

RETIN system: partial and global feature learning ImagEval/Tasks 4 and 5

Sylvie Philipp-Foliguet
ETIS, CNRS
6 avenue du Ponceau
95014 Cergy Cedex, France
philipp@ensea.fr

Philippe-Henri Gosselin
ETIS, CNRS
6 avenue du Ponceau
95014 Cergy Cedex, France
gosselin@ensea.fr

Matthieu Cord
LIP6
104, av du P. Kennedy
75016 Paris, France
Matthieu.cord@lip6.fr

ABSTRACT

In this paper, we present the methods involved in the tasks 4 and 5 of the TECHNOVISION/ImagEval contest. Two techniques are used, a global and a local one, both based on a description of low level visual content by histograms of colors and textures. The global approach considers images as a whole, and classifies images using a single histogram for each image. Kernel functions and SVMs are used for the classification. The local approach works on graphs of fuzzy regions. It searches for images containing the object of interest using graph matching. The method has the specificity to compare fuzzy regions with a supervised and kernel based similarity function.

Keywords

1. INTRODUCTION

The task of image retrieval can be seen as a classification task especially when a learning set is available. Both task 4 and 5 of the ImagEval contest consist in retrieving image categories. Task 5 is of a lower semantic level, since categories are "day/night" images or "painting" etc. Thus low-level features describing the colour and the texture seem sufficient to discriminate between categories. For task 4, the problem is to retrieve images containing a class of object, such as "Eiffel Tower" or "cow". In this case, two philosophies can be applied, one considers that the semantic is carried by the whole image, which means that cows usually are in meadows whereas Eiffel Tower is surrounded by a town landscape. The other one considers that if we want to retrieve a cow whatever the surrounding (that is to say, in an agriculture show, as well as in a Delhi's street or in a meadow), the categorization must take into account the only object and not the background. Task 5 has been solved by a classification system based on global image signatures and for task 4 we have proposed two solutions, one based on global signatures and the other one based on local signatures, representing parts of the image.

Both solutions, whether global or local share the same way of representing colour and texture information and the same classification engine. They differ for the image representation, since for local approach, the image is first segmented into regions and the image signature is composed of region features (including colour and texture information) and of region layout. And consequently they differ for the metric used to compare images.

As examples of the searched categories were given for both tasks, we used them as learning sets in a classification task. Categories are learnt separately and the classification is performed one at a time. In this framework, linear and non-linear classifiers can be used, but they are efficient only if used in an appropriate representation space. This problem has been solved by the use of kernel functions.

We will first present the way we represent colour and texture information, in a global context. If we classically use histograms to represent both features, we emphasize the way we obtained the colour and texture codebooks. The same codebooks are used for the region description in the local approach.

2. GLOBAL QUERY (Task 4, runs 7 & 9 and Task 5, runs 7&8)

In this approach, images are represented by colour and texture histograms, and classified with statistical learning methods.

CIE $L^*a^*b^*$ space is used for colour, and twelve Gabor filters in 3 different scales and 4 orientations are used for texture analysis. Colour and texture vectors are extracted from images, and the whole set of these vectors on the training database is considered to build a visual codebook. As this training set is very large, we developed a two-stages quantization algorithm.

Once the visual codebook is built, each image can be represented by a histogram. The kernel functions framework is then used to map these histograms into a Hilbert space, where linear classification techniques such as SVMs can discriminate between images in the category and the others.

2.1 Visual codebook computation

Building a visual codebook is an effective mean of extracting the relevant visual content of an image database, which is used by most of the retrieval systems.

A first approach is to perform a static clustering, like [12] where 166 regular colours are *a priori* defined. These techniques directly provide an index and a similarity for comparing images, but the visual codebook is far from being optimal, except in very specific applications.

A second approach is to perform a dynamic clustering, using a standard clustering algorithm such as *k*-means [5]. In this case, the visual codebook is adapted to the database content. Using a *k*-means algorithm leads to a sub-optimal codebook, where some codewords may over (or under) represents the visual content. A usual way to find a good visual codebook is to train several times the clustering algorithm and to keep the best codebook. However, because of the large number of vectors to be clustered, this strategy is very time consuming.

In this section, we first study new alternatives to the standard *k*-means algorithm, and select the most attractive in terms of efficiency and time cost. Next, we address the problem of the quantization of a very large number of vectors, where standard clustering algorithms can not be directly applied, since the whole vector set can not be stored in memory. We propose a clustering algorithm which leads to a near to optimal codebook with only one training pass.

2.1.1 Dynamic quantization

Vector quantization aims at finding the optimal set of codewords that will clusters a set of vectors. The class of a vector is then defined by the closest codeword, for a given distance (usually a Euclidian distance).

The optimisation problem addressed by vector quantization is not convex – this means that the algorithm must find the global minimum between multiple local minima. The success of convergence is

mainly determined by the initial codebook. The standard *k*-means algorithm uses a random initial codebook, and thus converges to a local minimum. Improvements about the initialisation have been proposed, like the *k*-means splitting or LBG [7]. The algorithm starts with only one codeword, and step after step, splits the clusters into two sub-clusters. In [8], Patanè proposes ELBG, an enhanced LBG algorithm, introducing an heuristic in order to jump from a local minimum to a better one. This heuristic swaps codewords so that their respective distortions are as much equal as possible.

We implemented and compared the three methods for the quantization of the RGB vectors of the images of the ANN database. Table 1 shows the results in terms of average PSNR (log value of the distortion), and the average computation time for the quantization in 256 colours of one image.

PSNR values are of the same order, slightly better for ELBG than for LBG and standard *k*-means, but ELBG is much faster than LBG (4 times) and faster than standard *k*-means. For those reasons we have adopted the ELBG algorithm in our large quantization process.

Table 1. Performances and computational time of the quantization methods

Method	PSNR(dB)	Time(sec)
<i>k</i> -Means	37.87±2.76	12.35±1.55
LBG	37.90±2.57	31.99±11.03
ELBG	38.69±2.82	8.49±1.27

2.1.2 Quantization of large datasets

The second problem is the large amount of samples to classify. As it is impossible to put all pixels in memory at the same time, the method has to be progressive, that is to say able to manage data part by part.

Adaptive *k*-means processes samples one by one. This method imposes that samples are processed in the most possible random way. But this condition is hard to obtain in image indexing, since for time constraints, pixels cannot be processed completely randomly. At least for run-time and practical reasons, it is better to process each image as a whole.

Fournier [5] performs an adaptive k -means by sub-sampling each image: only a tenth of the pixels of each image randomly chosen are processed. To compensate this sub-sampling, images are processed ten times.

We propose an adaptive quantization by k -means in two stages, both performing ELBG method:

- The first stage quantifies each image independently using the ELBG algorithm;
- The second stage quantifies the whole database from the codebooks obtained at the first stage using a modified ELBG algorithm.

The advantage is that each image is processed independently in the first stage and even in a parallel way. The number of codewords in that stage can be of a rather large size (at least greater than any desired codebook for now and the future).

2.1.3 Image signature

Once the colour and texture codebooks are generated, image signatures are computed. For each pixel, the closest codeword is detected, and both – colour and texture based – distributions are generated for each image over the visual codebooks.

Resulting from the visual analysis of the database content, the global signature for Tasks 4 and 5 for each image is composed of a 50-length vector, gluing colour and texture distributions over database-dedicated codebooks.

2.2 Similarity using kernel functions

Once signatures are computed, a metric or a similarity function has to be defined to compare images.

Basically, the Euclidian distance is used to compute the similarity between histograms, or more generally a Minkowski distance. However, these metrics are not necessary relevant for histograms. Alternatives have been proposed, such as histogram intersections, entropy, or χ^2 distance.

These metrics independently compare each value of the histograms, and do not address the problem of correlation between axes. More robust metrics have been proposed to solve this, like Earth Mover's Distance [14], or generalized quadratic distances.

Whenever these metrics are efficient for histograms, they all lead to a non-linear problem, and, most of the

time, particular learning techniques must be developed to use them. In order to use powerful learning techniques that have been recently introduced [15], we have chosen to use the kernel functions.

2.2.1 Kernel framework

The approach consists in finding a mapping φ from an input space X (here our histogram space) to a Hilbert space H . Thus, once found this mapping, all the addressed classification problems become linear. The best example is the Support Vector Machine classifiers, which discriminate data using hyperplanes. Working in a linear space, it is possible to express the binary classification problem, and to use the fast and efficient algorithm of SVMs. Another interest of the kernel function framework is that the mapping function φ has not to be explicitly expressed. The idea is to work not directly on the mapped vectors, but on their dot products:

$$k(x_i, x_j) = \langle \varphi(x_i), \varphi(x_j) \rangle$$

The kernel functions framework is interesting only if one can find a mapping relevant for the application. In our case, since we are working on histograms, an interesting kernel function is the Gaussian one:

$$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2\sigma^2}\right)$$

This function depends on a distance $d(x_i, x_j)$, which allows us to pickup one of the distances for histograms. For instance, we used the χ^2 distance for the evaluation:

$$d(x_i, x_j) = \sum_c \frac{(x_{ci} - x_{cj})^2}{x_{ci} + x_{cj}}$$

Note that we could use more robust distances, such as the Earth Mover's Distance [14], but this leads to a too high computational cost for the processing of huge databases.

2.3 Classification

Performing an estimation of the query concept can be seen as a statistical learning problem, and more

precisely as a binary classification task between the search class and the rest of the database [1][2].

In image retrieval, many techniques based on statistical learning have been proposed, for instance Bayes classification, k -Nearest Neighbours, Support Vector Machines, Gaussian Mixtures, or Gaussian random fields.

We evaluated the following methods through cross-validation on the train dataset :

- Similarity Refinement;
- Bayes classification;
- k -Nearest Neighbours;
- Support Vector Machines;
- Transductive Support Vector Machines;
- Kernel Fisher Discriminant.

We found that the best choice on the ImagEval databases is the Support Vector Machines, the Transductive SVM and Kernel Fisher Discriminant, which have close performances. However, considering computational time, inductive SVMs are far the best algorithm.

3. PARTIAL QUERY (Task 4, run 3)

The other way to retrieve images containing an object is to be able to focus on the area containing this object. To do this, we segment the images into rough regions and the classification process is performed on graphs of regions. In order to be compatible with the semantic entities, the regions must be colour uniform, but they do not need accurate edges. They will be represented by fuzzy sets, which are sufficient for the task of pattern recognition. Task 4 of object retrieval consists in learning sets of regions, represented by adjacency graphs of regions and then in matching these graphs to sub-graphs of images of the test database.

3.1 Fuzzy segmentation

The regions we are going to build are limited by high gradient norms and uncertain when two (or more) regions encounter.

The algorithm performs a cooperative contour / region approach. Uniform colour regions are extracted by a region growing algorithm performed on the gradient norm image. Membership degrees to a region are linked to the distances to region seeds.

Let Ω be a finite referential (set of N pixels). A fuzzy region R_j is a fuzzy set of Ω defined by a mapping μ_j from Ω to $[0, 1]$.

Definition: A *fuzzy segmentation* of Ω is a set of M fuzzy regions R_j whose supports are included in Ω and defined by the two following axioms. If $\mu_j(s)$ is the membership degree of pixel s to region R_j , then:

1. $\forall s \in \Omega, \forall j, \mu_j(s) \in [0, 1]$
2. $\forall j, 0 < \sum_{s \in \Omega} \mu_j(s) < N$.

Membership degrees are included between 0 and 1, they equal 1 for the core's pixels and 0 for the pixels that do not belong to the fuzzy region. The second axiom means that a fuzzy region must neither be empty nor complete (equal to Ω).

This definition is based on Ruspini's definition of a fuzzy partition [11], but without the third axiom that imposes that for each pixel, membership degrees to all regions sum up to 1.

The algorithm first computes a watershed algorithm on the colour gradient norm image [16]. Every local minimum of the gradient norm is a seed of a basin. This leads to a big number of basins. Thus in a second step, basins are recursively merged according to their areas and depths : a basin too small or not deep enough is absorbed by a bigger neighbour. In this case, the difference between the bottom levels of both basins is applied as a penalty on membership degrees of the absorbed one.

Each set of merged basins gives rise to a fuzzy region. Membership degrees are calculated from the "topographic distance" defined in [9].

For an easier computation, considering common images of about a few hundreds of pixels in each dimension and a standard contrast, membership degrees take values between 0 and 255.

They are initialized to 255 for pixels belonging to basin seeds (perhaps minus the difference of basin bottom levels). They decrease as pixels are going away from seeds : 1 for each spatial step (in 8-connectivity), and a value proportional to the difference between gradient norms, until they reach 0.

Membership degrees to region R

(build from gradient norm image g)

For each basin B of R

for each pixel $s \in \text{seed}(B)$

$\mu_B(s) \leftarrow 255 - \text{penalty}$

put s into Q_B

end for

while Q_B is not empty

extract s of Q_B

for each pixel v neighbour of s

$\mu = \mu_B(s) - (\lambda \cdot |g(v) - g(s)| + 1)$

if $\mu > \mu_B(v)$ then $\mu_B(v) \leftarrow \mu$

put v into Q_B

end for

end while

End for

So pixels belonging to areas of a homogeneous colour, and thereby of a small gradient norm have large membership degrees in the corresponding fuzzy region (cf. Figure 1). These degrees slowly decrease according to the spatial distance to the seed and strongly decrease when meeting an edge, area of a high gradient norm. Moreover the impulse noise is bypassed, because a shorter path is found "around" it.

A crisp segmentation can be obtained by affecting every pixel to the region for which it has the largest membership degree. This "defuzzification" is only used to display simultaneously all the fuzzy regions.

Parameter λ aims at balancing spatial distance with the closest region seed and difference of gradient. It will influence the spreading of regions. It has been set to 2 for all the following tests. The area threshold for basin merging induces the level of detail of the result. It is not beforehand set, but will increase or decrease from an initial value, until an expected number of regions, defined by an interval, is reached. It has been set to [2, 8] for the learning set which roughly contains the only object, and to [5, 20] for the test database, in which the object represents only a part of the image.

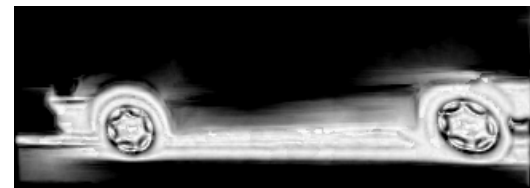
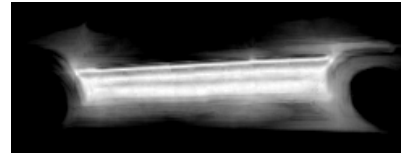


Figure 1. An example of a car and the 3 fuzzy regions representing this car (the whiter, the higher the membership degree)

3.2 Image signature

Each image of the database is indexed by a signature, which is composed of two parts : the first one is the set of features of all fuzzy regions, and the second one is a representation of the topology of the regions within the image.

In this section, membership degrees take values between 0 and 1, as usual.

For indexing regions, we used colour and texture distributions. Colour and texture codebooks are the same as for the global query (section 2.1), except that they are composed of 100 codewords for colour and 100 codewords for texture. To index fuzzy regions, the contribution of each pixel is weighted by its membership degree to the region.

The distribution of feature a for fuzzy region R , defined by its membership function μ , is defined by the probability for any real t [4] :

$$P_a(t) = \frac{\sum_{s \in R, a(s)=t} \mu(s)}{\sum_{s \in R} \mu(s)} \quad (1)$$

Distributions are thus obtained by adding the membership degrees of the pixels belonging to each class and a normalisation is performed by dividing by the region area. Thus pixels with small membership degrees – belonging to transitions or outliers inside a region – have little influence on the distribution shape.

Spatial relationship between regions are characterized by two ways. First their barycentre are computed as in [10] and normalised by the dimensions of the image, in order to take values between 0 and 1. Secondly adjacencies are stored in an adjacency matrix of regions with a value of 1 if both regions have at least one pixel in common, otherwise 0.

3.3 Graph matching

We are interested in a query constituted by a set of regions, adjacent or not. The problem is to retrieve images represented by sets of regions which match at best this query.

After segmenting into regions, each image of the database is represented by an attributed relational graph (ARG). Nodes correspond to regions and edges to adjacencies between regions. Each region is characterised by a set of features (colour, texture, ...) and edges represent adjacencies between regions and are characterised by spatial information such as “above”, “on the left of”, overlap, etc.

Let us first suppose that the query is made of only one example of the object we want to retrieve. This allows to explain our graph matching strategy. The query is thus an adjacency graph of regions and the search consists in looking for the sub-graph which matches the best the query ARG (cf. Figure 2), in each image of the database.

We solve both the problem of detection and of localisation of an object in an image.

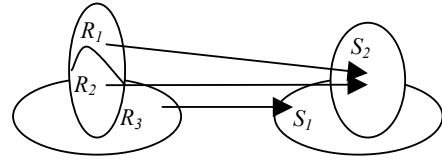


Figure 2. Example of an inexact match between two graphs

This problem is a problem of graph matching, but not limited to graph isomorphism, since one node of a graph may be matched to several nodes of the other graph (cf. Figure 2). Two matched subgraphs have not to be identical in terms of node number, node attributes and edge number but only similar for node attributes (measured by an appropriate similarity measure expounded below) and consistent for the edges. The relative positions of regions are taken into account during the matching in order to reduce the combinatorial. The problem we address is NP-complete, but the solution has to be found in an acceptable time for a user making a search in the database.

To solve this problem we used a tree structure (cf. Figure 3) inspired from [3]. The query regions are denoted $R_i, i = 1, \dots, m$ and the set of regions of a target image are denoted $S_j, j = 1, \dots, n$.

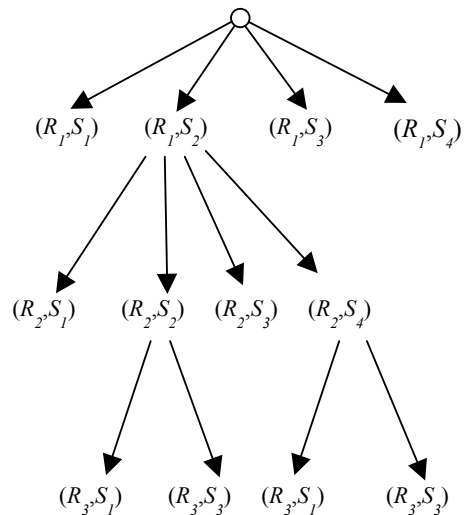


Figure 3. Search tree: each node corresponds to a match of two regions (R_i : query region, S_j : candidate region), each arrow corresponds to a topology compatibility

Each node of the tree represents a possible match between a query region R_i and a target region S_j . Each level of the tree corresponds to one query region. Edges between nodes represent topological consistency between pairs of regions. For example, the consistency between (R_i, S_j) and (R_{i+1}, S_k) can take into account :

- the adjacency of (R_i, R_{i+1}) and (S_j, S_k) which must be of the same type (adjacent or not);
- the relative position of (R_i, R_{i+1}) and (S_j, S_k) , for example if R_i is above R_{i+1} , S_j must be above S_k .

The advantage of such a structure is that with appropriate heuristics, all possibilities of matches of region pairs have not to be explored. Search is concentrated on branches which are susceptible to lead to the solution. Heuristics concerns the order of examination of the region pairs, and takes into account distances between regions and topological consistency. Moreover it can be extended to the comparison between a graph and a set of graphs in a classification task, as we will see below (section 3.4). We obtain an optimal (or sub-optimal) solution according to a given similarity function.

A match between two sub-graphs corresponds to a path from the root to a leaf. The dissimilarity between sub-graphs is the weighted sum of the dissimilarities for all nodes of a path :

$$\frac{1}{m} \sum_{i=1}^m w_i d(R_i, S^i)$$

Where S^i is the target region matched with R_i , d represents the dissimilarity between region features, w_i is a non negative weight and m is the number of query regions.

As dissimilarities and weights are positive or null, each new node added to a path increases the total dissimilarity of this path and never decreases it.

The tree building and its pruning are performed during the search and has to be fast enough to be compatible with a “real-time” use. The tree has at most n^m nodes if the target image is composed of n regions.

A classical heuristics for this “branch and bound” problem is to try the most promising solution at first.

The tree is generated by a depth-first search procedure. For each node, the son which has the best chance to lead to the path of minimal dissimilarity is examined at first. That is to say for each node of level i , the nodes of level $i+1$ are examined in the order of increasing dissimilarity. Thus the first path of length m is built by constructing at each node the branch leading to the pair of minimal dissimilarity. This path is not necessary the best path, but it gives a first solution for the graph matching. The depth-first procedure will then look for better solutions compared to this initial one.

By this way, the optimal solution is always found, without explicitly building all the possible paths. However this exhaustive search can be too slow (for too many query regions for example), the search can then be stopped at any time, (usually after a fixed amount of time) the found solution is then sub-optimal.

The properties of this tree are the following:

- there are as many levels in the tree as query regions ;
- as soon as the global dissimilarity of a path exceeds the value of the most promising one, the current node is not developed ;
- the optimal solution is always found ;
- a sub-optimal solution can be found in a given amount of time.

From the first property, it comes that one region of the target image can match several query regions, but not the contrary (cf. Figure 2). Thus the query images (those belonging to the learning set) are systematically over-segmented so that the object is divided into several regions, even if it consists in only one region in the test images.

3.4 Learning a set of graphs

In order to retrieve images containing a category of objects, these objects being very variable in shape and colour, we need several examples of the category. Thus we need a dissimilarity function not only between two regions (target and query) but also between one region (target) and a set of query regions.

Actually the system uses one classifier for each query region ; it is trained from varied examples. For instance, if the searched category is “car”, a classifier

is trained with regions of “lower-bodywork”, another classifier with regions of “upper-bodywork” (with “windows”) and a third one with regions of “wheel-and-shadow”. Any two-class classifier can be used, one class corresponds to one of the regions (for example, the class “lower-bodywork”), and the other class corresponds to the rest of the regions of all training images. The matching process works with the same search tree as in previous section. The only difference lies in the dissimilarity measure which no more involves a query region and a target region, but here involves a target region and a “semantic” class of regions (for example “lower-bodywork” or “wheel-and-shadow”). All examples of graphs for an object category must also share the same relative positions for the regions (adjacency, above, left to). For instance, all examples of “car” consist in a graph of three regions, one above the other, on the top “upper-bodywork”, then “lower-bodywork” and then “wheel-and-shadow”. The region “lower-bodywork” is adjacent to the two others.

Any two-class classifier can be used, but as for the global approach, the best results were obtained with a SVM classifier and a Gaussian kernel with χ^2 distance.

We tried various configurations for each class of object. The best results were obtained with two regions (cf. Figure 6) for all objects except “car” (3 regions), “road signs” and sunglasses (1 region). For “cow” category, the graph is made of 2 adjacent regions, with no spatial consistency (see one example 0).

For “US flag” and “Eiffel tower” categories, the graph is made of 2 adjacent regions, with a vertical consistency.

With this very original system using a graph matching, we are able to handle the great variability of the searched objects (see Figure 5 and 7). That explains why we won the contest for these two categories (US flag and cows), concerning the average precision.

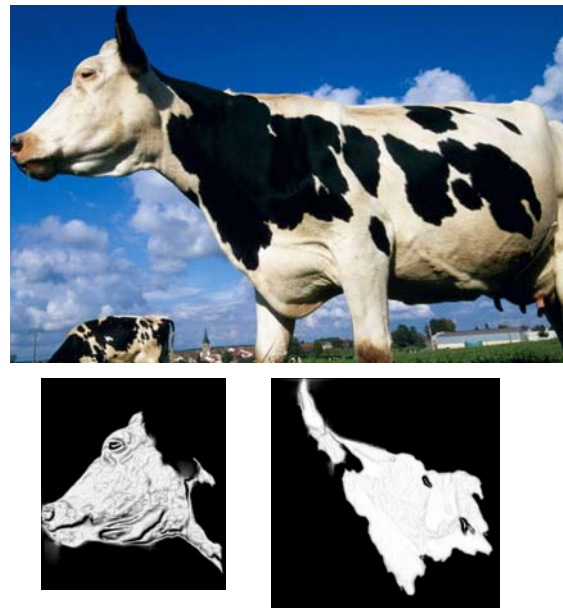


Figure 4. An example for the category “cow” :
2 fuzzy regions
head and part of the body

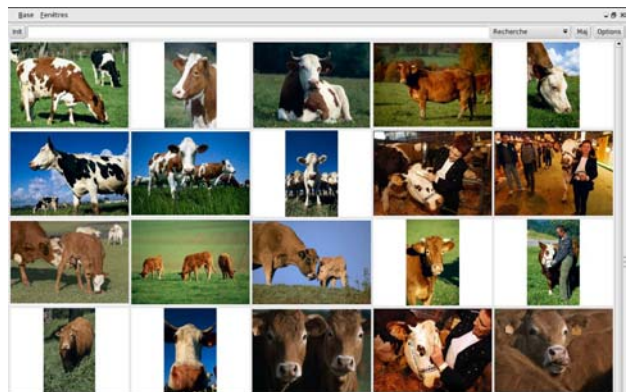


Figure 5. The 20 most relevant images retrieved for category “cow” (20 correct)

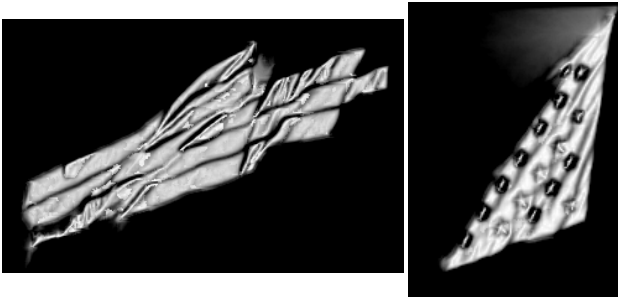


Figure 6. An example for the category “US flag” : 2 adjacent regions : stars (surrounded with red) and white and red strings (surrounded with green)

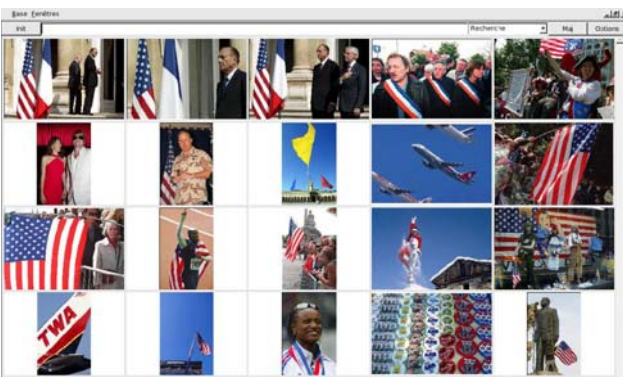


Figure 7. The 20 most relevant images retrieved for category “US flag” (12 correct)

4. ACKNOWLEDGMENTS

Our thanks to Julien Gony (engineer at ETIS lab.) for helping us to implement and tune the different learning strategies for the ImagEval contest.

5. REFERENCES

- [1] M. Cord, P-H. Gosselin, S Philipp-Foliguet, Stochastic exploration and active learning for image retrieval, *Image and Vision Computing*, 25, 14-23, 2007
- [2] M Cord, S Philipp-Foliguet, P-H. Gosselin, J Fournier, Interactive exploration to image retrieval , *Journal of Applied Signal Processing*, vol 2005, issue 14, Special Issue on Advances in Intelligent Vision Systems: Methods and Applications—Part II, 2173-2186, 2005
- [3] L. P. Cordella, P. Foggia, C. Sansone, M. Vento, Subgraph Transformation for the inexact Matching of Attributed Relational Graphs, *Computing*, 12, 43-52, 1998.
- [4] D. Dubois, M.C. Jaulent, A general approach to parameter evaluation in fuzzy digital pictures, *Pattern Recognition Letters* 6, 251-259, 1987
- [5] J. Fournier, M. Cord, S. Philipp-Foliguet, Retin: A content-based image indexing and retrieval system, *Pattern Analysis and Applications Journal*, Special issue on image indexation 4 (2/3) (2001) 153–173.
- [6] P.H. Gosselin, M. Cord, RETIN AL : an active learning strategy for image category retrieval, *IEEE Int. Conf. on Image Processing*, Singapore, 2004
- [7] Y. Linde, A. Buzo, R. Gray, An algorithm for vector quantizer design, *IEEE Transaction on Communication* 28 (1980) 84–94.
- [8] G. Patané, M. Russo, The enhanced LBG algorithm, *Neural Networks* 14 (9) (2001) 1219–1237.
- [9] S. Philipp-Foliguet, M. B. Vieira, A. de A. Araújo, Segmentation into fuzzy regions using topographic distance, 14th SIBGRAPI, 282-288, Florianopolis, Brazil, 2001
- [10] A. Rosenfeld, The fuzzy geometry of image subsets. *Pattern Recognition Letters* 2, 311-317, 1984.
- [11] E.H. Ruspini, A new approach to clustering. *Information and Control*, 15 (1), 22-32, 1969.
- [12] J. R. Smith, S-F. Chang, VisualSEEK: a fully automated content-based image query system, *ACM Multimedia Conf.*, Boston, USA, 87-98, 1996
- [13] N. Vasconcelos, Bayesian Models for Visual Information Retrieval, PhD thesis, Massachusetts Institute of Technology, 2000
- [14] Y. Rubner, Perceptual metrics for image database navigation, Ph.D. thesis, Stanford University (May 1999).
- [15] V. Vapnik, *Statistical Learning Theory*, John Wiley, New York, 1998.
- [16] L Vincent, P. Soille, Watersheds in digital spaces: an efficient algorithm based on immersion simulation, *IEEE Trans on PAMI*, 13 (6), 563-598, 1991.

