Image Content Classification by Using Visual Terms

Giuseppe Amato, Pasquale Savino, Vanessa Magionami
ISTI-CNR
{firstname.lastname}@isti.cnr.it

Abstract

We propose a technique for automatic recognition of content in images. Our technique uses machine learning methods to build classifiers able to decide about the presence of semantic concepts in images. Our classifiers exploits a representation of images in terms of vectors of visual terms. A visual term represents a set of visually similar regions that can be found in images. An image is indexed by first using a segmentation algorithm to extract regions, then extracted regions are replaced by the visual terms that represent them. We discuss how the set of visual terms is generated and how weights are assigned to visual terms to assess their relevance in images. A learning algorithm for Support Vector Machine is used to obtain a classifiers using training sets of images represented by using visual terms. The proposes technique offers very good performance as demonstrated by the experiments that we performed.

Categories and Subject Descriptors:

H.3 [Information Storage and Retrieval]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.7 Digital Libraries;

General Terms

Measurement, Performance, Experimentation

Keywords

Image content classification, machine learning

1. Introduction

Access to multimedia documents is typically obtained by searching machine understandable descriptions of the multimedia content of the document themselves. A description may consist of a free text annotation, or it may be represented by a more structured and articulated description, in accordance to some specific metadata schema as for instance Dublin Core [7], or MPEG-7 [15].

These descriptions are generally produced manually by cataloguers and describe content of multimedia data very effectively and precisely. However, there are some problems with manual generation of annotation. First, manual annotation is time consuming and expensive so it can be justified only when the searched material has very high worth. Re-annotating archived material, according to new strategies or new perspectives, is something that cannot be realistically performed at low costs. In addition, manual annotation is subjective: the quality of the annotation, and consequently the search performance, depends on the skills of the cataloguers.

For this reason, several systems offer image retrieval functionalities just relying on content-based (feature-based) image retrieval techniques [10,7] rather than exploiting semantic descriptions. Content-based image retrieval exploits similarity among low level features, such as color histograms, textures, or shapes [17], extracted from images and queries. The assumption is that visually similar images correspond to similar features, according to some mathematical definition of similarity among feature representations. With this search paradigm, users search by using other images as queries and retrieve images visually similar to the query image. Typically, this process is iterated after choosing one of the images returned by the system as a new query, until an image that satisfies the requirements is found.

However, inexperienced users are not satisfied by such systems given that their first intuition is

that of semantic similarity, rather than visual similarity. For instance, if color features are used to measure similarity, the use of a "red car" image as a query will bring back more red objects than cars. The *semantic gap* [8] refers to bad match between the low-level features, used by visual similarity search systems (content-based retrieval), and the high-level queries such as objects and concepts that users would like to express when searching in an image database.

This paper presents a novel techniques for automated recognition of image content based on the use of machine learning techniques and on the representation of images in terms of visual terms associated with image regions.

The paper is organized as follows. Section 2 gives an overview of the proposed technique. Section 3 describes the techniques for generating the visual lexicon, that is the set of visual terms used to represent visual content of images. Section 4 presents the technique used for associating visual terms with images. Section 5 deals with the issue of setting up classifiers for recognizing image content. Section 6 discusses the experiments that we performed to assess the performance of our approach. Section 7 concludes.

2. Overview of the approach

In this paper we propose a technique for automatic recognition of content in images. We represent the semantic content of images in terms of a finite set of concepts, denoted by labels. For instance we might want to recognize concepts like car, person, landscape, flower, city, people, countryside, sport, etc. Our techniques is based on the use of automatic classifiers to determine the occurrence of determinate concept in an image. We use Support Vector Machine (SVM) [6] to build classifiers, applied to a special representation of images.

SVM requires that objects (images in our case) to be classified have to be represented by points in a multi-dimensional Euclidian space. In order to build a classifier, learning algorithms for SVM require a training set composed of positive and negative examples of the class to be recognized. An SVM classifier is basically able to distinguish the portion of the space containing positive examples (occurrence of the concept) from that containing negative examples. In our context, positive points represent examples of images containing a certain concept, while negative examples represent images where the concept does not occur.

In order to effectively build SVM classifiers, we need a suitable representation for visual content of images, in terms of vectors, able to correctly represent the relationships between conceptual content and visual content. Humans decide about the occurrence of a concept in an image on the base of the co-occurrence of various combinations of visual regions. Accordingly, we represent the visual content of images in terms of visual regions. To make this representation finite and able to be encoded with vectors, we do not describe content of images with the regions that actually occur in an image. Rather, we represent the original regions with those taken from a controlled set of typical regions (prototypes of regions). We call the controlled set of typical regions, used to describe the content of images, the *visual lexicon* and each typical region in the visual lexicon a *visual term*. We use visual similarity between original regions and regions of the visual lexicon to determine which regions are used to represent an image. Note that the occurrence of a single visual term in an image does not imply any semantic meaning in an image. It just means that the image contains a region very similar to it. It is the combination of occurrence of several visual terms that indicates the presence of a certain semantic content in the image. Vector representation is obtained by associating images with vectors having the same size of the visual lexicon. Every element of the vector corresponds to one visual term. The value (weight) of an element of the array indicates the importance of the corresponding visual term in the image. Later, we will see that this weight is decided according to statistical properties of the data sets, concept to be recognized, and to the similarity between the actual regions contained in the image and the visual terms.

The intuition suggests that images containing the same semantic concept share a common group of regions and consequently the points that represent their visual content should be placed nearby

each other, or according to same unknown pattern (group of clusters for instance), in the underlying vector space. By using SVM technology and by using for each concept a training set composed of positive and negative image examples, it is possible to build for each concept a classifier that is able to determine if the points representing images to be classified fall in the positive or negative area of the space corresponding to the concept.

In order to build a framework able to support this process, there is a number of aspects that should be considered. We have to decide how the visual lexicon is built so that we are able to represent the needed visual aspects to recognize concepts. We have to define how images are indexed, that is how images are represented by means of visual terms. Finally we have to set-up a learning algorithm in order to build classifiers for concepts.

These aspects are discussed more in details in the next sections.

3. Visual lexicon generation

As discussed previously, in our approach images are indexed by using visual terms, consisting of typical visual regions that may occur in images. The occurrence of a particular combination of regions is used by a classifier to decide about the presence of a concept in an image. Clearly the variety of regions that can be found in all images is potentially infinite, and creating a visual vocabulary that contains all possible regions is not a good idea. The size of the visual lexicon can be prohibitive, and the probability that two images have the same regions is practically zero. However, as also discussed in [1, 12, 13], typically very similar regions play the same role, from a visual point of view, in the process of contributing to form a concept in the image. Intuitively, regions that play the same role should have the same representation. Assuming that visual similarity is an indication of equivalence of role of visual regions, we represent very similar visual regions by using the same prototype, which is used as a visual term in the visual lexicon.

So far, we talked about visual similarity without being very specific on it. However, visual similarity can be judged according to several aspects. In fact, in the literature, several visual features and correspondingly several similarity measures have been defined to characterize visual similarity. For instance, MPEG-7 [15] defines 5 visual descriptors, corresponding to global colours, scalable colours, hedge histograms, homogeneous textures, and region shape. Two regions that are judged to be very similar according to global colours might be very different according to their shape. In this respect, the visual lexicon should contain visual terms that are able to represent groups of regions according to various similarity criteria.

The visual lexicon that we use is a multi-feature visual lexicon, obtained as a composition of several mono-feature visual lexicons. Visual terms in a mono-feature visual lexicon are used to represent regions visually similar according to one specific feature. A multi-feature visual lexicon contains terms, belonging to different mono-feature lexicons, that are able to represent typology of regions according to different visual similarity criteria. There will be terms that represent regions that are similar according to their shape, others that represent terms that are similar according to their colour, etc.

In order to actually build the visual vocabulary we start with a training sets of images, which are representative of the dataset of images we want to deal with. We apply a region segmentation algorithm to this set of images and we obtain a set of regions. Provided that the image training set has been properly chosen and the region segmentation algorithm identifies good regions, we will end-up with a representative set of regions of the dataset. At this point we have to reduce the size of the region set in order to have the visual lexicon. In this work we have tested two different strategies.

The first very simple strategy consists in defining the size of the visual lexicon and to randomly chose the visual terms out of the region set obtained after segmentation.

For the second strategy we tried to apply the k-medoids [12] clustering algorithm to group together visually similar regions. Cluster representatives are the visual terms. An application of the clustering algorithm using a specific visual feature and similarity function generates a mono-feature

visual lexicon. Several applications of the clustering algorithm using different visual features generates the multi-feature visual lexicon.

In the experiments, we have used the ITI Segmentation algorithm [1] to segment the images. We used the five MPEG-7 [15] visual descriptors to represent the features extracted from regions. Therefore we have a visual lexicon composed of 5 different groups of terms.

4. Image indexing

The visual lexicon is used chose visual terms to be associated with images to describe their visual content. In principle, a visual term should be associated with an image when the image contains a region very similar to the visual term. The indexing process is composed of three different steps. Given an image to be indexed, this is first segmented by using a segmentation algorithm. The result of the segmentation returns a set of visual regions of the image. The second step chooses for each region a visual term to be used to represent the region. In order to do that each region is associated with the most similar visual term in each mono-feature lexicon. Therefore, if the visual lexicon is composed of *n* mono-features lexicons, each region extracted from the image will be represented by *n* visual terms. The last step associates a relevance value (a weight) to each selected visual term, trying to reflect the importance of the term in the image, in the dataset, and in the concept that has to be recognized. The technique for associating a weight to the selected visual terms is discussed more extensively in the following.

The first thing to do to decide the importance of a visual term is to determine the relevance of the term with respect to the image itself, without considering the entire dataset and the concept to be recognized. This measure is somehow very similar to what in text retrieval is called the Term Frequency (*TF*) of a term in a document. In text retrieval the term frequency TF^D_t of a term *t* in a document *D* is directly proportional to the number of occurrences of *t* in *D*. In our context, intuitively, the importance of a visual term *t* in an image *I* should be directly proportional to the similarity between a region *r* of the image and the visual term and to the size of the region in the image. In addition, it should be directly proportional to the number of regions, in the image, represented by *t*. This can be expressed as

$$TF_t^I = \sum_{r \in Regions(I,t)} sim(r,t) * cover(r,I)$$

where *Regions(I, t)* is the set of regions of *I* that are represented by *t*, *sim(r, t)* gives the similarity of region *r* to the visual term *t*, according to the low level feature of the lexicon of *t*, and *cover(r,I)* is the percentage of the area covered by *r* in *I*.

As we said before, *TF* just takes into considerations the content of a single image. However, the relevance of a term in an image should also take into considerations global aspects related to the dataset and the concepts to be recognized. In order to obtain the final weight of a term in an image we have compared two different techniques.

The first technique is the *TF*IDF* [16,17] technique widely used in text retrieval systems. It says that the weight of a term is directly proportional to the relevance of a term in the document (image in our case) and inversely proportional to the frequency of the term in the entire collection (Inverse Document Frequency, or simply *IDF*). In fact, if all documents have the same term, it means that that terms is not very indicative so the weight of that term should be penalized. We define the inverse document frequency of term *t* (IDF_t) as in traditional text retrieval systems. It is the logarithm of the ratio between the dataset size *N* and the number n_t of images, which contain *t*:

$$IDF_t = \log_e \frac{N}{n_t}$$

The weight of a term *t* in an image *I* is therefore obtained as

$$w_t^I = TF_t^I * IDF_t$$

The second technique that we have considered uses the information gain (*IG*) [3], which is

typically used for feature selection, to obtain the final weight of a term for an image. Given a concept c_i that we want to recognize, the *IG* says that the term *t* is relevant when it is able to discriminate when a document contains c_i. If term *t* is contained in all images that contains c_i and it does not occur in images that do not contain c_i, it is relevant. If term *t* is contained (or is not contained) in both images that contain c_i and do not contain c_i, it is not relevant. The *IG* of a term *t* for a concept c_i can be expressed as

$$IG_{t,c} = \sum_{\mathbf{c}\in\{c_i,\bar{c}_i\}} \sum_{\mathbf{t}\in\{t_k,\bar{t}_k\}} P(\mathbf{t},\mathbf{c}) \log_2 \frac{P(\mathbf{t},\mathbf{c})}{P(\mathbf{t})P(\mathbf{c})}$$

where c_i indicates that a random document *x* belong to the category c_i and $\bar{c}_i$ indicates that it does not belong to the category c_i, t_k indicates that the term t_k occurs in *x* and $\bar{t}_k$ indicates that it does not occur in *x*. For example, $P(\bar{t}_k, c_i)$ indicates the probability that, for a random document *x*, term t_k does not occur in *x* and *x* belongs to category c_i.

The weight of a term *t* in image *I*, when we want to recognize the concept c_i, is therefore obtained as

$$w_t^{I,c_i} = TF_t^I * IG_{t_k,c_i}$$

5. Classification

As we said previously, we represent the content of images by means of a finite set of concepts, denoted by labels as, for instance, car, person, landscape, flower, city, people, countryside, sport, etc. In order to recognize concepts in an image we build *binary classifiers*, that is, classifiers able to decide if a specific concept is present or not in an image. In this respect, we need to build a specialized classifier for each concept we want to recognize. A classifier takes as input an image and it says if the associated concept is present or not in the image. In order to obtain a complete description of an image, all available classifiers should be applied to the image to be classified.

5.1. Support Vector Machine

We build classifiers by applying the Support Vector Machine (SVM) [6] technology to the representation of images, in terms of vectors of weighted visual terms, described in previous sections. An SVM learns a decision rule for a concept by using a training set containing positive and negative examples. We denote the training as $T = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$ where $\mathbf{x_i}$ is the vector of weighted visual terms for image *i* of the training set. Each y_i is 1 or -1 according to the fact that the training image *i* is a positive or negative example of the concept to learn.

The learning phase of a classifier tries to determine what is the pattern of the portion of the vector space that includes vectors corresponding to images contains the concept, on the base of the training set. The simplest case is that of the linear SVM, where the learning phase determines an hyper-plane that divides the space in two subspaces. One sub-space corresponds to vectors of images that contain the concept, the other to vectors that do not contain the concept. In this case, omitting several theoretical details (see [6] for more information), the learning phase has to find a vector $\boldsymbol{\omega}$ such that the decision function

$$f(\mathbf{x}) = <\boldsymbol{\omega}, \mathbf{x}> + b$$

is able to optimally classify most of the training set examples.

In real cases it is not possible to linearly separate negative from positive examples, so a linear SVM is not effective. However, in most cases a linear separation is still possible by mapping the vectors in an higher dimensional space. Suppose you have a mapping function Φ that maps vectors in a space of large (even infinite) dimensions where a linear separation can be found between positive and negative examples. In this case the decision function can be written as

$$f(\mathbf{x}) = <\boldsymbol{\omega}, \Phi(\mathbf{x})> + b\,.$$

Actually, SVM methods do not define the mapping function Φ explicitly, but use the properties

of the kernel functions. A kernel function K, defined as $K(\mathbf{x}_i,\mathbf{x}_j)= <\Phi(\mathbf{x}_i)^T,\Phi(\mathbf{x}_j)>$, computes the dot-product of vectors obtained from the mapping of $\mathbf{x}_i$ and $\mathbf{x}_j$. There are simple kernel function that easily compute the dot-product of vectors mapped in very high or even infinite dimensional spaces without even knowing the actual mapping functions.

It can be proven [3] that the kernel based decision function, defined above, can be also represented in the dual form as

$$f(\mathbf{x}) = \sum_{i=1}^{n} y_i \alpha_i \, \mathrm{K}(\mathbf{x}, \mathbf{x}_i) + b$$

in terms of the training vectors. In this formulation, the learning phase consists in finding the parameters α_i (which basically determine the contribution of each example of the training set to the solution of the learning problem) rather than the vector $\boldsymbol{\omega}$. To execute the learning phase we have used the kernel adatron algorithm. The adatron was first introduced in [1] as a perceptron-like procedure to classify data and then a kernel-based was proposed in [6]. The learning strategy of the adatron is to perform a gradient ascent to solve the margin-maximization problem between the positive and negative examples of the training set.

In our implementation we used the SVM library in [4].

5.2. Tuning considerations

The performance of the SVM classification is strongly related to the choice of the kernel function, the kernel parameters, and some parameters related to the adatron algorithm, such as the penalty parameter *C*, the maximal tolerance on the margin, and maximum number of iterations.

There is a large number of kernel functions available. We have chosen the Gaussian Radial Basis Function (RBF) given its capability to recognize separate areas of the vector space where positively classified elements can be found. In fact, the occurrence of a concept in an image cannot, in general, be uniquely determined by the presence (or absence) of some visual terms. Rather, various different and dissimilar patterns can suggest the presence of the same concept. The RBF takes as parameter the variance σ of the underlying Gaussian.

In general, it is not not possible to know in advance the parameters that offer the best performance for a specific problem. A widely used procedure, to chose optimal parameter, is the *v* cross-validation. This procedure divides the training set into *v* different subsets (folders) of equal size: one folder is used as test set for the classifier and the other *v*-1 folders are used as training sets. The cross-validation process is then repeated *v* times using every time a new folder of the *v* subsets as test set. Finally the *v* results obtained can be averaged to produce an overall evaluation of the classification system.

In order to automatically find the optimum values for the penalty parameter *C* and the variance σ of the RBF, one parameter is kept fixed and the other parameter grows exponentially and vice versa. After identifying the best pairs, using cross validation, we performed a finer search on their neighbourhood. Once we have determined the best choice for (C,σ), we have also tested some possible values to find the best maximal tolerance on the margin and maximum number of iterations.

6. Experiments

We validated the technique that we propose by running some performance experiments that measure the quality of image content recognition, according to the various options that is possible to choose when setting up our classifier.

We first run experiments to measure the quality of the classifier when using either a visual lexicon obtained by clustering or by choosing regions at random (see Section 3 for more details). In this first test we also compare the performance by varying the size of the visual lexicon. Then, we evaluated the weighting strategy. We ran experiments by using images indexed by using the tf*idf

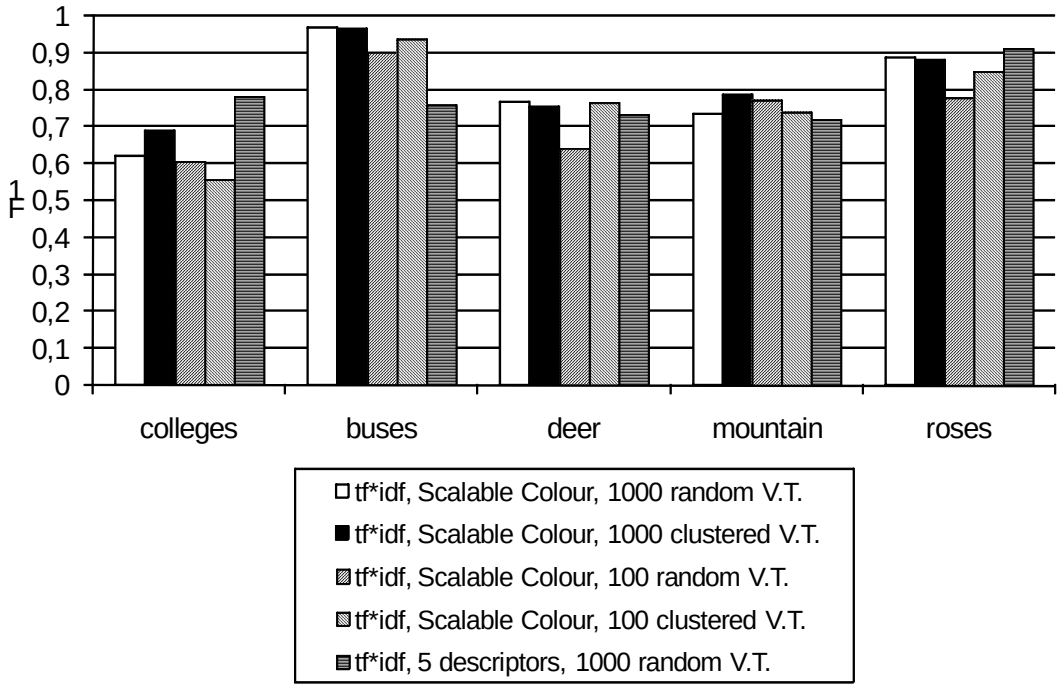

Figure 1: Classification performance using a visual lexicon obtained by randomly choosing the visual terms and by determining visual terms using the k-medoid clustering algorithm. Size of the visual lexicon is 10 and 1000. We used a mono-feature visual lexicon obtained by using the MPEG-7 Scalable Color only. We also compare a multi-feature visual lexicon using all 5 MPEG-7 descriptors. All tests were obtained using tf*idf to assign weight to elements of vectors associated with images.

technique and the tf*ig technique to decide the weights associated with each term of the visual lexicon when building the vectors associated with each image (see Section 4 for more details). Finally, we tested the performance of our classification technique by using a multi-feature visual lexicon, using 5 MPEG-7 visual descriptors to represent visual features of region (see again Section 3 for more details) . To perform our tests we used a subset of the COREL collection, containing images belonging to 5 different categories (buses, roses, colleges, mountains, and deers).

6.1. Settings for the experiments

We used the ITI segmentation algorithm [7] to obtain the regions used for generating the visual lexicon and for indexing images of the experiments. The ITI algorithm was set to extract about 10 regions from each image. From each region we extracted the five MPEG-7 visual descriptors (Scalable Color, Edge Histogram, Dominant Color, Region Shape, Homogenous Texture), by using the MPEG-7 reference software. We used a subset of 1000 randomly chosen images as training set to generate the visual lexicon. Various mono-feature visual lexicons were generated. We used different MPEG-7 visual descriptors, we used the clustering and the random choice techniques identify visual terms, and finally we set se size of the lexicon to 100 and 1000 visual terms. The proposed approach was tested by using each mono-feature visual lexicon and by combining 5 mono-feature visual lexicons, corresponding to the 5 MPEG-7 visual descriptors, in a single multi-feature lexicon.

To perform the experiments, as anticipated above, we used a subset of the COREL collection, containing images belonging to 5 different categories (buses, roses, colleges, mountains, and deers). Each category is composed of 100 examples. Images within each category are randomly divided in 10 folder each with 10 elements to perform *v* cross-validation (see also Section 5.2 for additional details). To determine the optimum values for *C* and σ we have considered first 2^{-5}, 2^{-}

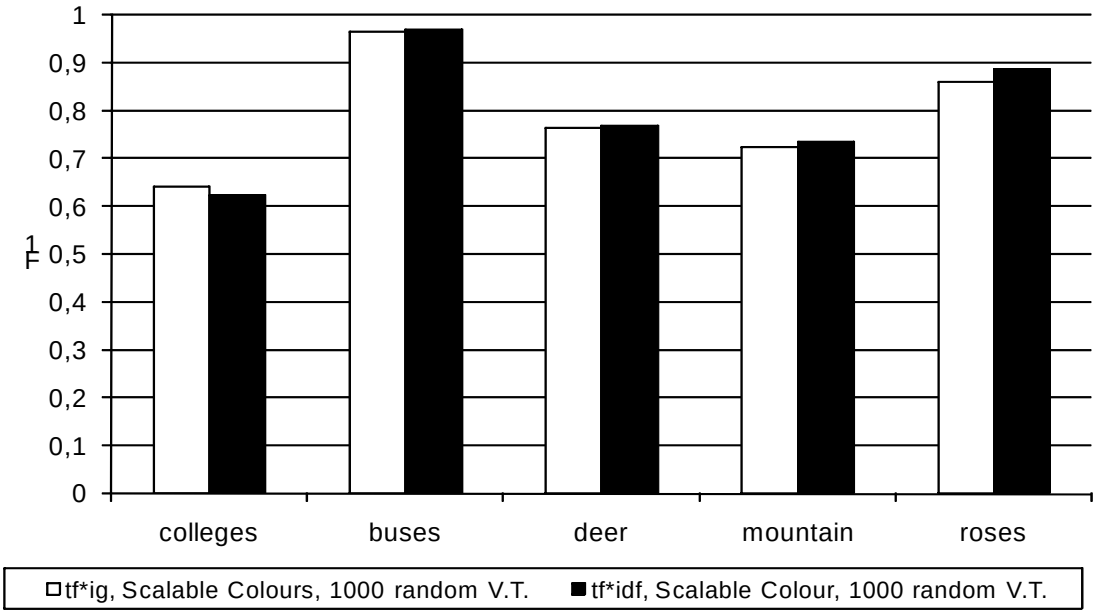

Figure 2: Classification performance using the tf*ig and tf*idf to assign weight to vectors of visual terms associated with images. The visual lexicon was obtained using random selection of visual terms. We just compared a mono-feature visual lexicon obtained using the MPEG-7 Scalable Colors descriptor

^{3},…,2^{11} as values of C and 2^{-15}, 2^{-13},...,2^{3} as values for σ (each value of C is combined with each value of σ). The best values identified in the first run of cross-validation were then further refined using fine-grained intervals. We have tested one category at a time. Positive example are taken from the 100 images composing the category itself, while negative examples are randomly chosen from the other categories. At each run of the cross-validation, one folder is considered as test set and the remaining folders are used as training set. The number of folders used as training set varies from 20 examples (10 positive and 10 negative) to 180 examples (90 positive and 90 negative).

6.2. Results

Objective performance measurement was obtained by using the well known F1 measure defined as

$$\mathrm{F1} = \frac{2a}{2a + b + c},$$

where a are the positive examples correctly classified, b are the positive examples incorrectly classified, c are the negative examples incorrectly classified. We computed the F1 measures for each concept according to each tested setting.

The first test that we performed was intended to compare the performance obtained using various strategies for generating the visual lexicon. We compared visual lexicons of size 100 and 1000 elements, obtained using both the random and clustering visual term generation. We compared individually the 5 mono-feature visual lexicons, corresponding to the 5 MPEG-7 visual descriptors, using tf*idf to associate weight with images and visual terms. The best performance was obtained by using the Scalable Color visual descriptor with all tested settings and results are reported in Figure 1. For brevity, we do not report results obtained with all visual descriptors. F1 values were in general very high, reaching 0.95 in some cases (buses concept). From the histograms we can infer that there is not an evident difference in performance between the random and the clustering strategies and between the visual lexicons of size 100 and 1000. Consider that the cost of using the clustering strategy is $O(k*N*iter)$, where k is the number of visual terms, N is the number of initial regions of the training set, and *iter* is the number of iterations of the k-medoid

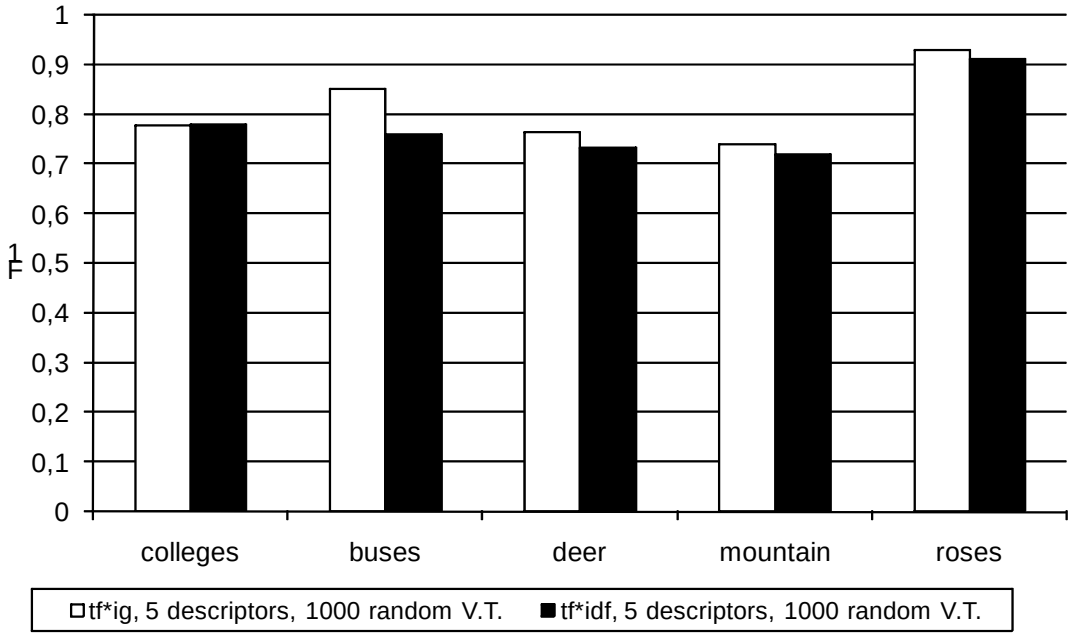

Figure 3: Classification performance using the tf*ig and tf*idf to assign weight to vectors of visual terms associated with images. The visual lexicon was obtained using random selection of visual terms. We used a multi-feature visual lexicon containing all 5 MPEG-7 descriptors.

algorithm. On the other hand the cost of the random strategy is simply $O(k)$, given that we have just to chose k random regions. The cost of the clustering strategy is much higher than that of the random strategy. However, clustering does not offer a performance that justifies its use. We can also observe that there is not a significant difference in performance when the size of the visual lexicon changes from 100 to1000. This suggests that the granularity of the information that we need to classify images with this technique does not need to be very fine grained. In Figure 1 we also report the results obtained by using a multi-feature visual lexicon. It can be seen that the results are comparable (sometimes a bit better sometime a bit worse) to those obtained when we used the best mono-feature visual lexicon, which is the one obtained using Scalable Color. Note that in general, it is not possible to know what is the best mono-feature visual lexicon in advance. The performance might depend, on the test sets, the training sets, the concepts. The advantage of the use of the multi-feature lexicon is that we do not have to decide in advance what is the best visual descriptor. In fact, thanks to the weighting strategy, the method offers good performance (comparable to the best visual descriptor) in all circumstances.

In the second test we compared the performance obtained using the tf*idf and tf*ig weighting strategies. Also in this case we used a mono-feature lexicon of size 1000 obtained using the Scalable Color visual descriptor with the random strategy. Surprisingly, our tests showed that there is not a significant difference in performance between the two methods. Results are shown in Figure 2.

For completeness we run the same test using the multi-feature visual lexicon. Results are reported in Figure 3. In this case the difference in performance of the two weighting strategies becomes more evident. The tf*ig method returns the best results on average. We believe that this effect is due to the capability of the tf*ig method to better select the relevant terms for a concept, which is more important in a multi-feature lexicon, where selecting a term has also the effect of deciding the importance of a visual descriptor with respect to the other. The effect was less evident in the previous experiment given that we tested the mono-feature visual lexicon obtained using the Scalable Color visual descriptor, which, as we said before, is the best descriptor for the dataset and concepts that we used.

7. Conclusions

We presented a techniques for automatic recognition of image content. In our approach an image's semantic content is represented by the set of concepts that are present in the image. A classifier has to be built for each concept that we want to recognize. We use an image visual representation based on visual terms representing regions of images. Machine learning techniques are used to train classifiers from training sets of images represented by vectors of visual terms. We have tested different techniques for creating the set of visual terms and for creating the vectors of visual terms, which represent the images. We have performed various tests to measure the quality of the technique and to assess the various options that we presented.

Experiments provided evidence that generating the visual terms by simply choosing a number of random regions, from the set of regions extracted from a training set of images, offers a performance comparable to that of selecting visual terms by using a clustering techniques. Clearly the cost of randomly selecting the images is negligible, compared to that of the clustering algorithm. In addition, we have evidence that the weighting techniques based on the information gain, in conjunction to the use of several different types of visual descriptors, to describe the visual appearance of regions, beats weighting based on tf*idf.

8. References

1. G Amato, V. Magionami, P. Savino, Image Indexing and Retrieval Using Visual Terms and Text-Like Weighting, DELOS Conference on Digital Libraries, Tirrenia (PI), Italy, 13-14 February 2006
2. J. Anlauf and M. Biehl. The adatron-an adaptive perceptron algorithm. *Europhysics Letters*, vol.10:pp.687–692, 1989.
3. B. E. Boser, I. Guyon, and V. Vapnik, A training algorithm for optimal margin classifiers, in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, Pittsburg, USA, 1992, pp. 144-152
4. G. Caron, SVM in Java, http://www.site.uottawa.ca/~gcaron/svm.htm
5. T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., New York, NY, USA, 1991.
6. N. Cristianini and J.Shawe-Taylor, *An Introduction to Support Vector Machines, and other kernel-based learning methods*, Cambridge University Press, Cambridge, UK, 2000
7. A. Del Bimbo, *Visual information retrieval*. San Francisco, CA: Morgan Kaufmann, 1999.
8. C. Dorai and S. Venkatesh, Bridging the semantic gap in content management systems: Computational media aesthetics in COSIGN *2001: Computational Semiotics CWI, Amsterdam (The Netherlands)*, September 2001.
9. Dublin Core Metadata Initiative, http://dublincore.org
10. M. Flickner, H. S. Sawhney, 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, 1995.
11. T.-T. Frieß, N. Cristianini, and C. Campbell. The Kernel-Adatron algorithm: a fast and simple learning procedure for Support Vector machines. In *Proc. 15th International Conf. on Machine Learning*, pages 188–196. Morgan Kaufmann, San Francisco, CA, 1998.
12. L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, 1990.
13. B. Le Saux, G. Amato, Image Classifier for scene analysis, *ICCVG 04, International Conference on Computer Vision and Graphics*, Warsaw, Poland September 22-24, 2004
14. V. Mezaris, I. Kompatsiaris, and M. G. Strintzis. Still image segmentation tools for object-based multimedia applications. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(4):701–725, June 2004.
15. P. Salembier, T. Sikora, and B. Manjunath. *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, Inc., New York, NY, USA, 2002.
16. G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, 1983.
17. J. R. Smith, Integrated spatial and feature image systems: Retrieval, analysis, and compression *Ph.D.*

dissertation, Graduate School of Arts and Sciences, Columbia University, 1997.
18. I. H. Witten, A. Moffat, and T. C. Bell. Bell: Managing Gigabytes: Compressing and Indexing *Documents and Images*. Morgan Kaufmann, 1999.