, we choose them from the set of examples and counter-examples so that they are the furthest from each other, like repulsive magnets. Our experiments show that results are improved using this initialization. In our problem, we do not know what is the best number of clusters, so we run the algorithm first with a two cluster model, and then in case of failure (i.e. loss function above a give threshold), we increment this number by one until success.

7. EXPERIMENTS

We conducted some experiments on a database of 600 pictures using our prototype called Citra (see figure 5). These pictures are non professional and are mainly indoor pictures of people and scenes, mountain sceneries, beach landscapes, forest/jungle and under-water pictures.

The figure 3 shows the recognition rate for 3 different concepts, namely "Underwater", "Sky" and "Night picture". The recognition rates are defined here as the *precision* value computed in the field of information retrieval, corresponding to the ratio of the number of correctly classified images divided by the number of images classified in that class by the system. To find out the impact of the size of the training set we made the system learn these concepts with an increasing number of examples and counter-examples selected randomly from the predefined sets of images from each class. As we can see on figure 3, the result of the indexing can vary quite significantly: for the two runs of Underwater images the standard deviation of difference between the recognition rate is 0.15 for an mean of 0.13, for the Night scenes the standard deviation is 0.09 for a mean value of 0.13, and for the images containing a Sky we have a standard deviation of 0.06 for a mean of 0.08.

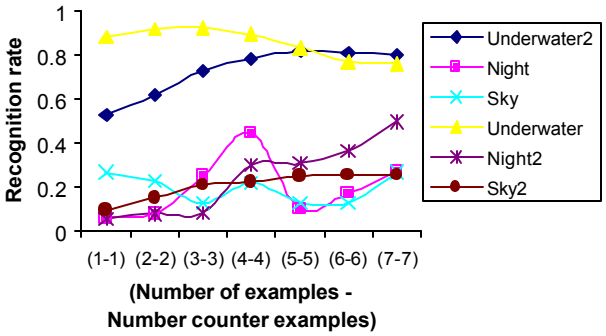

Figure 3: Recognition rates on two learning tests.

To overcome the variability of the results as presented in figure 3, we present in figure 4 the results obtained for the same indexing concepts averaged on several runs were the examples and counter-example were selected randomly from the predefined classes. By averaging several runs, we better reflects the fact that users will select examples and counter-examples based on parameters that we are not able to model (like her/his mood for instance).

We describe more precisely the results obtained. The low recognition rate for the concept "Sky" is explained by these two facts:

- The sky is usually only a region in the image; global features like those that we extract cannot focus on only a part of the image. Therefore, the rest of the image becomes noise in the extracted features.
- Skies in our database take many different appearances making the construction of a model arduous.

Results are better for the concept "Night picture". This concept concerns the whole image and is therefore well learnable by the system. We can see on figure 4 that increasing the size of the training set leads to a better recognition rate. Indeed, just like skies, night pictures can have different appearances depending for example on where the picture has been taken (inside or outside), if a flash was used or not etc. Hence, the more the training set contains facets of this concept, the better the system can retrieve different appearances of it. The recognition rate stabilizes though once the concept is fully described by enough samples.

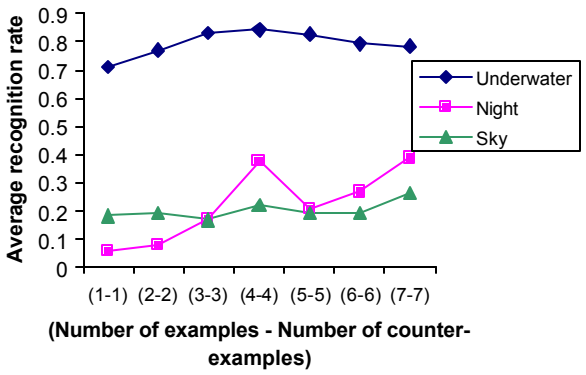

Figure 4: Recognition rates on several runs.

The recognition rate curve for "Underwater" is flatter than the others because images belonging to this concept have similar characteristics. Therefore these images do not have many distinct appearances and increasing the learning set size does not provide much more information to the system. Precision is good in this case because this concept suits our current framework: the concept underwater concerns the whole image (so global features can capture it well) and the appearance of such an image is well defined.

8. CONCLUSION

In this paper we have presented a new user-focused CBIR system. The user himself defines indexing and query terms and makes the system learn them by providing examples and counter examples. The learning is done using a clustering algorithm as the fitness function of a genetic algorithm. The genetic algorithm optimizes the similarity measure used in the clustering algorithm. If the learning is successful, the system records the similarity measure settings as well as the clusters formed and indexes all the images with this new feature. Retrieval is done using these terms.

Currently our system is limited by the fact that we only consider the whole image for features extraction. Hence, the system can learn concepts that imply the whole image but the objects level is not reached. Our next step is to consider regions in addition to global features, to enable queries on objects and their spatial relationships.

Our current implementation of the genetic algorithm only supports Boolean values. In the future we will make these values 'floats' in order to have more flexible weights in the similarity measurement.

As our approach performs an efficient feature selection, we must also add features so that the system could learn more complex concepts. Our current set of features is mainly based on colors. A lot of other meaningful ways to describe an image's content exist, focusing on shapes and textures. Including them in our system would probably lead to even better results.

9. REFERENCES

1. K. Rodden and K. Wood, How do People Manage Their Digital Photographs?, ACM Conference on Human Factors in Computing Systems Fort Lauderdale, April 2003.
2. A Review of Content-Based Image Retrieval Systems. Colin C. Venters and Dr. Matthew Cooper University of Manchester. www.jtap.ac.uk/reports/htm/jtap-054.html
3. A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R,. Jain, "Content-Based Image Retrieval at the End of Early Years," IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), vol. 22, no. 12, pp. 1349-1380, Dec. 2000.
4. M. Flickher, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D.Steele, and P. Yanker, "Query by Image and Video Content: The QBIC System," IEEE Computer, vol. 28, no. 9, pp. 23-32, Sept. 1995.
5. J. R. Smith and S.-F. Chang, Querying by color regions using the VisualSEEk content-based visual query system, In Intelligent Multimedia InformationRetrieval. IJCAI, 1996.
6. Q. Iqbal and J. K. Aggarwal, CIRES: A System for Content-based Retrieval in Digital Image Libraries , Invited session on Content Based Image Retrieval: Techniques and Applications International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, pp. 205-210, December 2-5, 2002.
7. J. Z. Wang, J. Li, and G. Wiederhold, "Simplicity: Semantics-sensitive integrated matching for picture libraries, " IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, pp. 947--963, 9 2001.
8. Demiriz, K.P. Bennett, and M.J. Embrechts. Semi-supervised clustering using genetic algorithms. R.P.I. Math Report No. 9901, Rensselaer Polytechnic Institute, Troy, New York, 1999.
9. Daming Shi, Wenhao Shu and Haitao Liu, "Feature selection for handwritten Chinese character recognition based on genetic algorithms", p.4201-6 vol.5, SMC'98 Conference Proceedings, 1998.
10. G. Harik, ``Linkage Learning via Probabilistic Modeling in the ECGA," IlliGAL Report No. 99010, University of Illinois at Urbana-Champaign, Urbana, IL, January 1999.
11. An Efficient K-Means Clustering Algorithm Tapas Kanungo, IEEE, David M.Mount, IEEE, Nathan S.Netanyahu, IEEE, Christine D.Piatko,Ruth Silverman, and Angela Y.Wu, IEEE

12. Budi Yuwono, Savio, L.Y. Lam, Jerry H. Ying, and Dik L. Lee. A World Wide Web resource discovery system. In Fourth International World Wide Web Conference, pages 145--158, Boston, 1995).
13. Koenemann, J., and Belkin, N. A case for interaction: a study of interactive information retrieval behavior and effectiveness, in Proceedings of CHI '96 Vancouver Canada, May 1996, ACM Press, 205-212.
14. Y. Deng, B.S. Manjunath, C. Kenney, M.S.Moore and H.Shin, An efficient color representation for image retrieval, IEEE Transactions on Image Processing, vol.10, (no.1), IEEE, Jan. 2001. p.140-147.
15. N. C. Rowe and B. Frew, Automatic Classification of Objects in Captioned Depictuve Photographs for Retrieval, Chapter 4 of Intelligent Multimedia Information Retrieval, M. T. Maybury Editor, AAAI Press/The MIT Press, 1997.
16. T. Minka, An Image Database Browser that Learns from User Interaction, MIT Technical Report TR#365, MIT, 1996.
17. R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, 1973, p. 16.

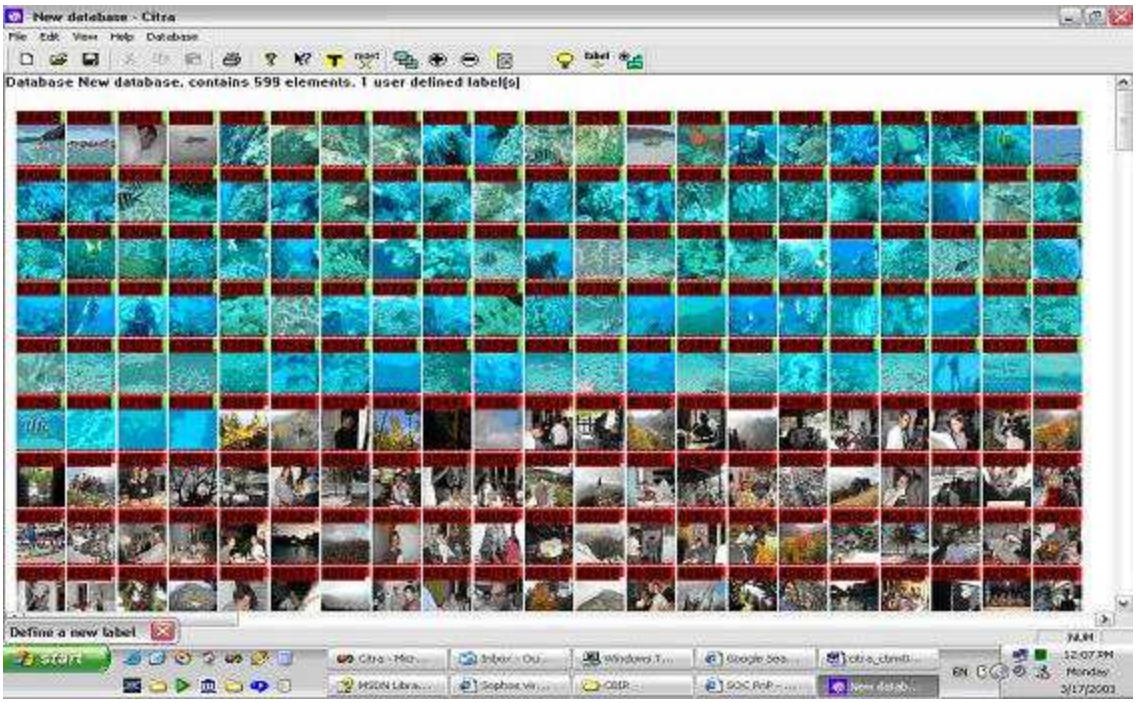

Figure 5: System interface after a query on "Underwater"