Combining visual dictionary, kernel-based similarity and learning strategy for image category retrieval

Philippe-Henri Gosselin, Matthieu Cord, Sylvie Philipp-Foliguet

Combining visual dictionary, kernel-based similarity and learning strategy for image category retrieval

Philippe Henri Gosselin[1] , Matthieu Cord[2] and Sylvie Philipp-Foliguet[1]

(1) gosselin@ensea.fr, ETIS / CNRS, 95014 Cergy-Pontoise, France
(2) matthieu.cord@lip6.fr, LIP6 / CNRS, 75016 Paris, France

Abstract

This paper presents a search engine architecture, RETIN, aiming at retrieving complex categories in large image databases. For indexing, a scheme based on a two-step quantization process is presented to compute visual codebooks. The similarity between images is represented in a kernel framework. Such a similarity is combined with online learning strategies motivated by recent Machine-Learning developments such as Active Learning. Additionally, an offline supervised learning is embedded in the kernel framework, offering a real opportunity to learn semantic categories. Experiments with real scenario carried out from the Corel Photo database demonstrate the efficiency and the relevance of the RETIN strategy and its outstanding performances in comparison to up-to-date strategies.

Key words: Multimedia Retrieval, Machine Learning, Kernel Functions, Quantization
PACS:

1 Introduction

Large collections of digital images are being created in different fields and many applicative contexts. Some of these collections are the product of digitizing existing collections of analogue photographs, paintings, etc, and others result from digital acquisitions. Potential applications include web searching, cultural heritage, geographic information systems, biomedicine, surveillance systems, etc.

The traditional way of searching these collections is by keyword indexing, or simply by browsing. Digital image databases however, open the way to content-based searching. Content-Based Image Retrieval (CBIR) has attracted a lot of research interest in recent years. A common scheme to search the database, is to automatically extract different types of features (usually color, texture, etc.) structured into descriptors (indexes). Theses indexes are then used in a search engine strategy to compare, classify, rank, *etc*, the images.

Major sources of difficulties in CBIR are the variable imaging conditions, the complex and hard-to-describe image content, and the gap between arrays of numbers representing images and conceptual information perceived by humans. In CBIR field, the semantic gap usually refers to this separation between the low-level information extracted from images and the semantics [1,2]: the user is looking for one image or an image set representing a concept, for instance a type of landscape, whereas current processing strategies deal with color or texture features !

Learning is definitively considered as the most interesting issue to reduce the semantic gap. Different learning strategies, such as offline supervised learning, online active learning, semi-supervised, etc., may be considered to improve the efficiency of retrieval systems. Some offline learning methods focus on the feature extraction or on the similarity function improvement. Using experiments, a similarity function may be trained in order to better represent the distance between semantic categories [3]. Thanks to local primitives and descriptors, such as salient points or regions, supervised learning may be introduced to learn object or region categories [4,5]. The classification function is next used to retrieve images from the learned category in large databases. Other strategies focus on the online learning to reduce the semantic gap [6,7]. Interactive systems ask the user to conduct search within the database. The information provided by the user is exploited by the system in a relevance feedback loop to improve the system effectiveness. Online retrieval techniques are mainly of two types: geometrical and statistical. The geometrical methods refer to search-by-similarity or query-by-example (QBE) systems, based on calculation of a similarity between a query and the images of the database [8,9]. Recently, machine learning approaches have been introduced in computer vision and CBIR context and have been very successful [10,11]. Discrimination methods (from statistical learning) may significantly improve the effectiveness of visual information retrieval tasks [12].

In this paper, we introduce our general strategy RETIN to manage indexing and category retrieval by content in large image databases. Some modules concern the indexing step and other ones learning strategies based on offline or online supervising. A first version of our system has been already published [13]. We propose here a new generation of RETIN. In the manner of Fayyad description of the challenges of data mining and knowledge discovery [14], our whole context of visual data mining is summarized on Fig. 1. Starting from raw data, the first challenge is to extract visual descriptors and to structure them into indexes, *i.e.* visual signatures. The indexing step is composed by

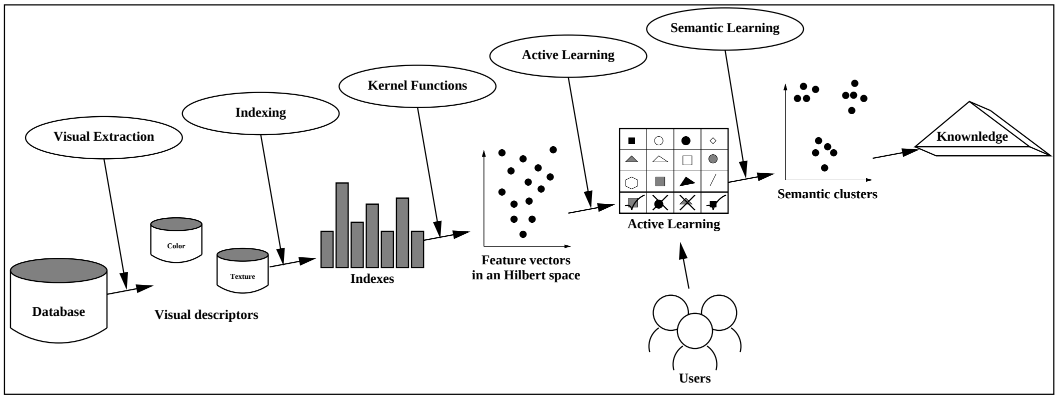


Fig. 1. An overview of the steps that compose the RETIN process. Working on the raw data, low-level processes consist in extracting visual features and signatures. Consolidated level focus on similarity and online learning of image categories using user interaction. High level semantic analysis deeply exploits users' feedbacks to reduce the semantic gap.

a new scheme to get visual signatures from images. Let's say that this is the low level of analysis. The comparison between the indexes is carried out using kernel framework. Searching with user interaction allows to extract subsets of relevant images from the database. A machine-learning-oriented scheme is proposed to embed all the modules of the search in a coherent and efficient framework. This is the intermediate level of abstraction and data mining. To go further towards the knowledge extraction and database structuring, a semantic learning scheme is also proposed (Fig. 1). All the former user interactions are used to progressively learn data clusters in the database. This is our high level or semantic level of data analysis.

We emphasize in this article the global efficiency and consistency of our search engine architecture to deal with complex category retrieval in large databases. Some specific contributions are also proposed in each part. For indexing, the computing of visual codebooks is a real challenge, we propose an original two-step vectorization scheme in section 2. The similarity between images is the core of the search, we propose a kernel framework to manage this aspect in section 3. It allows us to propose a powerful online learning strategy motivated by recent machine-learning developments such as active or transductive learning, presented in section 4. Offline supervised learning is also embedded in our kernel framework, our innovative long-term learning strategy is presented in section 5.

2 Visual codebook based quantization

Building a visual codebook is an effective way 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 [15] where 166 regular colors 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 clustering algorithm, such as k-means. In this case, the visual codebook is adapted to the image database. When using color features, this strategy extracts the dominant colors in the database[16]. Using a k-means algorithm leads to a sub-optimal codebook, where codewords are under- or over-representing the visual content. An usual way to find a good visual codebook is to train several times the clustering algorithm and to merge the codebooks or to keep the best one. However, because of the large number of vectors to be clustered, this strategy has a very high cost in computational time.

In this section, we first study new alternatives to the standard k-means algorithm, and select the most efficient 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 algorithm can not be directly applied, since the whole vector set can not be stored in memory. In this last sub-section, we propose a clustering algorithm which leads to a near to optimal codebook with only one training pass.

2.1 Low-level feature extraction

In order to build the visual codebook, we first need a large set $V = \{\mathbf{v}_1, ..., \mathbf{v}_m\}$ of feature vectors extracted from the images of the database. In this paper, we use two visual features:

- Color from $CIEL^\star a^\star b^\star$ space. Each pixel of coordinates (x, y) is converted to a $L^\star a^\star b^\star$ vector of dimension 3, i.e. $\text{pixel}(x, y) \mapsto (L^\star(x, y)\ a^\star(x, y)\ b^\star(x, y))^\top$.
- Texture from complex Gabor filters. We process each image of the database with 12 complex Gabor filters, in 3 scales and 4 orientations. The output of these 12 filters provide 12 images $F_1, \ldots, F_{12}$. For each pixel of coordinates (x, y), we consider the vector of 12 dimensions whose values correspond to the 12 filter outputs at the same coordinates (x, y). That is to say $\text{pixel}(x, y) \mapsto (F_1(x, y)\ \ldots\ F_{12}(x, y))^\top$.

2.2 Dynamic quantization

Vector quantization aims at finding the optimal set $W^\star = \{\mathbf{w}_1, ..., \mathbf{w}_\kappa\}$ of codewords able to represent a set $V = \{\mathbf{v}_1, ..., \mathbf{v}_m\}$ of vectors. This issue is solved by splitting the set V into clusters. Each vector will then be represented by the closest vector of W, $q_W(\mathbf{v}) = \underset{\mathbf{w}\in W}{\operatorname{argmin}}\ d(\mathbf{v},\mathbf{w})$, for a given distance d (usually the Euclidean distance). The problem can be addressed as an optimization problem which aims at minimizing the *distortion* of each cluster:

$$W^\star = \underset{W}{\operatorname{argmin}}\ D_W(V) \tag{1}$$

where the distortion of a set V for a codebook W is defined by :

$$D_W(V) = \sum_{\mathbf{v}\in V} d(\mathbf{v}, q_W(\mathbf{v}))^2 \tag{2}$$

The distortion measures the average squared distance between a vector $\mathbf{v}$ and its corresponding codeword $q_W(\mathbf{v})$. Minimizing this criterion aims at getting compact and equidistribued clusters.

The optimization 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 initialization have been proposed, like the k-means splitting or LBG [17]. The algorithm starts with only one codeword, and step after step, splits the clusters into two sub-clusters. Patanè proposes ELBG, an enhanced LBG algorithm, that introduces a heuristic in order to jump from a local minimum to a better one [18]. 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 (see appendix). Fig. 2 shows the results in terms of average PSNR (log value of the distortion), and the average computation time for the quantization in 256 colors 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.

2.3 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.

Method	PSNR(dB)	Time(sec)
k-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

Fig. 2. Performances and computational time of the quantization methods.

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 [19] 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 quantizes each image;
- The second stage quantizes the whole database from the dictionaries obtained at the first stage.

The advantage is that each image is independently processed 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). The set of feature vectors V_i of image i are computed and quantized using ELBG in κ codewords. The codebook for image i is denoted $W_i = \left\{\mathbf{w}_i^j, j = 1, ..., \kappa\right\}$. In the second stage, the set $\{W_i, i = 1, ..., n\}$ is clustered into the expected number of codewords with ELBG classifier. To take into account the fact that images can be of various sizes, the distortion of any class C, represented by w_C is modified in Eq (2) by adjunction of a weighting coefficient equal to the cardinality of the class. So after the first stage, we have for each image i the set of codewords $\left\{\mathbf{w}_i^j, j = 1, ..., \kappa\right\}$ and the set of corresponding weights $\left\{z_i^j, j = 1, ..., \kappa\right\}$, where z_i^j is the cardinal of class j in image i.

So the formula for distortion of $\widetilde{X} = \{(\mathbf{w}_i^j, z_i^j)\}$ becomes :

$$\widetilde{D}(\widetilde{X}) = \sum_j \sum_i z_i^j \times d(\mathbf{w}_i^j, q(\mathbf{w}_i^j))^2 \tag{3}$$

and the computation of codewords becomes :

$$\mathbf{w}_{C_j} = \frac{\sum\limits_{i \in C_j} z_i^j \times \mathbf{w}_i^j}{\sum\limits_{i \in C_j} \mathbf{z}_i^j}$$

2.4 Image signature computation

Once a codebook W has been generated, unique for the whole database, the histogram H_i of image i is computed for each visual feature. We replace each feature vector $\mathbf{v}(x, y)$ corresponding to pixel (x, y) with the the closest codeword $q_W(\mathbf{v}(x, y))$ in the codebook. Next, we count the number of times each codeword is present in the image to build the histogram. The histogram is finally normalized to get a distribution vector $\mathbf{d}_i = H_i/||H_i||_{L_1}$. The image signature $\mathbf{x}_i$ is then the concatenation $(\mathbf{d}_i^{feature1} \; \mathbf{d}_i^{feature2} \; \dots)^\top$ of distributions for all visual features (in this paper, color and textures).

The final step is the tuning of the size of the visual codebooks, that we study in the next section.

2.5 Experiments

The adaptive classification of Fournier[19] and our two-stage method are compared in Fig. 3 and Fig. 4 on the Corel Photo database (see appendix for details).

Although we have used the distortion and the time cost criteria to select the quantization method in the first stage of our algorithm, we use here the Mean Average Precision (see appendix for definition) in order to evaluate the performances of a codebook in the CBIR context. Indeed, this statistic is used a lot in information retrieval framework.

For Fournier's method, the complete quantization must be done again for each codebook size. For our method, ELBG is first computed to get a quantization of each image into 256 image-dependent codewords. The codewords and their weights are then clustered by the second stage with ELBG.

Concerning color codebooks, both methods are close, with a small advantage for our method. Both methods have a maximal MAP for 50 codewords. Concerning texture quantization, the proposed method clearly outperforms Fournier's one. The global maximum is also obtained for 50 codewords. Another interest is the time saving with our method, which is much faster than

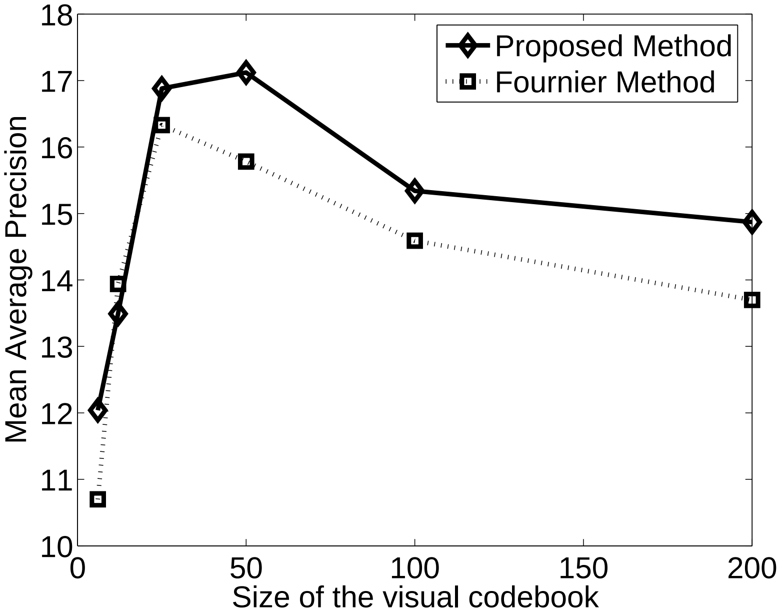


Fig. 3. Color quantization into 6, 12, 25, 50, 100, and 200 codewords ($L^\star a^\star b^\star$) with Fournier's adaptive quantization method, and our two-stage method.

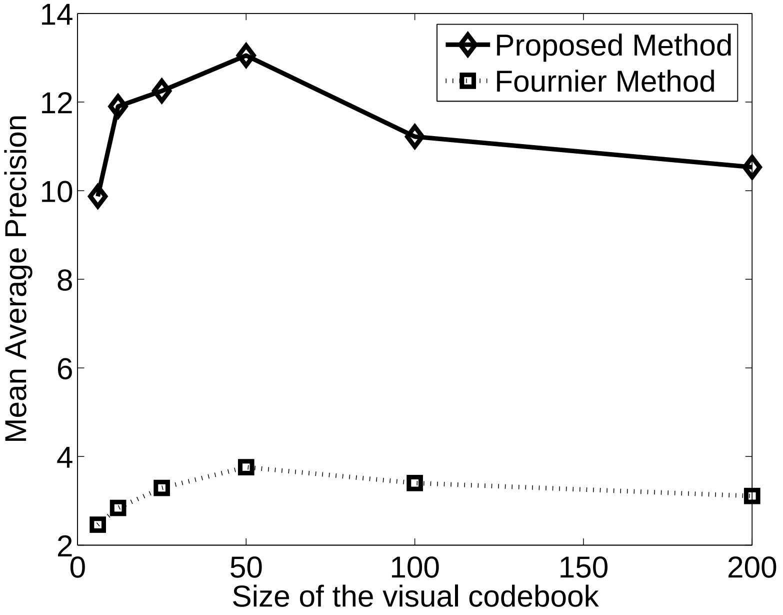


Fig. 4. Texture quantization into 6, 12, 25, 50, 100, and 200 codewords (Gabor filters), with the adaptive quantization method of Fournier, and the two-stage method we propose.

Fournier's one, since the first stage can be achieved in parallel on several machines. As the quantization in 25 codewords almost reaches the same performances as the quantization in 50 codewords, for signatures twice smaller and a time saving, we have opted for a quantization into 25 codewords for color and 25 codewords for texture.

These experiments also show the interest of our method for tuning the size of the visual codebook. Assuming that we have *a priori* knowledge about the categories likely to be searched, several visual codebooks of various sizes can be easily computed and evaluated since only the second step of the algorithm is necessary.

Furthermore, the two stages allow a fast adding/removal of images in the database. When adding new images, only the computation of their visual descriptors and the second stage of the method are required to compute the new codebook.

3 Similarity using kernel functions

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

Basically, the Euclidean 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 [20], entropy [21,22], or χ^2 distance [23]. 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 in [24], Earth Mover's Distance [25], or generalized quadratic distances ($d(\mathbf{x}_i, \mathbf{x}_i) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)}$)

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 [26], we have chosen to use kernel functions.

3.1 Kernel framework

The approach consists in finding a mapping Φ from input space X (here our histogram space) to a Hilbert space $\mathcal{H}$. Thus, once found this mapping, all the addressed learning problems become linear. Furthermore, we do not directly work on the mapped vectors, but on their dot products $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$.

In our case, since we are working on histograms, an interesting kernel function is the Gaussian one $k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{d(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2}}$. This function depends on a distance $d(\mathbf{x}_i, \mathbf{x}_j)$, which allows us to pickup one of the distances for histograms. For

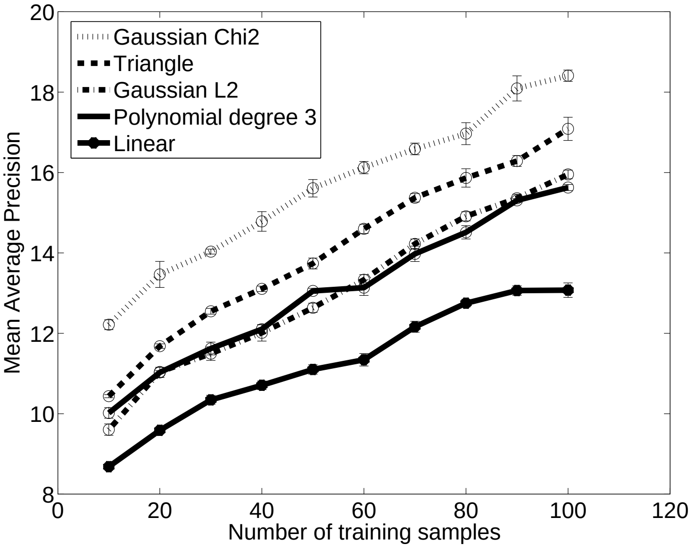


Fig. 5. Mean Average Precision(%) for a classification by SVM according to the number of training data, for several kernel functions on the Corel Photo database.

instance, we can use the χ^2 distance $d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{r=1}^{p} \frac{(x_{ri}-x_{rj})^2}{x_{ri}+x_{rj}}$. Note that we could use more robust distances, such as the the Earth Mover's Distance [25], but this leads to a too high computational cost for the processing of huge databases.

In order to evaluate the interest of this kernel against the standard ones, we have compared their performances for a SVM classifier (see appendix for details). Results are shown on Fig. 5. The linear kernel, which can be seen as the "no kernel" strategy, gives the worst performances. It is followed by the polynomial kernel (of degree 3), which was originally tuned for the tracking of high-level correlations of data. Close to this one is the Gaussian kernel, with an Euclidean distance, and next is the triangle kernel, which is invariant to scale variation. Finally, the Gaussian kernel with a χ^2 distance gives the best performances, results which are consistent with the use of histograms as index. Thus, in the following experiments, we will use a Gaussian kernel with a χ^2 distance.

Note that, although the Gaussian distance χ^2 is the most interesting for our indexes, it will be no longer true on non-histograms ones. However, assuming that one can find a kernel function relevant for one's indexes, all the results about the learning techniques we present in the next sections are still valid, since they are made to work in a Hilbert space induced by a kernel function.

In any cases, we assume that we are working in a Hilbert space. Then, several standard operators may be expressed using k, as for instance the Euclidean distance $d(\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j))^2 = k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j)$ [27]. Similarity s may also be defined as the dot product in the induced space $s(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j)$. But other measures, as for instance the angle between two vectors, may be used, for instance $s(\mathbf{x}_i, \mathbf{x}_j) = \frac{|k(\mathbf{x}_i,\mathbf{x}_j)|}{\sqrt{k(\mathbf{x}_i,\mathbf{x}_i)k(\mathbf{x}_j,\mathbf{x}_j)}}$.

We use kernel function k as the similarity function, and kernel matrix $\mathbf{K}$ defined by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ as the similarity matrix. As k is a kernel function, matrix $\mathbf{K}$ is symmetric and semi-definite positive (*sdp*), that is to say a Gram matrix. This matrix embeds the index information $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and the similarity function k related to the whole database. All the data mining processes, classification ranking, and so on, are only based on this Gram matrix data. The advantage of this framework is then to well separate the learning problem from the similarity definition.

We propose in the next section online learning algorithms before introducing an extended kernel framework merging the low-level similarity matrix $\mathbf{K}$ with high-level information obtained from user interaction.

4 Active classification for interactive retrieval

Indexes and similarity function allow to compare any pair of images. In CBIR, the retrieval may be initialized using a query as an example. The top rank similar images are then presented to the user. Next, the interactive process allows the user to refine his request as much as necessary. Many kinds of interaction between the user and the system have been proposed [28], but most of the time, user information consists of binary annotations (labels) indicating whether or not the image belongs to the desired category. The positive labels indicate *relevant* images for the searched category, and the negative labels *irrelevant* images.

In document retrieval framework, a strategy is to consider the *query concept*. The aim of this strategy is to refine the query according to the user labeling. A simple approach, called *query modification*, computes a new query by averaging the feature vectors of relevant images [1]. Another approach, the *query reweighting*, consists in computing a new similarity function between the query and a picture in the database. A usual heuristic is to weight the axes of the feature space [29]. In order to perform a better refinement of the similarity function, optimization-based techniques can be used. They are based on a mathematical criterion for computing the reweighting, for instance Bayes error [30], or average quadratic error [31,32]. Although these techniques are efficient for target search and monomodal category retrieval, they have difficulties to track complex image categories.

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 relevant and irrelevant classes [12]. In image retrieval, many techniques based on statistical learning have been proposed, for instance Bayes classification [33], k-Nearest Neighbors [34], Support Vector Machines [28,12,11,35], Gaussian Mixtures [36], or Gaussian random fields [37].

4.1 Statistical learning approach

We have chosen a statistical learning approach for the RETIN system because of its capacity to retrieve complex categories. This capacity is in part due to the possibility to work with kernel functions, with all the advantages we described in the previous sections.

However, a lot of strategies consider CBIR as a pure classification problem, and thus are not fully adapted to the special characteristics of this context. For instance, we have shown in a previous paper [38] that the few training data and the imbalance of the classes lead to a noisy boundary.

We summarize here the characteristics of our context:

(1) *High dimension.* Feature vectors are usually large (from 100 to 1000), which leads to the problem named as the *curse of dimensionality.*
(2) *Complex classes.* As image categories are unknown beforehand, it is difficult to make high assumptions about the distribution of the data. For instance, an usual Gaussian distribution assumption is rarely true. As a result, images of a given category can be dispatched in several small clusters.
(3) *Imbalance of data.* The size of the relevant class is very small against the size of the database (generally 100 times smaller). Thus, the context is fairly different from classification problems where both classes have a close size.
(4) *Few training data.* At the beginning of a retrieval session, the system must return results with very few labels. Furthermore, users will not give more than some hundreds of labels. As a result, the size of the training set is usually at most 1% of the database size.
(5) *Interactive learning.* The training set is built step by step, and each result depends on the previous ones.
(6) *Ranking vs error of classification.* System performances depend on the

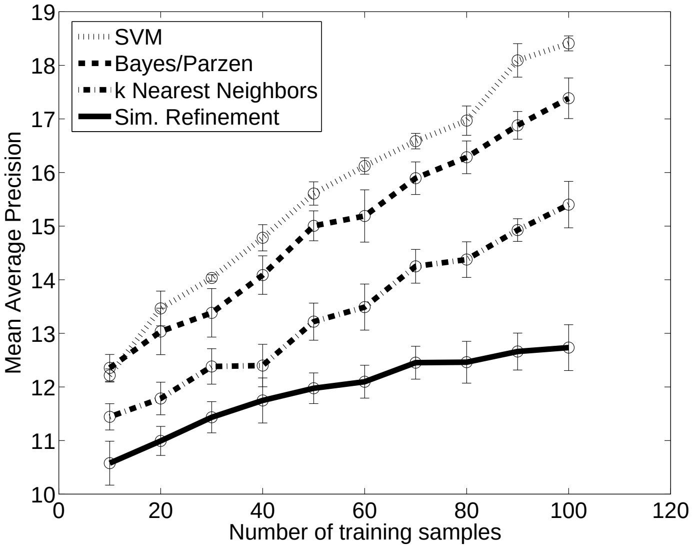


Fig. 6. Mean Average Precision(%) according to the size of the training set.

users satisfaction, which can be modeled by the Mean Average Precision. Thus, we aim at optimizing the ranking of images, instead of the usual classification error.

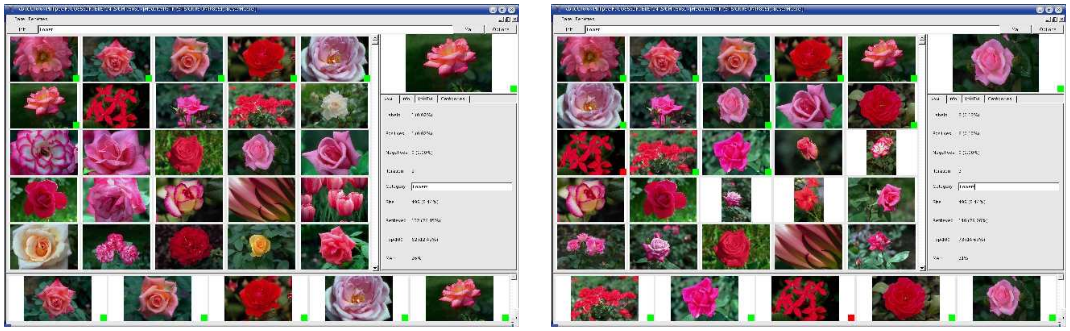
(7) *Computation time and scalability.* Our aim is to propose a system which can be used for real applications. Thus, we need fast methods, as we can not ask a non-expert user to wait for several minutes between each feedback step. A common way to define a fast and scalable method is to bound its computational complexity to $O(n)$, where n is the size of the database.

4.2 A comparison of classification methods for CBIR

The following methods have been evaluated :

- Similarity Refinement [32];
- Bayes classification [33];
- k-Nearest Neighbors [34];
- Support Vector Machines [12];
- Transductive Support Vector Machines [39];
- Kernel Fisher Discriminant [40].

The results in terms of Mean Average Precision are shown on Fig. 6, except for the TSVM and KFD which give results very close to inductive SVMs. One can see that the statistical methods give the best results, showing their interest towards geometric methods, like the similarity refinement. This also shows the interest of kernel based methods in order to deal with the high dimensions (1)



(a) First iteration

(b) Second iteration

Fig. 7. RETIN User Interface. Main part: ranked retrieved images; Right part: miscellaneous information about one image; Bottom part: images selected by active learning.

and the complex classes (2), since each of these methods (except the geometric one) are able to build efficient classifiers. In the sequel, we will use the SVM as the best method in this context, and because of its simple mathematical framework (hyperplan classifiers).

4.3 RETIN Active Learning Method

In order to deal with the imbalance of classes (3), the few training data (4) and the interaction with a user (5), we have opted for an active learning strategy. This strategy, which is already used in text [41] and image [42] retrieval, addresses the problem of the *selection* of the most interesting images the user should label. In first retrieval systems, a common strategy was to label the most relevant images. However, it has been shown that a different selection can lead to significantly better results [43].

We propose an active learning scheme to interact with a user searching for an image concept in the database. The process selects at each feedback step a set $I^\star$ of q images, displayed to the user for labeling.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be the image signatures, and $\mathbf{y} = \{y_1, \ldots, y_n\}$ the user labels ($y_i = 1$ if relevant, $y_i = -1$ if irrelevant, $y_i = 0$ if unlabeled). The examples are the images $i \in I$ with a non-zero label, *i.e.* couples $(\mathbf{x}_i, y_i)$ where $y_i \neq 0$,

Initialization. A retrieval session is initialized from one image given by the user. The top similar pictures are then displayed to the user.

Classification. A binary classifier is trained with the examples the user has given. We use a SVM with a Gaussian χ^2 kernel (*cf.* section 3). The result is a function $f_{\mathbf{y}}(\mathbf{x})$ which returns the relevance of each image $\mathbf{x}$, after a training with examples $(\mathbf{x}_i, y_i)$, $i \in I$.

Correction. We have shown in a previous paper that the classifier boundary is usually noisy during the first feedback step, because of scarcity of training samples (4) and the imbalance of classes (5) [38]. We propose to add an active correction of the boundary, which aims at translating the classifier boundary to an area of the feature space where the labels are the most uncertain. Details about this method can be found in [38].

Selection. When the user is not satisfied with the current classification, the system selects a set of images the user should label. The selection will be such as the labeling of those images will give the best performances. We divide the selection into three steps.

The first step aims at reducing the computational time (7), by pre-selecting some hundreds of pictures which may be in the optimal selection set. We propose to pre-select a set indexed by J of the closest pictures to the (corrected) boundary. This process is computed very fast, and the uncertainly-based selection method has proved its interest in CBIR context.

The second step is the computation of the selection criterion. In active classification, the criterion is the minimization of the error of classification (or *risk*). In these cases, the risk is computed for each classification function $f_{\mathbf{y},t(\mathbf{x}_i)}$, which is trained with the label $t(\mathbf{x}_i)$ of an unlabeled image $i \notin I$ added to current training set $\mathbf{y}$. Finally, the selected image $i^\star$ is the one which minimizes the risk:

$$i^\star = \underset{i \notin I}{\operatorname{argmin}} \ \operatorname{risk}(f_{\mathbf{y},t(\mathbf{x}_i)})$$

The main difficulty of this task is the fact that the label $t(\mathbf{x}_i)$ is unknown, and an estimation is required. This estimation is replaced by a *cost* function denoted $g_{\mathbf{y}}(\mathbf{x}_i)$, and including the pre-selection, the problem can be written as:

$$i^\star = \underset{i \in J}{\operatorname{argmin}} \ g_{\mathbf{y}}(\mathbf{x}_i)$$

Pure active classification techniques aim at minimizing the classification error. However, in our context, our aim is to optimize the image ranking, which can be modeled by the Mean Average Precision. Although decreasing classification error also increases the MAP, we have shown that the direct maximization of the MAP is more efficient [44]. Thus, we propose a precision-oriented cost function, which selects the images around the boundary that will increase the most this criterion. Details about this method can be found in [44].

The third step of active selection computes the batch selection. As we focus on real-time applications, we use a fast method close to the angle diversity

[45]. The method selects q images using the previously computed cost $g_{\mathbf{y}}(\mathbf{x}_i)$, and returns the set $I^\star$ of image indexes proposed for labeling:

```
$I^\star = \{\}$
**for** $l \in 1, ..., q$
    $i^\star = \operatorname*{argmin}_{i \in J - I^\star} \left( g_{\mathbf{y}}(\mathbf{x}_i) + \max_{j \in I \cup I^\star} s(\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)) \right)$
    $I^\star = I^\star \cup \{i^\star\}$
**endfor**
```

where $s(\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j))$ is the similarity between images $\mathbf{x}_i$ and $\mathbf{x}_j$.

Feedback. The user labels the selected images, and a new classification and correction are performed.

The process is repeated as many times as necessary.

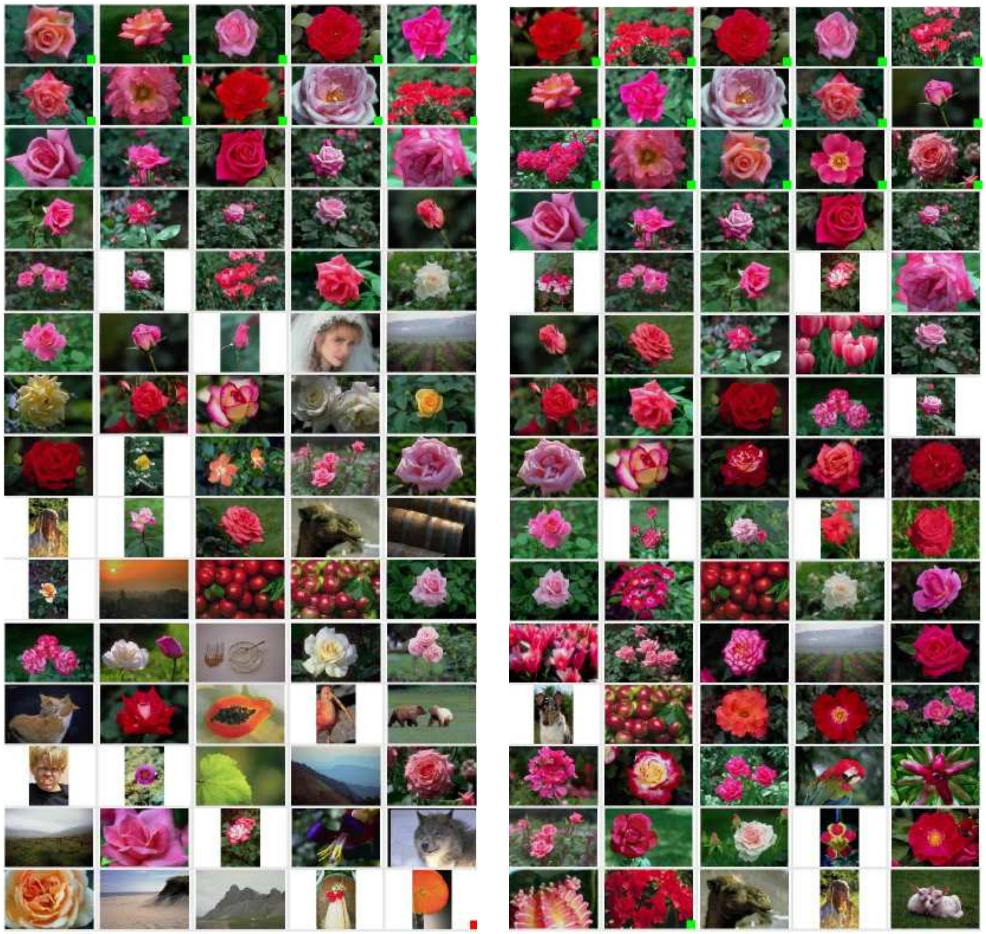
4.4 Experiments

An example of retrieval session is presented on Fig. 7. The interface is compound of three sub-parts. The main one at the top left displays the current ranking of the database. For instance on Fig. 7, we can see the closest pictures to the one brought by the user (top left, with a small green square). The sub-part at the bottom displays the current selection of the active learner. The user can give new labels by clicking the left or right mouse button. Once new labels are given, the user can ask for an update, and the new ranking is displayed in the main part. The right sub-part displays information about one image.

We show on Fig. 8 the 50 most relevant pictures after 3 and 5 iterations of 5 labels for the concept "roses", starting with the query of Fig. 7. One can see that the system is able to retrieve the images of the concept, while discriminating pictures with close visual characteristics. For instance, several non-rose pictures with very close colors and textures returned at the beginning of the search (*cf.* Fig. 7) are no more high-ranked 5 iterations later, while the relevant ones are still present (*cf.* Fig. 8).

4.5 Statistical evaluation

The RETIN active method introduced in this paper is compared to uncertainty-based methods : Tong SVM$_{active}$ [42], and Roy & McCallum method that aims



(a) 3 feedbacks

(b) 5 feedbacks

Fig. 8. The 75 most relevant pictures for the concept "roses". A small green square indicates an image labeled as relevant, and a red one an image labeled as irrelevant. (a) Top rank after 3 iterations of 5 labels. (b) Top rank after 5 iterations of 5 labels.

at minimizing the error of generalization [41]. A non active method, which randomly selects the images, is also considered for comparison.

The performances are evaluated by simulating the use of the system. For each simulation, an image category is randomly chosen and 100 images of the category are selected using one of the learning methods. After each SVM classification of the database, the Mean Average Precision is computed. These simulations are repeated many times in order to compute the mean and the standard deviation of the MAP (see appendix for details). The results of the experiments are shown in Fig. 9.

First, one can see the benefit of active learning in our context. In these experiments, the gain goes from 11% to 15%. The method which aims at minimizing the error of generalization is the less efficient active learning method. The most efficient method is RETIN active learning method, especially in the first it-

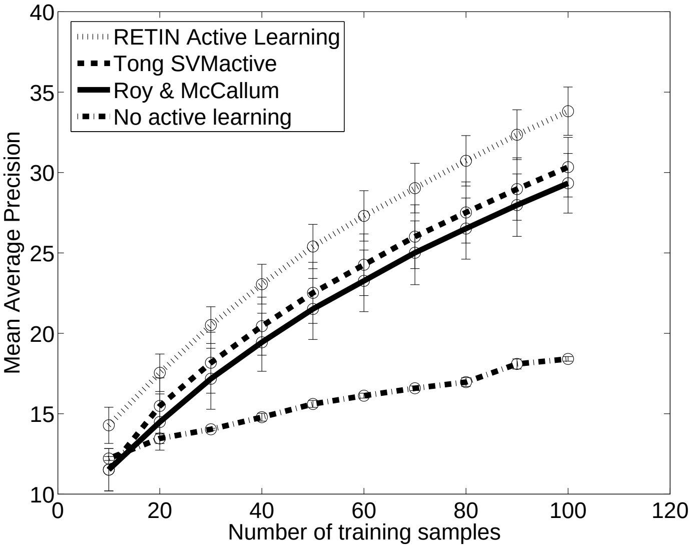


Fig. 9. Mean Average Precision(%) for different active learners.

erations, where the number of samples is the smallest. About computational time per feedback, the SVM$_{active}$ method needs at most 22ms, the method of Roy & McCallum several minutes, and the RETIN method at most 45ms.

We ran simulations with the same protocol that in the previous section, but changed the number of labels per feedback. In order to get comparable results, we ensure that the size of the training set at the end of a retrieval session is always the same:

- 30 feedbacks of 4 labels;
- 15 feedbacks of 8 labels;
- 8 feedbacks of 15 labels;
- 4 feedbacks of 30 labels;

We compute the precision/recall curves for all the concepts of the database. Results for the "savanna" concept are shown in Fig. 10; let us note that all concepts gave similar results modulo a scaling factor. As one can see on this figure, the more there is feedback steps, the more performances are increased. Increasing feedback steps leads to more classification updates, which allows a better correction and selection.

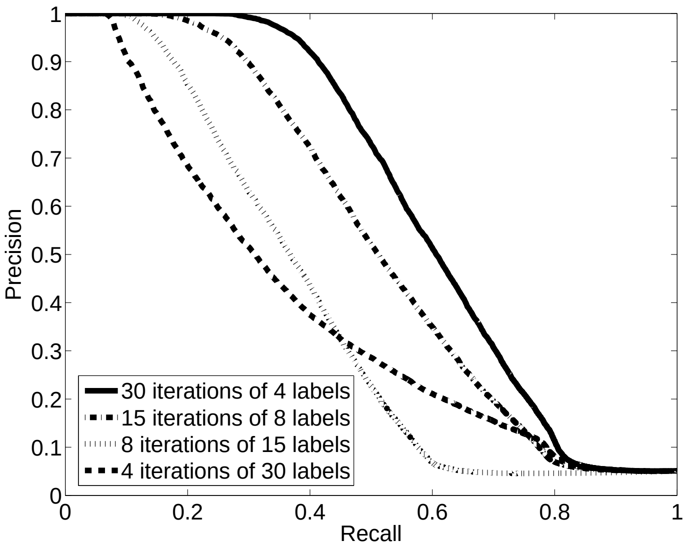


Fig. 10. Precision/Recall curves for the concept ‘savanna’ on the Corel Photo database.

5 Semantic Kernel Learning

Relevance feedback and active learning increase the system performances, but only during the current retrieval session. Once the session is over, labels are discarded. The purpose of this section is to present how the RETIN learning system uses all the labels accumulated during previous interactive sessions to improve the feature representation of the images. With such an optimized representation, we attempt to get a better match with semantic concepts. The labels are sampled from a hidden concept that the user has in mind during his retrieval session. Thus, if a large number of labels are available thanks to several retrieval sessions, their combinations should make the hidden concepts stand out.

Let us note *semantics* the whole information (users' annotations) accumulated during many retrieval sessions. Different strategies may be used to learn information about the database from this *semantics*:

- Some approaches deal with feature selection or competition [46]. The Latent Semantic Index and its kernel version have been proposed to model the correlation between feature variables [47].

- Other approaches compute and store a similarity matrix. A lot of approaches are based on the Kernel Alignment [48]. The idea is to adapt a kernel matrix (which is a particular similarity matrix) considering user labeling. This problem can be solved using semi-definite programming [1] [49]. However, it has been designed mostly for transduction and clustering, *i.e.*, two-class problems. For general database searches, there are many concepts or categories, overlapping each other. Some methods, building and updating a similarity matrix, have been experimented [50]. Usually, there is no assumption about the properties of the similarity matrix. For instance, the updated matrix may lost the induced metric properties. Moreover, these similarity matrix-based approaches have also a high computational cost. The memory complexity is at least $O(n^2)$, where n is the number of images in the database.

Our semantic learning RETIN strategy is based on a kernel matrix adaptation, and is designed to model mixed categories. We also manage the complexity constraint using efficient eigenvalue matrix decomposition; the method has a $O(n)$ complexity and memory need, and so it is applicable to large databases.

[1] Semi-definite programming allows efficient algorithms.

5.1 Adaptive approach

Let us note $\mathbf{K}_t$ the kernel matrix after $t-1$ retrieval sessions. Matrix $\mathbf{K}_t$ is symmetric and semi-definite positive *sdp* (*cf. 3.2*) . We propose algebraic transformations always keeping the *sdp* property of the kernel matrix.

The labels provided at session t are stored in vector $\mathbf{y}_t$ of size n, with 1 for relevant images, -1 for irrelevant images, and 0 for unlabeled images. After several uses of the system, the label sets can be gathered in a matrix such as the following one where each column represents a retrieval session:

$$\begin{array}{c|cccccccc}
 & \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 & \mathbf{y}_4 & \mathbf{y}_5 & \mathbf{y}_6 & \mathbf{y}_7 & \dots \\
\hline
\mathbf{x}_1 & 1 & 1 & 0 & 0 & -1 & 1 & 0 & \dots \\
\mathbf{x}_2 & 1 & 1 & 1 & 1 & -1 & 0 & 1 & \dots \\
\mathbf{x}_3 & 1 & 0 & 1 & -1 & 0 & 0 & 0 & \dots \\
\mathbf{x}_4 & 0 & -1 & 1 & 0 & 0 & -1 & 0 & \dots \\
\mathbf{x}_5 & -1 & 0 & 0 & 1 & 1 & -1 & 0 & \dots \\
\mathbf{x}_6 & 0 & 0 & -1 & 0 & 1 & 0 & -1 & \dots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots &
\end{array}$$

Labels give partial information about the category the user has in mind, a large majority of images is unlabeled for a given $\mathbf{y}_t$.

After retrieval process t, the current kernel matrix $\mathbf{K}_t$ is updated using the following expression:

$$\mathbf{K}_{t+1} = (1-\rho)\mathbf{K}_t + \rho \times \mathrm{merge}(\mathbf{K}_t, \mathbf{y}_t) \qquad (4)$$

where $\rho \in [0,1]$ is the system attentiveness, and $\mathrm{merge}(\mathbf{K}_t, \mathbf{y}_t)$ is an operator that returns a matrix containing the semantics from the previous sessions ($\mathbf{K}_t$) and the current session $\mathbf{y}_t$. This matrix must be *sdp* so that $\mathbf{K}_{t+1}$ keeps the *sdp* property.

5.2 Merging semantics of the previous and current sessions

Our first aim is both to increase the similarity between positive labeled images, and to decrease the similarity between negative and positive labeled images.

For this, we add the following kernel to the current one:

$$\mathbf{K}_{\mathbf{u}_t} = \mathbf{u}_t(\mathbf{u}_t)^\top \text{ with } u_{ti} = \begin{cases} 1 \text{ if } y_{ti} > 0 \\ -\gamma \text{ if } y_{ti} < 0 \\ 0 \text{ otherwise} \end{cases}$$

Parameter $\gamma \in [0, 1]$ handles the increasing of similarity between negative labeled images[2]. $\mathbf{K}_{\mathbf{u}_t}$ is a *sdp* matrix because of rank 1 with one positive eigenvalue ($||\mathbf{u}_t||^2$).

Our second aims is to average the similarities between all the positive labeled images. For that, we add the matrix $\mathbf{T}\mathbf{K}_t\mathbf{T}^\top$ to the current kernel matrix, with $\mathbf{T}_t$ a $n \times n$ matrix. To simplify the notation, let us consider that the q_+ first values of $\mathbf{y}_t$ are the positive ones. The matrix $\mathbf{T}_t$ is expressed as:

$$\mathbf{T}_t = \begin{pmatrix} \frac{1}{q_+} & \cdots & \frac{1}{q_+} & & & \\ \vdots & & \vdots & & & \\ \frac{1}{q_+} & \cdots & \frac{1}{q_+} & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{pmatrix}$$

It is also easy to prove the *sdp* property of $\mathbf{T}_t\mathbf{K}_t\mathbf{T}_t^\top$, if $\mathbf{K}_t$ is *sdp*, using the following property: $\mathbf{M}$ is *sdp* $\iff \forall \mathbf{x} \in \mathbb{R}^n,\ \mathbf{x}^\top\mathbf{M}\mathbf{x} \geq 0$.

As a result, the merging operator is:

$$\text{merge}(\mathbf{K}_t, \mathbf{y}_t) = \mathbf{T}_t\mathbf{K}_t\mathbf{T}_t^\top + b\mathbf{K}_{\mathbf{u}_t} \tag{5}$$

with $b \in \mathbb{R}^+$ so that diagonal terms of $\mathbf{T}_t\mathbf{K}_t\mathbf{T}_t^\top + b\mathbf{K}^{\mathbf{u}_t}$ equal 1.

5.3 Final operator

From eq. (4) and (5), the RETIN matrix kernel updating the semantic learning is:

$$\mathbf{K}_{t+1} = (1 - \rho)\mathbf{K}_t + \rho a(\mathbf{T}_t\mathbf{K}_t\mathbf{T}_t^\top + b\mathbf{K}_{\mathbf{u}_t}) \tag{6}$$

[2] In a multiple category context, negative labeled images are usually not in the same category. Thus in this case a small value (0.1) of γ is preferable.

Parameters a and b control the matrix progression during iterations.

5.4 Semantic kernel computation

We use a low-rank approximation matrix $\hat{\mathbf{K}}_t$, in order to have a storage linear to the size of the database. As the kernel matrix is real and symmetric, we are able to compute its eigendecomposition. The approximation consists in keeping the p largest eigenvalues. Thus, assuming that $p \ll n$, the storage of $\mathbf{K}_t$ is $O(n)$. Note that using this approximation, the kernel matrix can be seen as a linear kernel on the vectors of $\mathbf{X} = \mathbf{V}\sqrt{\mathbf{\Lambda}}$, where $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$ is the eigendecomposition of $\mathbf{K}$.

The direct computation of $\mathbf{K}_{t+1}$ is $O(n^2)$. We use a factorization technique for the computation of the eigenspectrum of $\mathbf{K}_{t+1}$. The factorization is followed by a QR decomposition and the computation of the eigenspectrum of a very small matrix (compared to n). This method has a $O(n)$ complexity.

5.5 Experiments

We compared the method proposed in this paper to a distance learning method [51] on the Corel Photo database (see appendix for details),

The semantic kernel matrix is initialized using the color and Gabor signatures previously introduced:

$$\mathbf{K}_{t=0} = \mathbf{X}^\top \mathbf{X}$$

with $\mathbf{X} = (\mathbf{x}_i)_{i \in [1,n]}$ the $p \times n$ distribution matrix, for which each column $\mathbf{x}_i$ is a vector representation of the ith image of the database.

In the following simulations, and for each semantic learning method, we optimize the kernel matrix using from 100 to 500 label sets of 100 non-zeros values. For each kernel, system performances are evaluated with the Mean Average Precision (*cf.* Appendix). Note that here we used a Gaussian $L2$ instead of the χ^2, since the resulting new feature vectors have negative values.

Parameter ρ. The method has been evaluated with ρ values 0.01, 0.05, 0.1, 0.5 and 1. As a rule, when ρ increases, the system learns faster. However, over 0.5 the learning becomes unstable: the MAP may increase a lot for some categories, whereas it decreases for other ones.

Parameter γ. The method has been evaluated with γ values 0.01, 0.05, 0.1, 0.5 and 1. The system has the best learning performances when $\gamma = 0.1$. Below this value, the system learns slowly, and above, the learning is inefficient: with a value of 1, the MAP decreases.

Number m of non-zero eigenvalues. The method has been evaluated with m values 10, 25, 50, 100 and 200. Globally, the higher will result in the better the performances. However, starting from a given value (here 50), performances do not increase much. Furthermore, it seems that the number of eigenvalues is mainly linked to the number of categories users are looking for, not to the number of images in the database. We experimented the system with 5 categories covering the whole database, and in this case 25 eigenvalues were enough.

In the following experiments, the default values are $\rho = 0.1$, $\gamma = 0.1$, and $m = 50$. Two scenarios are presented.

5.5.1 Online optimization

We first evaluate the kernel matrix optimization during the use of the retrieval system. The retrieval system is normally used during 100 sessions, and labels are stored. Then, we inject these 100 label sets into the semantic learning method, and get a new kernel matrix and/or feature vectors. The new feature vectors are then immediately used in next retrieval sessions. This process is then repeated every 100 retrieval sessions. Using this protocol for our method and the Xing distance learning method[51], the system has been evaluated every 100 retrieval sessions.

The results are shown in Fig. 11. The performances increase with our method, but not for the distance learning method of Xing. This is certainly because a distance learning method can not make high changes in the similarities between images. Furthermore, the categories in these experiments are mixed[3] , contrary to Xing experiments [51].

5.5.2 Offline optimization

We have also experimented the method when a partial knowledge on the database is available. For instance, one can have some keywords on sub-parts of the database. In order to simulate this partial knowledge, we randomly built 500 label sets of 50 positive and 50 negative values. Then we injected from 100 to 500 of these label sets in the semantic learner. The performances were evaluated for each size.

Fig. 12 shows the results. One can see that, with such semantic training sets,

[3] Mixed categories means that one image belongs to more than one category.

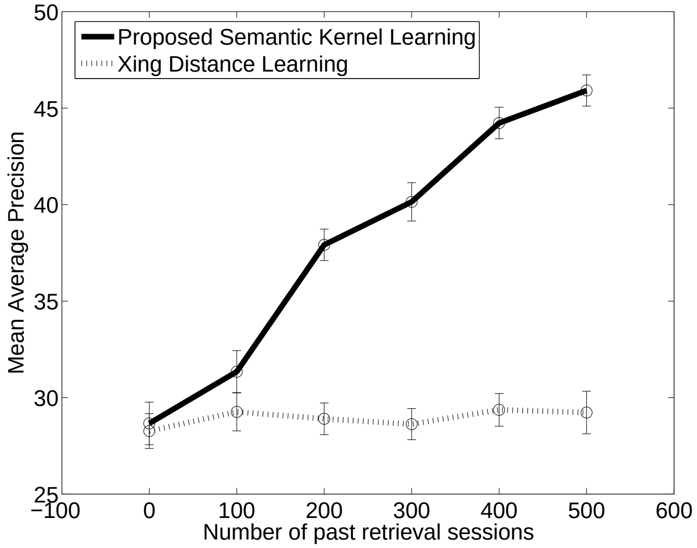


Fig. 11. Mean Average Precision (%) using an optimized kernel matrix, from 0 to 500 retrieval sessions, each retrieval session is initialized with 1 relevant image, a user performs 10 feedback step, and labels 10 images per feedback steps. Every 100 retrieval sessions, the 100 last label sets are injected into the semantic learner to optimize the kernel matrix.

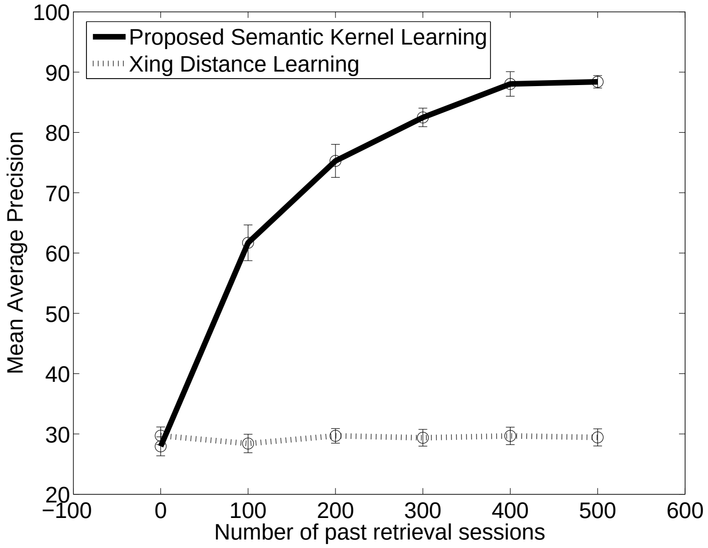


Fig. 12. Mean Average Precision(%) using an optimized kernel matrix, from 0 to 500 label sets. This protocol assumes that a partial knowledge (for instance, keywords) has been used to generate the label sets. Each label sets has 50 positive labels, and 50 negative labels.

the performances of our method widely increase with the training set size.

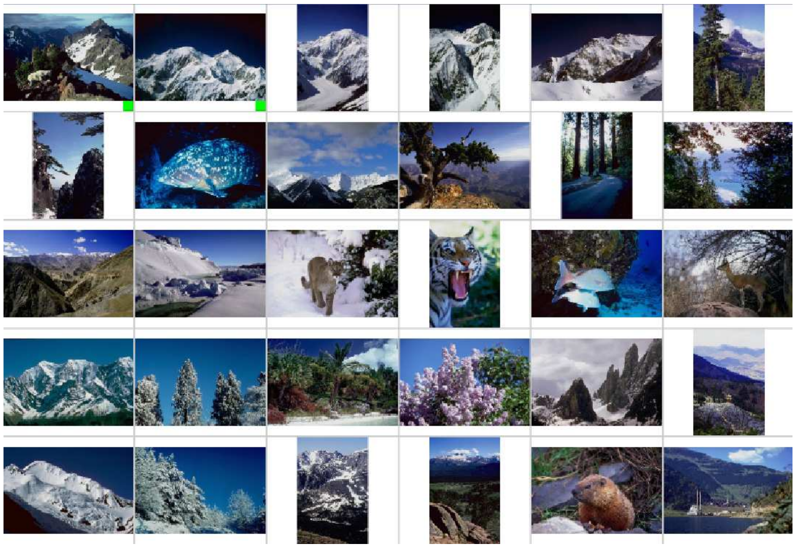


Fig. 13. Top rank before semantic learning. The most relevant pictures for the concept "mountains".

5.5.3 Other experiments

We have also compared our method with the distance learning method of Schultz and Joachim [52], that uses label sets with exclusively 2 positives and 1 negative values. Our method is still efficient with such a training set, but the distance learning does not improve the results, certainly for the same reason than for the Xing one.

Finally, an example of retrieval is reported on Fig. 13 (before semantic learning) and on Fig. 14 (after semantic learning). In both cases, the user is looking for mountains, and the query is composed of two positive examples (the images with a small green square in figures). Before optimization, there are irrelevant pictures amongst pictures the closest to the query. After optimization, since users have labeled mountains as being in the same concept during the previous sessions, the closest images all are mountains.

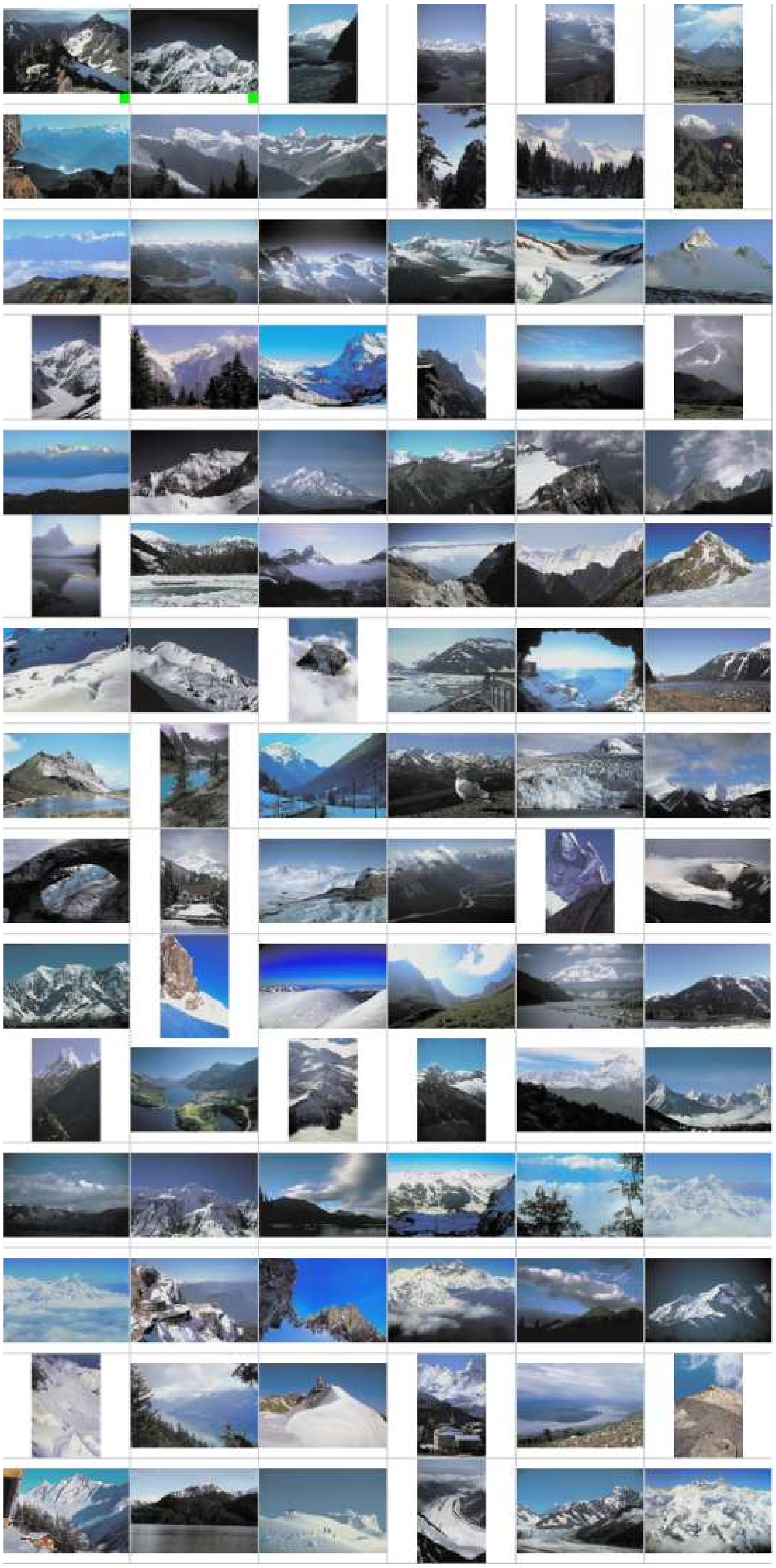


Fig. 14. Top rank after semantic learning. Most relevant pictures for the concept "mountains".

6 Conclusion

In this paper, a complete data mining system dedicated to image retrieval in large databases has been presented. It includes new solutions to the image indexing and to the database mining, both parts being improved throughout system use sessions.

Concerning image representation, we have opted for a dynamic quantization of the feature space and have proposed an adaptive quantization in two stages, which is both fast and efficient. The resulting color and texture-based codebooks perfectly match the content of the database. A nice trade-off between compactness and exhaustiveness of the image signatures is thus performed.

The core of the retrieval system is the similarity measure. We used kernel functions to represent similarity. This framework allows us to well separate the image coding from the latter processing such as classification, ranking, learning. We have compared various kernel functions and various classifiers. In our context of semantic category retrieval in large databases of general photographs, with very few training data, a SVM with a Gaussian kernel is the best choice.

Another contribution of the paper is our active learning scheme, that exploits the Mean Average Precision statistic in the generalization error criterion to boost the retrieval process. Adding to a specific SVM boundary correction, the RETIN active learning strategy outperforms the state-of-the-art methods proposed by Tong & Chang and Roy & Mc Callum.

Finally, we have also proposed a method to keep the semantic categories build by the various users over the sessions, even if categories are mixed. The kernel matrix framework is extended to learn new similarity matrices as soon as additional user information is available. It is an efficient way to improve the retrieval quality within large databases, since the MAP is multiplied by two after 500 retrieval sessions compared to a single session. This performance can be much more improved by injecting prior knowledge such as a partial classification of the database.

A perspective of this work is to translate this active learning scheme to primitives extracted from the images such as regions or points of interest in order to be able to answer other requests such as partial queries.

Appendix

CBIR tests are carried out on the generalist Corel Photo database, which contains more than $50,000$ pictures. To get tractable computation for the statistical evaluation, we randomly selected 77 of the Corel folders, to obtain a database of $6,000$ images. To perform interesting evaluation, we built from this database 50 categories of different sizes and complexities like birds (219), castles (191), doors (199), Europe (627), food (315), mountains (265) ...

The CBIR system performances are measured using precision(P), recall(R) and statistics computed on P and R for each category. We use the mean average precision (MAP) which represents the value of the P/R integral function. This metric is used in the TREC VIDEO conference[4] , and gives a global evaluation of the system (over all the (P,R) values).

The performances are evaluated by simulating the use of the system. For each simulation, an image category is randomly chosen and 100 images of the category, drawn at random or with active learning, constitute the learning set for the SVM. After each classification of the database, the Mean Average Precision (MAP) is computed. These simulations are repeated 1000 times, and all values of MAP are averaged. Next, we repeat ten times these simulations to get the mean and the standard deviation of the MAP.

References

[1] Y. Rui, T. Huang, S. Mehrotra, M. Ortega, A relevance feedback architecture for content-based multimedia information retrieval systems, in: IEEE Workshop on Content-Based Access of Image and Video Libraries, 1997, pp. 92–89.

[2] S. Santini, A. Gupta, R. Jain, Emergent semantics through interaction in image databases, IEEE Transactions on Knowledge and Data Engineering 13 (3) (2001) 337–351.

[3] A. Mojsilovic, B. Rogowitz, Capturing image semantics with low-level descriptors, in: International Conference in Image Processing (ICIP'01), Vol. 1, Thessaloniki, Greece, 2001, pp. 18–21.

[4] C. Schmid, Weakly supervised learning of visual models and its application to content-based retrieval, International Journal of Computer Vision 56 (1) (2004) 7–16.
URL `http://lear.inrialpes.fr/pubs/2004/Sch04`

[4] http://www-nlpir.nist.gov/projects/trecvid/

[5] Y. Chen, J. Wang, Image categorization by learning and reasoning with regions, International Journal on Machine Learning Research 5 (2004) 913–939.

[6] N. Vasconcelos, M. Kunt, Content-based retrieval from image databases: current solutions and future directions, in: International Conference in Image Processing, Vol. 3, Thessaloniki, Greece, 2001, pp. 6–9.

[7] R. Veltkamp, M. Tanase, Content-based image retrieval systems : A survey, Technical report UU-CS-2000-34, Department of Computing Science, Utrecht University (October 2000).

[8] Y. Ishikawa, R. Subramanya, C. Faloutsos, MindReader: Querying databases through multiple examples, in: Proc. 24th Int. Conf. Very Large Data Bases, VLDB, 1998, pp. 218–227.

[9] Y. Rui, T. Huang, Optimizing learning in image retrieval, in: Conf on Computer Vision and Pattern Recognition (CVPR), Vol. 1, Hilton Head, SC, 2000, pp. 236–243.

[10] N. Sebe, I. Cohen, A. Garg, T. Huang, Machine Learning in Computer Vision, Springer Verlag, ISBN 1-4020-3274-9, 2005.

[11] S. Tong, D. Koller, Support vector machine active learning with application to text classification, Journal of Machine Learning Research (2001) 2:45–66.

[12] O. Chapelle, P. Haffner, V. Vapnik, Svms for histogram based image classification, IEEE Transactions on Neural Networks 10 (1999) 1055–1064.

[13] M. Cord, P. Gosselin, S. Philipp-Foliguet, Stochastic exploration and active learning for image retrieval, Image and Vision Computing (25) (2006) 14–23.

[14] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Advances in knowledge discovery and data mining, AAAI/MIT Press.

[15] J. Smith, S. Chang, VisualSEEK: a fully automated content-based image query system, in: ACM Multimedia Conference, Boston, USA, 1996, pp. 87–98.

[16] B. Manjunath, J.-R. Ohm, V. Vasudevan, A. Yamada, Color and texture descriptors, IEEE Transactions on Circuits and Systems for Video Technology 11 (6) (2001) 703–715.

[17] Y. Linde, A. Buzo, R. Gray, An algorithm for vector quantizer design, IEEE Transaction on Communication 28 (1980) 84–94.

[18] G. Patanè, M. Russo, The enhanced LBG algorithm, Neural Networks 14 (9) (2001) 1219–1237.

[19] 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.

[20] M. Swain, D. Ballard, Color indexing, International Journal of Computer Vision 7 (1) (1991) 11–32.

[21] S. Kullback, Information Theory and Statistics, Wiley (New York), 1959.

[22] J. Puzicha, T. Hofmann, J. Buhmann, Non-parametric similarity measures for unsupervised texture segmentation and image retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 1997, pp. 267–272.

[23] I. Sethi, N. Patel, Statistical approach to scene change detection, in: Storage and Retrieval for Image and Video Databases, 1995, pp. 329–338.

[24] M. Stricker, M. Orengo, Similarity of color images, in: SPIE, Storage and Retrieval for Image Video Databases III, Vol. 2420, 1995, pp. 381–392.

[25] Y. Rubner, Perceptual metrics for image database navigation, Ph.D. thesis, Stanford University (May 1999).

[26] V. Vapnik, Statistical Learning Theory, Wiley-Interscience, New York, 1998.

[27] J. Shawe-Taylor, N. Cristianini, Kernel methods for Pattern Analysis, Cambridge University Press, ISBN 0-521-81397-2, 2004.

[28] E. Chang, B. T. Li, G. Wu, K. Goh, Statistical learning for effective visual information retrieval, in: IEEE International Conference on Image Processing, Barcelona, Spain, 2003, pp. 609–612.

[29] S. Aksoy, R. Haralick, F. Cheikh, M. Gabbouj, A weighted distance approach to relevance feedback, in: IAPR International Conference on Pattern Recognition, Vol. IV, Barcelona, Spain, 2000, pp. 812–815.

[30] J. Peng, B. Bhanu, S. Qing, Probabilistic feature relevance learning for content-based image retrieval, Computer Vision and Image Understanding 75 (1-2) (1999) 150–164.

[31] N. Doulamis, A. Doulamis, A recursive optimal relevance feedback scheme for cbir, in: International Conference in Image Processing (ICIP'01), Thessaloniki, Greece, 2001.

[32] J. Fournier, M. Cord, S. Philipp-Foliguet, Back-propagation algorithm for relevance feedback in image retrieval, in: International Conference in Image Processing (ICIP'01), Vol. 1, Thessaloniki, Greece, 2001, pp. 686–689.

[33] N. Vasconcelos, Bayesian models for visual information retrieval, Ph.D. thesis, Massachusetts Institute of Technology (2000).

[34] S.-A. Berrani, L. Amsaleg, P. Gros, Recherche approximative de plus proches voisins : application la reconnaissance d'images par descripteurs locaux, Technique et Science Informatiques (2003) 22(9):1201–1230.

[35] B. L. Saux, Classification non exclusive et personnalisation par apprentissage : Application à la navigation dans les bases d'images, Ph.D. thesis, INRIA (2003).

[36] N. Najjar, J. Cocquerez, C. Ambroise, Feature selection for semi supervised learning applied to image retrieval, in: IEEE ICIP, Vol. 2, Barcelena, Spain, 2003, pp. 559–562.

[37] X. Zhu, Z. Ghahramani, J. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, in: International Conference on Machine Learning, 2003.

[38] P. Gosselin, M. Cord, RETIN AL: An active learning strategy for image category retrieval, in: IEEE International Conference on Image Processing, Vol. 4, Singapore, 2004, pp. 2219–2222.

[39] T. Joachims, Transductive inference for text classification using support vector machines, in: Proc. 16th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, 1999, pp. 200–209.

[40] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, K.-R. Müller, Fisher discriminant analysis with kernels, in: Y.-H. Hu, J. Larsen, E. Wilson, S. Douglas (Eds.), Neural Networks for Signal Processing IX, IEEE, 1999, pp. 41–48.
URL `citeseer.ist.psu.edu/mika99fisher.html`

[41] N. Roy, A. McCallum, Toward optimal active learning through sampling estimation of error reduction, in: International Conference on Machine Learning, 2001.

[42] S. Tong, E. Chang, Support vector machine active learning for image retrieval, in: ACM Multimedia, 2001, pp. 107–118.

[43] D. Cohn, Active learning with statistical models, Journal of Artificial Intelligence Research 4 (1996) 129–145.

[44] P. Gosselin, M. Cord, Precision-oriented active selection for interactive image retrieval, in: IEEE International Conference on Image Processing, Atlanta, GA, USA, 2006.

[45] K. Brinker, Incorporating diversity in active learning with support vector machines, in: International Conference on Machine Learning, 2003, pp. 59–66.

[46] H. Müller, W. Müller, D. M. Squire, S. Marchand-Maillet, T. Pun, Long-term learning from user behavior in content-based image retrieval, Tech. rep., Computer Vision Group, University of Geneva, Switzerland (2000).

[47] D. R. Heisterkamp, Building a latent semantic index of an image database from patterns of relevance feedback, in: International Conference on Pattern Recognition, Quebec City, Canada, 2002, pp. (4):132–137.

[48] N. Cristianini, J. Shawe-Taylor, A. Elisseff, J. Kandola, On kernel target alignement, in: Neural Information Processing Systems, Vancouver, Canada, 2001.

[49] G. R. G. Lanckriet, N. Cristianini, N. Bartlett, L. El Ghaoui, M. I. Jordan, Learning the kernel matrix with semi-definite programming, in: International Conference on Machine Learning, Sydney, Australia, 2002.

[50] J. Fournier, M. Cord, Long-term similarity learning in content-based image retrieval, in: International Conference in Image Processing (ICIP), Rochester, New-York, USA, 2002.

[51] E. P. Xing, A. Y. Ng, M. I. Jordan, S. Russell, Distance metric learning, with application to clustering with side-information, in: Neural Information Processing Systems, Vancouver, Bristish Columbia, 2002.

[52] M. Schultz, T. Joachims, Learning a distance metric from relative comparisons, in: Neural Information Processing Systems, 2003.