Semantic Kernel Updating for Content-Based Image Retrieval

Philippe-Henri Gosselin, Matthieu Cord

Semantic Kernel Updating for Content-Based Image Retrieval

Philippe H. Gosselin
ETIS / CNRS UMR 8051
ENSEA, 6, avenue du Ponceau,
95014 Cergy-Pontoise, France
gosselin@ensea.fr

Matthieu Cord
ETIS / CNRS UMR 8051
ENSEA, 6, avenue du Ponceau,
95014 Cergy-Pontoise, France
cord@ensea.fr

Abstract

A lot of relevance feedback methods have been proposed to deal with Content-Based Image Retrieval (CBIR) problems. Their goal is to interactively learn the semantic queries that users have in mind. Interaction is used to fill the gap between the semantic meaning and the low-level image representations.

The purpose of this article is to analyze how to merge all the semantic information that users provided to the system during past retrieval sessions. We propose an approach to exploit the knowledge provided by user interaction based on binary annotations (relevant or irrelevant images). Such semantic annotations may be integrated in the similarity matrix of the database images. This similarity matrix is analyzed in the kernel matrix framework. In this context, a kernel adaptation method is proposed, but taking care of preserving the properties of kernels. Using this approach, a semantic kernel is incrementally learnt.

To deal with practical constraint implementations, an eigendecomposition of the whole matrix is considered, and a efficient scheme is proposed to compute a low-rank approximated kernel matrix. It allows a strict control of the required memory space and of the algorithm complexity, which is linear to the database size.

Experiments have been carried out on a large generalist database in order to validate the approach.

1. Introduction

Content-Based Image Retrieval (CBIR) has attracted a lot of research interest in recent years. This paper addresses the problem of category search, which aims at retrieving all images belonging to a given category from an image database. Content-Based Image Retrieval is characterized by the gap between high-level semantic and low-level image representation [18]. Users are looking for an image set with a semantic meaning, whereas systems deal with low-level features.

Contrary to the early systems, where people were focused on fully automatic strategies, recent approaches introduce human-computer interaction into CBIR [20, 19]. Starting with a coarse query, 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 [2], but most of the time, user information consists of binary annotations (labels) indicating whether or not the image belongs to the desired category. In such a strategy, the system uses these labels to refine the computation of new relevant pictures. Many efficient methods have been proposed in the CBIR community. However, as efficient as a relevance feedback method is, it is still limited by the low-level image representation.

In a "relevance feedback only" framework, labels provided by users are never reused: this knowledge is lost at the end of each retrieval session. Some researchers from CBIR community propose to memorize this semantic information [6, 10, 14]. The accumulation of labels during many retrieval sessions constitutes a knowledge about the database content. Semantic learning, which exploits this information, enables the system to enhance the representation of the pictures, and thus its overall performances.

In this paper, we introduce a new approach to manage this knowledge. All the semantic information passed by users to the system during many retrieval sessions is analyzed. As it is composed of binary annotations, we can directly represent it in the similarity matrix of the database images. This similarity matrix is analyzed in the kernel matrix framework. In this context, some constraints have to be verified in order to get the kernel matrix properties. That means we can not manipulate these matrices, for instance by modifying some values, without taking strong precautions. A kernel adaptation method is proposed, taking care of keeping the nice metric properties of kernels. The proposed modifications aims at enhancing the similarity between any images which have been labelled as relevant dur-

ing one of the retrieval sessions.

The exploitation of this semantic matrix arises major difficulties in terms of storage and computational complexity. To deal with practical constraint implementations, eigendecomposition of the whole matrix is considered, and an efficient scheme is proposed to compute a low-rank approximated kernel matrix. This optimization scheme allows a strict control of the required memory space and of the algorithm complexity, which is linear to the database size ($O(N)$ complexity, where N is the size of the database).

This approach allows the use of any kernel- or metric-based relevance feedback method. After any retrieval session, the kernel matrix may be updated in order to reflect user average behavior. In order to validate the approach, our method is experimented with a SVM classifier and an active learner on a generalist image database.

In this scope, we first present in Section 2 some methods aiming at learning semantics, from CBIR and statistical learning community, and then our framework to build an efficient semantic learner. In Section 3, the adaptive approach, based on a kernel matrix updating technique, is proposed. In Section 4, a method to build the semantic kernel according to user labelling is introduced. In Section 5, the practical implementation strategy based on eigenvalue kernel matrix analysis is presented. In Section 6, experiments carried out on the COREL photo database are reported.

2. Semantic learning

2.1. CBIR approaches

Researchers from CBIR community propose solutions to store and exploit semantics, usually with an *ad hoc* approach.

Some approaches are the competition of feature space dimensions [14]. These methods have no high storage problems and allow the use of different feedback learning approaches. Otherwise they can not memorize many semantic links.

Some other approaches compute and store a similarity matrix [7]. For a set of user annotations, an heuristic updates some parts of the similarity matrix to reflect users general behavior. Such approaches fill the similarity matrix with no assumption about its properties. For instance, the matrix can not induce a metric: there is no possible generalization about the relevance between two pictures. It also suffer from high memory needs: the system has to store N^2 elements, with N the size of the database. This is not feasible for large values of N. For instance, for an one million image database, similarity matrix using float numbers would require 4TB (4×10^{12} Bytes).

Some researchers propose to perform a clustering of the database to enhance system performances [10]. These approaches can help in the storage of similarities, and in the browsing of a database. Such methods improve retrieval process using relevance feedback, but they usually need a specific relevance feedback tuning.

2.2. Statistical learning approaches

Researchers from statistical learning community also propose solutions to store and exploit semantics, usually with strong theoretical models.

Actually, a lot of approaches are based on the Kernel Alignment [5]. The idea is to adapt a kernel matrix (which is a particular similarity matrix) considering user labelling. This problem can be solved using semi-definite quadratic programming[1] [13], and uses a low-rank approximation of kernel matrix. These methods seem to be interesting for CBIR, and should be experimented in this context. However, they have been designed mostly for transduction and clustering, $i.e.$, two class problems. They have also a high computational cost.

Other approaches have also been proposed – the Latent Semantic Index and its kernel version [4]. Latent semantic index are used for text retrieval, and solutions for CBIR have been proposed [11]. They are tuned to build a representation of the semantics from a set of user labels.

2.3. Semantic learning framework

CBIR and statistical learning approaches have each one its advantages and drawbacks. The first one allows to memorize more semantics, but with an *ad hoc* modeling of the problem, which needs dedicated feedback learning. The later one lets memorize less semantics, but strong theoretical learning models can be used for relevance feedback.

Considering all these points, we propose the following constraints in order to build an efficient semantic learner:

- *Scalability / Processing of huge databases.* A complexity in terms of computation and memory needs higher than $O(N)$ quickly bounds the scalability of a retrieval system. For instance, a $O(N^2)$ complexity on current computers limits the maximum number of pictures to be around 10,000.
- *General framework easy to combine with relevance feedback techniques.* A lot of work has been made in relevance feedback. Actually, several efficient methods have been proposed. Using a dedicated relevance feedback method needs the rebuilding of learning models for each specific approach.
- *System must be dynamic.* The update of a semantic learner should be able to learn any semantic in any

1 Semi-definite programming allows efficient algorithms.

case. Suppose that users change their mind about some part of the database, if the semantic learner stalls in its current state, it becomes useless and perhaps can decrease system performances.

In the following sections, we propose an adaptive approach and a method considering these constraints.

3. Adaptive kernel matrix approach

One way to store semantics is to store similarities between pictures of the database, according to the system usage. These informations can be store in a $N \times N$ matrix K^s, with N the database size.

3.1. Similarity matrix as a kernel

In our framework, the capability to combine the semantic learning with relevance feedback methods is required.

Interactive retrieval techniques are mainly of two types: statistical and geometrical [19, 16]. The geometrical methods refer to search-by-similarity systems [12, 17]. The objective of the statistical methods is to update a relevance function [1, 3] or a binary classification of images using the user annotations. Recently, statistical learning approaches have been introduced in CBIR context and have been very successful [19, 2].

Statistical learning method are usually kernel-based techniques. In this paper, we are dealing with the class of kernels k that correspond to dot product in induced space $\mathcal{H}$ via a map Φ:

$$\begin{aligned} \Phi \quad : \quad & \mathbb{R}^p \rightarrow \mathcal{H} \\ & x \mapsto \Phi(x) \end{aligned}$$

that is,

$$k(\mathbf{x}, \mathbf{x}') = < \Phi(\mathbf{x}), \Phi(\mathbf{x}') >$$

In practice, $\Phi(\mathbf{x})$ in never computed because the induced space may be very large, and sometimes infinite. An usual way to perform this computation is to use the *kernel trick*: build a function respecting some conditions (for instance, Mercer's conditions) or store all possible values of $k(.,.)$ on a finite input space, for instance all the pictures of a database:

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$$

The resulting $N \times N$ matrix K is called the Gram matrix. This matrix is symmetric and semi-definite positive. The Gram matrix is a particular similarity matrix.

Let $D = (\mathbf{x}_i)_{i \in [1,N]}$ be the $p \times N$ matrix, for which each column $\mathbf{x}_i$ is a vector representation of the ith picture of the database. The following matrix:

$$K = D'D$$

is the dot product matrix ($i.e.$ $K_{ij} = < \mathbf{x}_i, \mathbf{x}_j >$). Dot product may also be seen as a similarity metric. For two close (collinear) vectors, scalar dot product is large, and for two far (orthogonal) vectors, scalar dot product is small.

Thus, if the similarity matrix K^s is a Gram matrix (symmetric sdp matrix[2]), we can use it as a kernel. It follows the use of any kernel-based method for relevance feedback, or any metric-based method with metric d defined on $(\mathbf{x}, \mathbf{x}')$ by:

$$d(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}) + k(\mathbf{x}', \mathbf{x}') - 2k(\mathbf{x}, \mathbf{x}')$$

3.2. Adaptive approach

In order to store semantics all along system usage, the similarity matrix K_t^s at the end of any retrieval session t can be updated. At this step, user has provided several annotations through relevance feedback. These labels are stored in a N vector $\mathbf{y}_t$, with 1 value for relevant pictures, -1 value for irrelevant pictures, and 0 value for unlabelled pictures.

These labels are semantic about the current search category. All pictures with a 1 value in $\mathbf{y}_t$ are in the same category and all pictures with a -1 value in y are not in the category. Thus, we can build a new similarity matrix $g_{\mathbf{y}_t}(K_t^s)$ according to $\mathbf{y}_t$, with $g_{\mathbf{y}_t}$ a matrix operator.

Assuming that $g_{\mathbf{y}_t}(K_t^s)$ correctly stores all the semantic in $\mathbf{y}_t$, we should not replace the current similarity matrix by this one. Several possible categories can be found on the same database, and all these categories do not build a clustering: many pictures can be in several categories. In order to grasp this polysemic meaning, we choose an adaptive approach. We combine the current semantic with the new one using a weighted sum:

$$K_{t+1}^s = (1 - \rho) K_t^s + \rho g_{\mathbf{y}_t}(K_t^s) \tag{1}$$

where weight $\rho \in [0, 1]$ is the *vigilance* parameter. This parameter is the amount of changes one which to apply to current knowledge – how much system must handle new labels. This parameter can not be removed from such a system, for trivial reasons. It must be tuned according to a given application. If labels come from an expert, ρ should be high. If labels come from an anonymous user (for instance, a person seeking pictures on the internet), ρ should be low.

Thus, after each retrieval process t, the current similarity matrix is updated according to user labels. This similarity matrix is representing the average semantic provided by users.

4. Semantic kernel

In the previous section, we assumed that the matrix operator $g_{\mathbf{y}_t}(.)$ introduced in eq. 1 correctly stores all the semantic in $\mathbf{y}_t$. This section proposes a choice for $g_{\mathbf{y}_t}(.)$.

2 *sdp* denotes semi-definite positive.

The aim of the $g_{\mathbf{y}_t}(.)$ operator is to return a sdp matrix integrating the y_t semantic. A first choice of $g_{\mathbf{y}_t}(.)$ should be this one:

$$g_{\mathbf{y}_t}(K_t^s) = \mathbf{y}_t\mathbf{y}_t'$$

Such a matrix explicitly catches the semantic expressed in $\mathbf{y}_t$. However, we are in a global semantic update scheme. This matrix stores high similarity between relevant pictures, low similarity between relevant and irrelevant pictures, but also high similarity between irrelevant pictures. Considering a database with only 2 categories, high similarity between irrelevant pictures would not be a problem, but in our case, we have no assumption about this.

We propose to use the following matrix:

$$E = \mathbf{u}\mathbf{u}' \text{ with } u_i = \begin{cases} 1 & \text{if } i \in I^+ \\ -\gamma & \text{if } i \in I^- \\ 0 & \text{otherwise} \end{cases}$$

where $\gamma \in [0, 1]$ is the *joint of irrelevant data* parameter, I^+ the indexes of positives values of y_t, and I^- the indexes of negatives values of y_t.

The E matrix has close to zero similarity between irrelevant pictures (γ^2), and low similarity between relevant and irrelevant pictures ($-\gamma$).

In order to speed up the learning of the semantic, we propose to use a matrix T to propagate new semantic in the kernel matrix:

$$T = \begin{pmatrix} T^+ & 0 \\ 0 & Id \end{pmatrix}$$

with $T^+ = \frac{1}{q^+}$, $q^+ = |I^+|$. We suppose in this representation of T, that all relevant labels have indexes in $[1, q^+]$.

Computing TK_t^sT' averages the relevant row/columns of the kernel matrix. All relevant pictures will have the same similarity between each other, and also the same self-similarity. All relevant pictures will have the same similarity with all other pictures.

The use of these matrices leads to the following operator:

$$g_{\mathbf{y}_t}(K_t^s) = a \times (TK_t^sT' + bE) \tag{2}$$

with $a, b \in \mathbb{R}^+$.

The matrix returned by $g_{\mathbf{y}_t}$ is symmetric sdp, because of the following properties of symmetric semi-definite positive matrices: Let M be a $N \times N$ symmetric matrix:

- M is $sdp \iff \forall \mathbf{x} \in \mathbb{R}^N,\ \mathbf{x}'M\mathbf{x} \geq 0$;
- M is $sdp \iff$ eigenvalues of M are positives or zero;
- Let E be a $N \times N$ symmetric sdp matrix. Then: M is $sdp \Rightarrow M + E$ is symmetric sdp;
- Let T be a $N \times N$ matrix. Then: M is $sdp \Rightarrow TMT'$ is symmetric sdp.

We compute b such as relevant diagonals of $TK_t^sT' + bE$ are 1, and a such as $\sum_{ij}(K_{t+1}^s)_{ij} = \sum_{ij}(K_t^s)_{ij}$.

5. Kernel computation

5.1. Kernel matrix decomposition

The whole $N \times N$ kernel matrix can not be stored in memory – such an approach would not be linear with the database size.

In order to compress the information included in this matrix, we propose to use a low-rank approximation of this one. As the kernel matrix is real and symmetric, we are able to compute its eigendecomposition. The approximation consists of keeping the m largest eigenvalues:

$$K_t^s = V_t\Lambda_tV_t' \tag{3}$$

with V_t as the $N \times m$ eigenvector matrix, and Λ_t the $m \times m$ eigenvalue diagonal matrix.

This algebraic approximation should however preserve most of the information (sometimes it can even be the exact development of the whole matrix). The tuning of m parameter will be discussed in the experiment section.

Assuming that $m \ll N$, the storage of K^s is considered as linear with the size of the database.

5.2. Iterative scheme

This section proposes a method to compute K_{t+1}^s, with a complexity linear to the database size. We denote by I the indexes of non-zero values of y_t, $q = |I|$. For sake of simplicity, we suppose that the q first values of y_t are the non-zeros values ($I = \{1, ..., q\}$).

Thanks to Eq. 1 and Eq. 2, K_{t+1}^s may be expressed as follows:

$$\begin{aligned} K_{t+1}^s &= (1-\rho)K_t^s + \rho g(K_t^s) \\ &= (1-\rho)K_t^s + \rho a(TK_t^sT' + bE) \end{aligned}$$

We have to develop this expression in order to find an approximation compatible with our constraints.

First, we use the eigendecomposition of K_t^s (*cf.* Eq. 3):

$$K_{t+1}^s = (1-\rho)V_t\Lambda_tV_t' + \rho a(TV_t\Lambda_tV_t'T' + bE) \tag{4}$$

Our solution aims at computing the eigendecomposition of K_{t+1}^s in an efficient way. That will allow us to approximate K_{t+1}^s using only the m largest eigenvalues. The scheme may be iterated in that way.

A straight eigenvalue decomposition is not feasible. As one can see in Eq. 4, there is no simple way to compute Λ_{t+1} and V_{t+1} from Λ_t and V_t. We propose a matrix factorization and an adapted QR decomposition to reduce to a small eigendecomposition problem:

$$K_{t+1}^s = ABA'$$

with the $N \times (q + 2m)$ matrix A:

$$A = \begin{pmatrix} Id & TV_t & V_t \end{pmatrix}$$

and the $(q+2m) \times (q+2m)$ matrix B:

$$B = \begin{pmatrix} \rho abE & 0 & 0 \\ 0 & \rho a \Lambda_t & 0 \\ 0 & 0 & (1-\rho)\Lambda_t \end{pmatrix}$$

The QR decomposition of A (detailed in Appendix A) leads to a smaller eigenproblem. Indeed, if $A = QR$, with Q an orthonormal $N \times (q+m)$ matrix, and R an upper triangular $(q+m) \times (q+2m)$ matrix, then it follows that:

$$\begin{aligned} K^s_{t+1} &= ABA' \\ &= QRB(QR)' \\ &= Q(\underbrace{RBR'}_{UMU'})Q' \\ &= (QU)M(QU)' \end{aligned}$$

with UMU' is the eigendecomposition of the $(m+q) \times (m+q)$ matrix RBR'.

Finally, we approximate K^s_{t+1} with its m largest eigenvalues:

$$K^s_{t+1} \simeq V_{t+1}\Lambda_{t+1}V'_{t+1}$$

with:

$$\begin{aligned} \Lambda_{t+1} &= M(J,J) \\ V_{t+1} &= (QU)(:,J) \end{aligned}$$

J indexes of m largest eigenvalues of RBR'.

5.3. Constraints

This method deals with the constraints of the problem:

- *Scalability / Processing of huge databases.* The operation which depends on the database size (N) are:
  - QR decomposition of V^u: $O((N-q)m^2)$;
  - Computation of QU: $O(N(m+q)^2)$.

  With $N = 100,000$, $q = m = 50$, this update needs 12s to compute with a Pentium 3GHz.
- *General framework easy to combine with relevance feedback techniques.* Because this method keep semi-definite positive property, any kernel based method can be used;
- *System is dynamic.* The computation of a such as $\sum_{ij}(K^s_{t+1})_{ij} = \sum_{ij}(K^s_t)_{ij}$ allow this update to be repeated indefinitely. The kernel matrix will never diverge to a zero or infinite matrix. Because $a < 1$, all diagonals of kernel matrix will always be lowered at each update. Thus $b > 0$, and E is always added.

6. Experiments

6.1. Feature Distributions

Color and texture information are exploited. $L^*a^*b^*$ space is used for color, and Gabor filters, in twelve different scales and orientations, are used for texture analysis. Both spaces are clustered using an enhanced version of LBG algorithm [15]. We take the same class number for both spaces. Tests have shown that $c = 25$ classes is a good choice for all our feature spaces. Image signature is composed of one vector representing the image color and texture distributions. The input size p is then 50 in our experiments.

6.2. Image Database

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 11 categories[3] (*cf.* Table 1) of different sizes and complexities. The size of these categories varies from 111 to 627 pictures, and the complexity varies from monomodal (low semantic) to highly multimodal (high semantic) classes, relatively to feature vectors. Some of the categories have common images (for instance, castles and mountains of Europe, birds in savannah). For any category search, there is no trivial way to perform a classification between relevant and irrelevant pictures.

6.3. Performance Metric

The CBIR system performances are measured using precision(P), recall(R) and statistics computed on P and R for each category. Let us note A the set of images belonging to the category, and B the set of images returned to the user, then: P $= \frac{|A \cap B|}{|B|}$ and R $= \frac{|A \cap B|}{|A|}$. Usually, the cardinality of B varies from 1 to database size, providing many points (P,R).

We use the average precision P_a 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).

For each evaluation of performances on one of the categories, a robot simulates the usage of the system. Simulation start with 1 relevant image, randomly choosen in one

3 A description of this database and the 11 categories can be found at: http://www-etis.ensea.fr/∼cord/data/mcorel.tar.gz. This archive contains lists of image file names for all the categories.

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

of the 11 categories. The robot labels 10 pictures according to the active learner, during 5 feedback steps. At the end of this search, the training set has 51 labels (0.85% of database size). For each category, 100 retrieval processes are simulated, and as many P_a values are computed. Average precision is then computed on the average of the 100 P_a values.

6.4. Relevance Feedback Method

The relevance feedback method used in these experiments is the RETIN AL method [9]. This method uses a kernel-based classification method (Support Vector Machines) to discriminate between the relevant and irrelevant pictures. It also uses an active learner to choose the best images to label. A comparison of this method to several other relevance feedback methods has proven its efficiency [8].

Semantic kernel matrix K^s is initialized with the dot product between each picture distributions:

$$K^s_{t=0} = D'D$$

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

Next, at each end of a retrieval process, the semantic kernel matrix is updated according to user labels.

6.5. Experiments

The semantic kernel matrix has $m = p = 50$ eigenvalues, in order to initialize it with no approximation. Vigilance parameter ρ is set to 0.1, and joint of irrelevant data parameter γ is set to 0.1.

First experiments are the performances of the system with no semantic learning ((*cf.* Table 1)). Next, system usage is simulated with a semantic kernel update at the end of each retrieval process. Initialization, number of labels per feedback, and number of feedback steps are the same than in the performance evaluation. Thus, $q = 51$ in the proposed algorithm. This simulation is repeated several times starting randomly from any picture from one of the categories. At 1000, 2000, 3000, 4000 and 5000 semantic kernel updates, system performances are computed as described previously, and reported in Table 1.

First, one can notice that the system performances are category dependent. Results on *birds* category are very low in comparison to performances on *doors* category. This can be explained by the capabilities of the low-level features to represent well the semantic categories. For instance, *birds* images have very few common colors and textures, while *doors* images have many common features (horizontal and vertical textures).

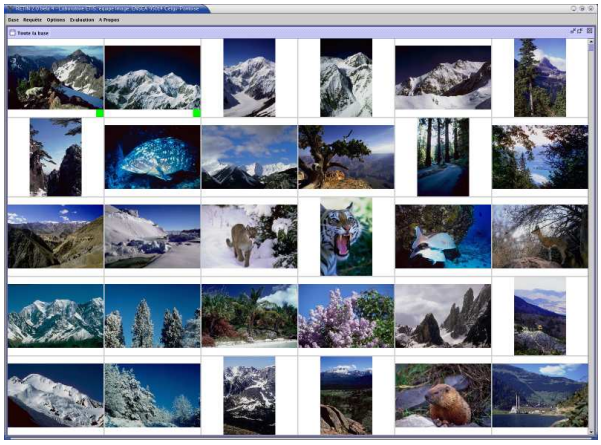

Figure 1. Images 1-30 returned at $t = 0$ (no semantic kernel update) for the mountains category.

Performances converge after a large number of updates. This is explained by the behavior of the active learner. As kernel is updated according to the labels at the end of a retrieval process, and these labels depend on the active learner, then the kernel update also depends on the active learner. The active learner we use in these experiments has not been made to deal with the semantic kernel update scheme. Thus, with a category initially highly clustered according to feature vectors (for instance, *birds* category), this active learner usually does not seek far enough to catch two pictures in two clusters of the same category. However, when it happens, both clusters are joint and performances quickly rise (for instance, *dogs* category between 0 and 1000 updates). For any category, the more the kernel is updated, the higher the performances are.

Statistical evaluation based on P_a gives average results for all the recall values. To better illustrate the efficiency of semantic learning, we also report from Figure 1 to Figure 3 the top precision with or without semantic kernel updates. Figure 1 shows the first images returned by the system with no semantic kernel update. In accordance to the feature distributions, mountain pictures are returned, but also some pictures with a different semantic (fish, for instance). After 5000 uses of the system, the first images returned by the system with the same labels are only mountain pictures (*cf.* Fig. 2). The first non-mountain picture is the 139th image returned by the system (*cf.* Fig. 3).

7. Conclusion

In this paper, a semantic learning approach for image database processing is proposed. The method aims at merg-

Categories			Ave. Precision *vs* semantic kernel updates					
name	size	description	0	1000	2000	3000	4000	5000
birds	219	birds from all around the world	9	14	27	28	31	34
castles	191	modern and middle ages castles	12	16	33	47	50	51
caverns	121	inside caverns	45	61	71	76	75	76
dogs	111	dogs of any species	15	52	72	71	72	73
doors	199	doors of Paris and San Francisco	80	87	92	95	96	95
Europe	627	European cities and countryside	21	26	33	36	40	44
flowers	506	flowers from all around the world	48	62	71	77	81	82
food	315	dishes and fruits	34	47	65	79	81	85
mountains	265	mountains	26	38	46	66	70	78
objects	116	single objects on an uniform background	54	70	74	73	73	78
savannah	408	animals in African savannah	29	33	57	67	73	76

Table 1. Average precision for each category, after 0,1000,2000,3000,4000 and 5000 semantic kernel updates. Simulation protocol for each precision/recall computation: initialization with 1 relevant image, 10 annotations per feedback, 5 feedback steps.

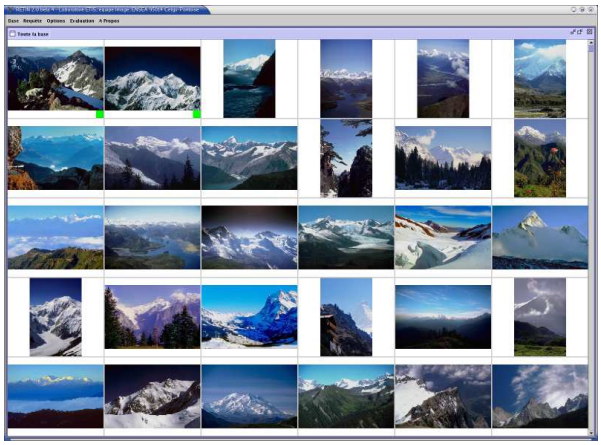

Figure 2. Images 1-30 returned at $t = 5000$ (5000 semantic kernel updates) for the mountains category.

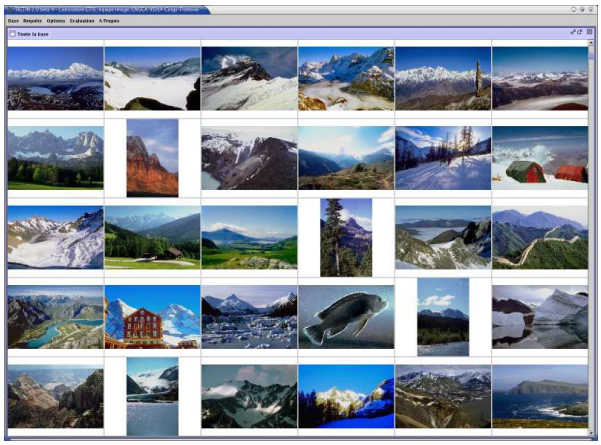

Figure 3. Images 121-150 returned at $t = 5000$ (5000 semantic kernel updates) for the mountains category.

ing all the semantic information based on binary annotations provided by users during retrieval sessions.

All the semantic annotations are used to update the similarity matrix of the database images. We adopted the kernel matrix framework to develop our method. This strong theoretical framework offers nice properties on matrices and efficient combinations with kernel-based techniques for image retrieval classifiers.

To deal with practical implementations, an efficient algebraic scheme has been carried out to compute the kernel matrices with an acceptable computation complexity. We keep under control the required memory space and the algorithm complexity, which is linear to the database size. This last point is decisive for scalability. The proposed method allows the processing of huge databases.

The method has been validated through experiments in an images category retrieval application. The results show the efficiency of the proposed method which may be used as a powerful tool to improve performances.

The framework of semantic kernel building and adaptation is large. We have proposed one application related to on-line image category retrieval, but many applications may be concerned. Actually, any approach dealing with semantic information and metrics may take advantages of such

a kernel framework. Our current investigations aims at using these kernels for browsing and off-line clustering of the database.

Appendix A

QR decomposition of A

This part presents the QR decomposition of the following matrix, according to notations in section 5:

$$A = \begin{pmatrix} Id & TV_t & V_t \end{pmatrix}$$

We denotes by $V^l = V_t(I,:)$ the labeled rows of V_t, and $V^u = V_t(\bar{I},:)$ the unlabeled rows of V_t. According to the previous assumption that $I = \{1, ..., q\}$,

$$V_t = \begin{pmatrix} V^l \\ V^u \end{pmatrix}$$

If $Q^u R^u$ is the QR decomposition of V^u, then:

$$\begin{aligned} A &= \begin{pmatrix} Id & TV & V \end{pmatrix} \\ &= \begin{pmatrix} Id & T^+V^l & V^l \\ 0 & V^u & V^u \end{pmatrix} \\ &= \underbrace{\begin{pmatrix} Id & 0 \\ 0 & Q^u \end{pmatrix}}_{Q} \underbrace{\begin{pmatrix} Id & T^+V^l & V^l \\ 0 & R^u & R^u \end{pmatrix}}_{R} \end{aligned}$$

because we assume that $m < N$, Q is an orthonormal $N \times (q+m)$ matrix, and R is an upper triangular $(q+m) \times (q+2m)$ matrix.

References

[1] G. Caenen, G. Frederix, A.A.M Kuijk, E.J. Pauwels, and B.A.M. Schouten. Show me what you mean! PARISS: A CBIR-interface that learns by example. In *International Conference on Visual Information Systems (Visual'2000)*, volume 1929, pages 257–258, 2000.

[2] E. Chang, B. T. Li, G. Wu, and K.S. Goh. Statistical learning for effective visual information retrieval. In *IEEE International Conference on Image Processing*, Barcelona, September 2003.

[3] I.J. Cox, M.L. Miller, T.P. Minka, T.V. Papathomas, and P.N. Yianilos. The bayesian image retrieval system, PicHunter: Theory, implementation and psychophysical experiments. *IEEE Transactions on Image Processing*, 9(1):20–37, 2000.

[4] N. Cristianini, H. Lodhi, and J. Shawe-Taylor. Latent semantic kernels. *Journal of Intelligent Information Systems (JIIS)*, 18(2), March 2002.

[5] N. Cristianini, J. Show-Taylor, A. Elisseff, and J. Kandola. On kernel target alignement. In *Neural Information Processing Systems (NIPS)*, Vancouver, Canada, December 2001.

[6] J. Fournier and M. Cord. A flexible search-by-similarity algorithm for content-based image retrieval. In *International Conference on Computer Vision, Pattern Recognition and Image Processing (CVPRIP'02)*, Duram, North Carolina, USA, March 2002.

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

[8] P.H. Gosselin and M. Cord. A Comparison of Active Classification Methods for Content-Based Image Retrieval. In *International Workshop on Computer Vision meets Databases (CVDB), ACM Sigmod*, Paris, June 2004.

[9] P.H. Gosselin and M. Cord. RETIN AL: An Active Learning Strategy for Image Category Retrieval. In *IEEE International Conference on Image Processing (ICIP)*, Singapore, October 2004.

[10] J. Han, M. Li, H. Zhang, and L. Guo. A memorization learning model for image retrieval. In *IEEE International Conference on Image Processing (ICIP)*, Barcelona, Spain, September 2003.

[11] D. R. Heisterkamp. Building a latent semantic index of an image database from patterns of relevance feedback. In *International Conference on Pattern Recognition (ICPR)*, Quebec City, Canada, 2002.

[12] Y. Ishikawa, R. Subramanya, and C. Faloutsos. MindReader: Query databases through multiple examples. In *24th VLDB Conference*, pages 218–227, New York, 1998.

[13] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semi-definite programming. In *International Conference on Machine Learning (ICML)*, Sydney, Australia, 2002.

[14] H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun. Long-term learning from user behavior in content-based image retrieval. Technical report, Computer Vision Group, Computing Centre, University of Geneva, rue Général Dufour, 24, CH-1211 Genève, Switzerland, 2000.

[15] G. Patanè and M. Russo. The enhanced LBG algorithm. *IEEE Transactions on Neural Networks*, 14(9):1219–1237, November 2001.

[16] R. Picard. A society of models for video and image libraries. *IBM Systems Journal*, 35(3/4):292–312, 1996.

[17] Y. Rui and T.S. Huang. Optimizing learning in image retrieval. In *Conf on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 236–243, Hilton Head, SC, June 2000.

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

[19] N. Vasconcelos and M. Kunt. Content-based retrieval from image databases: current solutions and future directions. In *International Conference in Image Processing (ICIP'01)*, volume 3, pages 6–9, Thessaloniki, Greece, October 2001.

[20] R.C. Veltkamp. Content-based image retrieval system: A survey. Technical report, University of Utrecht, 2002.