Neural Comput & Applic (2004) 13: 130–139
DOI 10.1007/s00521-004-0415-2

ORIGINAL ARTICLE

Iker Gondra · Douglas R. Heisterkamp · Jing Peng

Improving image retrieval performance by inter-query learning with one-class support vector machines

Received: 7 April 2004 / Accepted: 20 April 2004 / Published online: 26 May 2004

Abstract Relevance feedback (RF) is an iterative process which improves the performance of content-based image retrieval by modifying the query and similarity metric based on the user's feedback on the retrieval results. This short-term learning within a single query session is called intra-query learning. However, the interaction history of previous users over all past queries may also be potentially exploited to help improve the retrieval performance for the current query. The long-term learning accumulated over the course of many query sessions is called inter-query learning. We present a novel RF framework that learns one-class support vector machines (1SVM) from retrieval experience to represent the set memberships of users' high-level concepts and stores them in a "concept database". The "concept database" provides a mechanism for accumulating inter-query learning obtained from previous queries. By doing a fuzzy classification of a query into the regions of support represented by the 1SVMs, past experience is merged with current intra-query learning. The geometric view of 1SVM allows a straightforward interpretation of the density of past interaction in a local area of the feature space and thus allows the decision of exploiting past information only if enough past exploration of the local area has occurred. The proposed approach is evaluated on real data sets and compared against both traditional intra-query-learning-only RF approaches and other methods that also exploit inter-query learning.

1 Introduction

The rapid development of information technologies and the advent of the World-Wide Web have resulted in a tremendous increase in the amount of available multimedia information. As a result, there is a need for effective mechanisms to search large collections of multimedia data, especially images. In traditional image retrieval, keywords are manually assigned to images and, for any particular query, images with matching keywords are retrieved [15]. However, it is usually the case that all the information contained in an image cannot be captured by a few keywords. Furthermore, a large amount of effort is needed to do keyword assignments in a large image database and, because different people may have different interpretations of image contents, there will be inconsistencies [15].

In order to alleviate some of these problems, content-based image retrieval (CBIR) was proposed. Some early systems include those described in [7] and [12]. A CBIR system extracts some features (such as color, shape, and texture) from an image. The features are then the components of a feature vector which makes the image correspond to a point in an input space. In order to determine closeness between two images, a similarity measure is used to calculate the distance between their corresponding feature vectors. However, because of the gap between high-level concepts and low-level features and the subjectivity of human perception, the performance of CBIR systems is not satisfactory [15].

Relevance feedback (RF) attempts to overcome these problems by gathering semantic information from user interaction. In order to learn a user's query concept, the user labels each image returned in the previous query round as relevant or not relevant. Based on the feedback, the retrieval scheme is adjusted and the next set of images is presented to the user for labeling. This process iterates until the user is satisfied with the retrieved images or stops searching. Many approaches for improving the performance of RF have been proposed

I. Gondra (✉) · D. R. Heisterkamp
Department of Computer Science,
Oklahoma State University, Stillwater,
Oklahoma, USA
E-mail: gondra@cs.okstate.edu
Tel.: + 1-405-7449552
Fax: + 1-405-7449097

J. Peng
Department of Electrical Engineering and Computer Science,
Tulane University, New Orleans, Louisiana, USA

[13, 14, 16]. In [14] a probabilistic feature relevance learning (PFRL) method that automatically captures feature relevance based on user's feedback is presented. It computes flexible retrieval metrics for producing neighborhoods that are elongated along less relevant feature dimensions and constricted along most influential ones. This technique has shown promise in a number of image database applications. Recently, support vector machines (SVM) have been applied to CBIR systems with RF to significantly improve retrieval performance [3, 9, 19].

We can distinguish two different types of information provided by RF. The short-term learning obtained within a single query session is *intra-query* learning. The long-term learning accumulated over the course of many query sessions is *inter-query* learning. By accumulating knowledge from users, inter-query learning aims at enhancing future retrieval performance. Thus, both short and long-term learning are useful in CBIR. However, in most current systems, all prior experience from past queries is lost. That is, the system only takes into account the current query session without using any long-term learning.

In this paper, we propose a novel RF approach that uses one-class SVM (1SVM) to capture users' high-level concepts and utilizes them as previous experience to be used in future queries. For each query, the set membership of the user's high level concept is learned by training a 1SVM. First, the user-labeled relevant images are mapped into a nonlinearly transformed kernel-induced feature space. Then, risk minimization is performed by attempting to include most of those images into a hypersphere (i.e., 1SVM) of minimum size. The 1SVM is then stored in a "concept database", which is a repository of inter-query learning and is used to improve the retrieval performance on future queries. The use of kernels allows the 1SVM to deal with the non-linearity of the distribution of training images in an efficient manner, while at the same time, providing good generalization. In addition, the geometric view of 1SVM allows a straightforward interpretation of the density of past interaction in a local area of the feature space and thus allows the decision of exploiting past information only if enough past exploration of the local area has occurred.

The rest of this paper is organized as follows. Section 2 gives a brief introduction to SVMs and describes 1SVMs in detail. A description of our proposed approach for exploiting both intra and inter-query learning is presented in Sect. 3. In Sect. 4, we report experimental results which confirm the effectiveness of our method. Concluding remarks are given in Sect. 5.

1.1 Previous work

A few approaches [8, 11, 20] attempt inter-query learning (i.e., RF from past queries are used to improve the retrieval performance of the current query). The initial results from those approaches for inter-query learning show a tremendous benefit in the initial and first iteration of retrieval. Inter-query learning thus offers a great potential for reducing the amount of user interaction by reducing the number of iterations needed to satisfy a query.

In [8] latent semantic analysis (LSI) [6] was used to provide a generalization of past experience. LSI is an important technique in information retrieval. It uses the context (document) of a word usage to uncover its hidden (i.e., latent) semantics. LSI creates a semantic space by performing a singular value decomposition on a term-by-document matrix. In [8], the images in a database are viewed as the fundamental vocabulary of the system. The RF from each query is considered as a document composed of many terms (images). Thus, assuming that the terms of a document have a latent semantic relationship, it is possible to use LSI to capture inter-query learning.

Both [11] and [20] take the approach of complete memorization of prior history. In [11] the correlation between past image labeling is merged with low-level features to rank images for retrieval. The model estimates the semantic correlation between two images based on their co-occurrence frequency (i.e., the number of query sessions in which both images were labeled relevant). Intuitively, the larger the co-occurrence frequency of two images is, the more likely that they are semantically similar. Given a query $\mathbf{x}\in\Re^d$ (i.e., the feature vector of a query image), the semantic similarity to each image is initialized to its feature-based similarity. Then, semantic similarities are iteratively updated based on correlation with top-ranked images. Thus, images having strong correlations with the top-ranked images are likely to have a high semantic similarity with $\mathbf{x}$, even if their feature-based similarity is low [11].

In [20] the extra inter-query information is efficiently encoded by adding a virtual feature (VF) to the feature vector of an image. Initially, the VF of each image is empty. Given a query $\mathbf{x}$, the k nearest neighbor images to it are retrieved and the user labels each of them as relevant or not relevant. Then, a number from a system counter is concatenated to the VFs of all user-labeled relevant images to indicate that they deliver the same concept as $\mathbf{x}$. To determine relevance between $\mathbf{x}$ and database images, the VF of $\mathbf{x}$ is computed as the concatenation of the VFs of all user-labeled relevant images in the previous RF iteration. The VFs of $\mathbf{x}$ and the database images are then used in a probabilistic dissimilarity measure that dynamically adjusts the distance between $\mathbf{x}$ and the database images [20]. One of the shortcomings of this method is that it needs at least one RF iteration and thus inter-query learning cannot be used to improve the precision of the initial retrieval set.

2 Support vector machines

A support vector machine is a system for efficiently training linear learning machines in a kernel-induced feature space while at the same time respecting the insights provided by generalization theory and exploiting optimization theory [4]. The objective of support vector classification is to create a computationally efficient method of learning "good" separating hyperplanes in a high dimensional feature space, where "good" corresponds to optimizing the generalization bounds given by generalization theory [4].

Suppose we are given training data as a set of n observations. Each observation is a pair $\{\mathbf{x_i}, y_i\}$ where $\mathbf{x_i} \in \Re^d$ and $y_i \in \{-1, 1\}$ is the corresponding class label. Now suppose that we have a learning machine whose task is to learn the mapping $\mathbf{x_i} \rightarrow y_i$. The machine is defined by a set of possible mappings $\mathbf{x} \rightarrow f(\mathbf{x}, \alpha)$ where the functions $\mathbf{x} \rightarrow f(\mathbf{x}, \alpha)$ are labeled by the adjustable parameters α. Thus, a learning machine is a family of functions $f(\mathbf{x}, \alpha)$ and a particular choice of α results in a "trained machine" [2]. If there are no restrictions on the family of functions $f(\mathbf{x}, \alpha)$, the trained machine may not generalize well on unseen data (even when there is no error in the training data). This problem is known as overfitting and it drove the initial development of SVMs [2]. Statistical learning theory, or VC (Vapnik–Chervonenkis) theory, shows that the best generalization performance can be obtained when the capacity of the learning machine is restricted to one that is suitable to the amount of available training data [2].

Suppose we have a learning machine that is a class of separating hyperplanes $\langle \mathbf{x} \cdot \mathbf{w} \rangle + \mathbf{b} = 0$, where $\mathbf{w} \in \Re^d$ and $\mathbf{b} \in \Re$, corresponding to decision functions $f(\mathbf{x}) = \text{sign}(\langle \mathbf{x} \cdot \mathbf{w} \rangle + \mathbf{b})$. It can be shown that the optimal hyperplane (i.e., the one that minimizes the generalization error or the bound on the actual risk) corresponds to the one with maximal margin of separation between the two classes [2]. The optimal hyperplane has the smallest capacity (also known as the lowest VC dimension). In order to find the optimal separating hyperplane, a constrained quadratic optimization problem is solved. The solution has an expansion: $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \mathbf{x_i}$. Those points for which $\alpha_i > 0$ are called support vectors and lie closest to the hyperplane (see Fig. 1). All other points have $\alpha_i = 0$ thus the support vectors are the critical elements of the training set [2]. The final decision function is of the form: $f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i \langle \mathbf{x} \cdot \mathbf{x_i} \rangle\right)$.

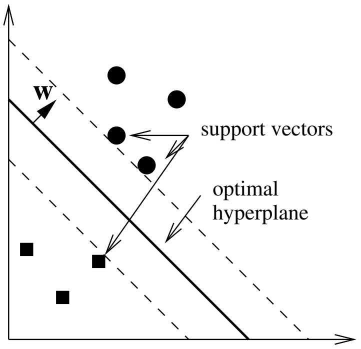

Fig. 1 A simple linear SVM

In order to generalize to the case where the decision function is not linearly separable, SVMs first map the data into some other (possibly infinite dimensional) feature space F using a mapping $\Phi: \Re^d \rightarrow \Re^D$, with $D \geq d$ (see Fig. 2).

Both the quadratic optimization problem and the final decision function depend on the data through dot products in F (i.e, on functions of the form $\langle \Phi\ (\mathbf{x_i}) \cdot \Phi\ (\mathbf{x_j}) \rangle$). Therefore, we can use a kernel to avoid having to perform an explicit mapping into the feature space. A kernel calculates the dot product in the feature space of the image of two points from input space, $K\ (\mathbf{x_i}, \mathbf{x_j}) = \langle \Phi(\mathbf{x_i}) \cdot \Phi(\mathbf{x_j}) \rangle$. Table 1 shows some commonly used kernel functions. Distance in the feature space can be calculated by means of the kernel function [4]. Given $\mathbf{x_i}$ and $\mathbf{x_j}$ in input space, the corresponding distance in feature space is:

$$\begin{aligned} \text{dist}_F(\mathbf{x_i}, \mathbf{x_j})^2 &= ||\Phi(\mathbf{x_i}) - \Phi(\mathbf{x_j})||^2 \\ &= K(\mathbf{x_i}, \mathbf{x_i}) - 2K(\mathbf{x_i}, \mathbf{x_j}) + K(\mathbf{x_j}, \mathbf{x_j}) \end{aligned}$$

This is known as the kernel trick and it allows SVMs to implicitly project the original training data to a higher dimensional feature space.

2.1 One-class SVM

In a one-class classification problem, data from only one of the classes (the target class) are available. For instance, user-labeled relevant images give us information about the user's high-level concept. Thus, in one-class classification, the task is to create a boundary around the target class such that most of the target data are included while, at the same time, minimizing the risk of accepting outliers (i.e., data that do not belong to the target class) [18].

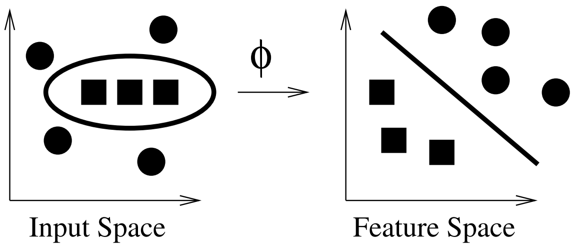

Fig. 2 An SVM maps the training data nonlinearly into a higher dimensional feature space via Φ. By the use of a kernel function, the optimal separating hyperplane can be computed without explicitly carrying out the map into the feature space

Table 1 Common kernels

Kernel	Formula
Linear	$K\ (\mathbf{x_i}, \mathbf{x_j}) = \langle \mathbf{x_i} \cdot \mathbf{x_j} \rangle$
Polynomial	$K\ (\mathbf{x_i}, \mathbf{x_j}) = (\langle \mathbf{x_i} \cdot \mathbf{x_j} \rangle + 1)^n$
Gaussian	$K(\mathbf{x_i}, \mathbf{x_j}) = e^{-\frac{\lVert \mathbf{x_i} - \mathbf{x_j} \rVert^2}{\sigma^2}}$

The strategy that is followed in a 1SVM consists of mapping the training data to a higher dimensional feature space and then attempting to include most of them into a hypersphere of minimum size. We show how the computation of the 1SVM from the user-labeled relevant images is performed. Consider training data $\{\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_n}\}$ where $\mathbf{x_i} \in \Re^d$ is the feature vector of the ith user-labeled relevant image in the retrieval set. Let $\Phi: \Re^d \rightarrow \Re^D$ be a non-linear mapping from the original (d dimensional) input space to the (D dimensional) feature space with $D \geq d$. The strategy is to map the training data to the higher dimensional feature space and include most of them into a hypersphere of minimum size. That is, the task is to minimize the following objective function (in primal form):

$$\min_{R \in \Re, \zeta \in \Re^n, \mathbf{a} \in \Re^D} R^2 + C \sum_{i=1}^{n} \zeta_i$$

with constraints that (almost) all the training data are within the hypersphere:

$$||\Phi(\mathbf{x_i}) - \mathbf{a}||^2 \leq R^2 + \zeta_{\mathrm{i}}, \quad \zeta_{\mathrm{i}} \geq 0, \quad i = 1, 2, \ldots, n$$

where R and $\mathbf{a}$ are the radius and center of the hypersphere, respectively (see Fig. 3). The parameter $0 \leq C \leq 1$ is the soft–hard margin penalty and it gives the trade-off between the size of the hypersphere and the number of training data that can be included. By setting partial derivatives to zero in the corresponding Lagrangian we obtain the following expression for $\mathbf{a}$:

$$\mathbf{a} = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x_i})$$

Replacing partial derivatives in the Lagrangian and noticing that $\mathbf{a}$ is a linear combination of the training data (which allows us to use a kernel function), we obtain the following objective function (in dual form)

$$\min_{\alpha} \sum_{i=j=1}^{n} \alpha_i \alpha_j K(\mathbf{x_i}, \mathbf{x_j}) - \sum_{i=j=1}^{n} \alpha_i K(\mathbf{x_i}, \mathbf{x_i})$$

with constraints:

$$0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{n} \alpha_i = 1$$

where K is an appropriate Mercer kernel. We use the Gaussian kernel (see Table 1). A quadratic programming method is used to find the optimal α values in the objective function [18]. Given $\mathbf{x}$ in input space and hypersphere center $\mathbf{a}$, their distance in feature space is:

$$\begin{aligned}\mathrm{dist}_F(\mathbf{x}, \mathbf{a}) &= ||\Phi(\mathbf{x}) - \mathbf{a}||^2 \\ &= K(\mathbf{x}, \mathbf{x}) - 2\sum_{i=1}^{n} \alpha_i K(\mathbf{x}, \mathbf{x_i}) + \sum_{i=j=1}^{n} \alpha_i \alpha_j K(\mathbf{x_i}, \mathbf{x_j})\end{aligned}$$

Also, $\mathbf{x}$ falls inside the hypersphere when this distance is smaller than or equal to the radius:

$$\mathrm{dist}_F(\mathbf{x}, \mathbf{a}) \leq R^2$$

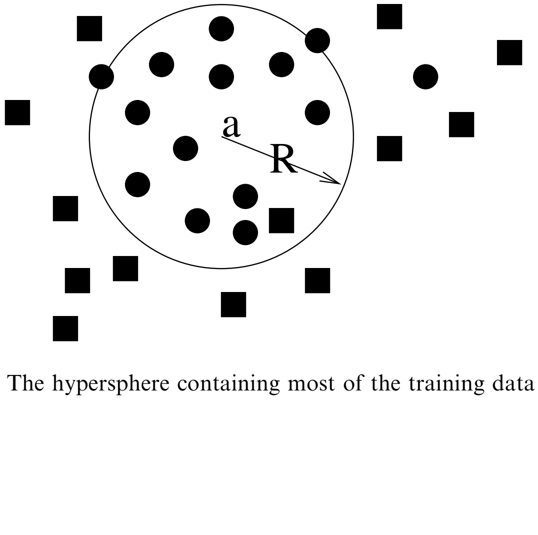

Fig. 3 The hypersphere containing most of the training data

3 Proposed approach

The approach that is used for selecting the set of images that is presented to the user (i.e., the retrieval set) is based on exploiting both intra and inter-query learning. Figure 4 shows a diagram of the proposed method. At the first stage, the user submits a query image $\mathbf{x}$ to the system. Let k be the size of the retrieval set and $0 \leq w_{\mathrm{intra}} \leq 1$ be the intra-query learning weight. The intra-query learning is exploited by including $(w_{\mathrm{intra}})k$ nearest neighbor images to $\mathbf{x}$ in feature space into the retrieval set. The remaining $(1-w_{\mathrm{intra}})k$ images in the retrieval set are obtained by exploiting the accumulated inter-query learning in the "concept database". Thus, the ratio of intra to inter-query learning that is used in processing a query is w_{intra}:$(1-w_{\mathrm{intra}})$. We now explain how the remaining $(1-w_{\mathrm{intra}})k$ "inter-query learning" images are selected.

Using 1SVMs we obtain set membership knowledge (about previous users' high-level concepts), which can be visualized as hyperspheres in feature space. In order to integrate this prior experience with the current query $\mathbf{x}$, a fuzzy classification of $\mathbf{x}$ into the existing regions of support (i.e., 1SVMs) is performed. Thus, the "concept database" is searched and it is determined whether $\mathbf{x}$ falls into any of the accumulated 1SVMs. Because it is very common for an image to be ascribed into many different concepts, we expect to have queries that fall into many hyperspheres. One possible way of exploiting inter-query learning would be to perform a hard classification by selecting $(1-w_{\mathrm{intra}})k$ nearest neighbor images to the closest hypersphere's center (i.e., closest prototype). However, this is not a very good strategy since a query may be a member of several concept sets (i.e., it may fall into many hyperspheres). Thus, it may as well be ascribed to the concept corresponding to any one of the other 1SVMs. Furthermore, a query may be ascribed to a combination of different concepts.

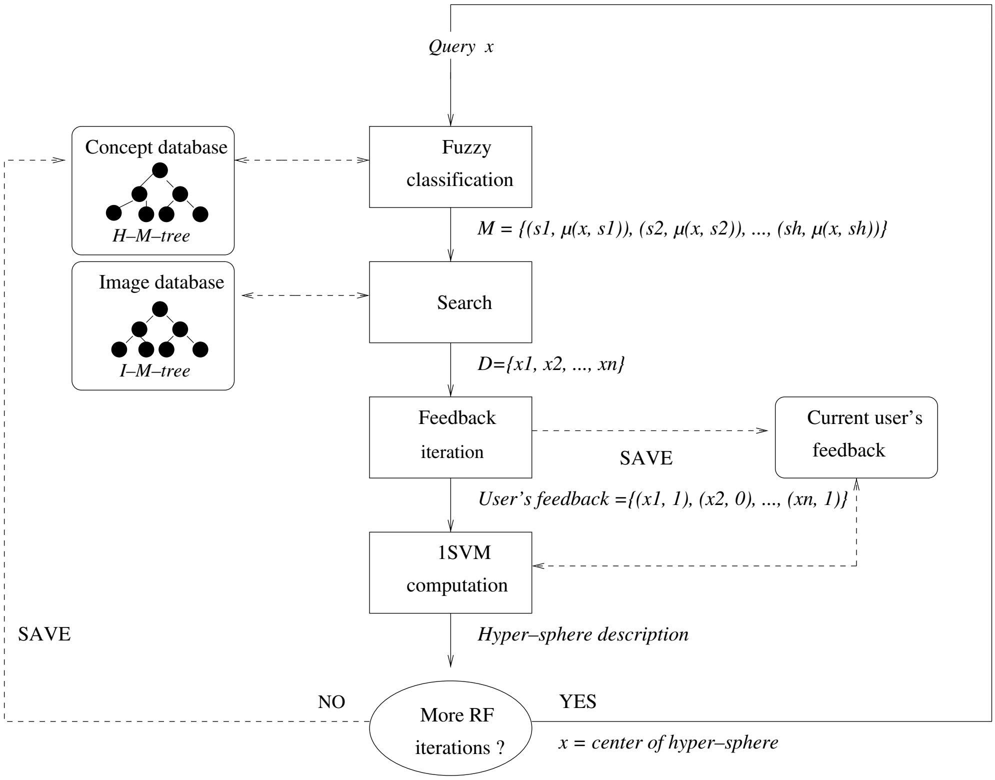

Fig. 4 System diagram

The results of experiments conducted in [1] for learning users' text preferences suggest that, for simple queries (i.e., queries that can be ascribed to one concept), a purely exploitative strategy delivers good performance. However, for complex queries (i.e., queries that can be ascribed to more than one concept), there is a trade-off between faster learning of the user's query concept and the delivery of more relevant documents. Therefore, instead, we use the ideas from possibilistic cluster analysis [10] and assign a degree of membership to each one of the 1SVMs (i.e., to each cluster) according to the degree by which **x** can be ascribed to its particular concept.

Given a set of points, the fuzzy c-means algorithm searches for an optimal set of clusters. The clusters are represented by their corresponding centers and each point has a degree of membership in each cluster which models the degree of the point belonging to the cluster [10]. In our case, the set of clusters (in the form of 1SVMs) is formed by the historical interaction of users with the system. We then use the following membership function to assign degrees of membership to the h hyperspheres into which **x** falls:

$$\mu(\mathbf{x}, s_i) = \frac{1}{\sum_{j=1}^{h} \frac{\mathrm{dist}_F(\mathbf{x},\mathbf{a_i})}{\mathrm{dist}_F(\mathbf{x},\mathbf{a_j})}}, \quad i = 1, 2, \ldots, h$$

where s_i is the ith hypersphere with corresponding center $\mathbf{a_i}$. Therefore, the degree of membership of **x** into a 1SVM is based on the relative distances between **x** and the centers of all hyperspheres into which **x** falls. Suppose **x** falls into h hyperspheres. If c_i denotes the concept that is embodied by the ith hypersphere then the belief (or our degree of confidence) that **x** is delivering concept c_i is equal to $\mu(\mathbf{x}, s_i)$.

To form the retrieval set, sample representative images from each hypersphere into which **x** falls are included. The number of representatives that a particular concept c_i has in the retrieval set is proportional to $\mu(\mathbf{x}, s_i)$. Thus, if $N(c_i)$ denotes the number of images of concept c_i that appear in the retrieval set then $N(c_i) < N(c_j)$ whenever $\mu(\mathbf{x}, s_i) < \mu(\mathbf{x}, s_j)$. Because **x** may fall into many hyperspheres but only $(1-w_{\text{intra}})k$ "inter-query-learning" images are to be included in the retrieval set, priority is given to hyperspheres with higher $\mu(\mathbf{x}, s_i)$. Thus, after $(1-w_{\text{intra}})k$ images are selected, the remaining hyperspheres with smaller $\mu(\mathbf{x}, s_i)$ are ignored.

The retrieval set is thus formed by exploiting both intra and inter-query learning. Then, the user evaluates the relevance of images in the retrieval set. The user-labeled relevant images are used as training data for a 1SVM. The center of the resulting hypersphere becomes the new query for the second round of RF and this process continues until the user is satisfied with the results or quits. When the session is over, all the RF provided by the user is used as training data for the 1SVM that is stored in the "concept database". Our algorithm is summarized below.

```
$T \leftarrow \emptyset$
```

```
1. Let $\mathbf{x}$ be the current query
2. Search the concept database; find hyperspheres into which $\mathbf{x}$ falls
   $S \leftarrow \{s_1, s_2,...,s_h\}$
3. Do a fuzzy classification of $\mathbf{x}$ into $S$
   $M \leftarrow \{\mu(\mathbf{x},s_1),\mu(\mathbf{x},s_2),...,\mu(\mathbf{x},s_h)\}$
4. Form the retrieval set $D$
   $D \leftarrow \{(w_{\mathrm{intra}})k$ nearest neighbor images to $\mathbf{x}\}$
   $D \leftarrow D \cup \{(1-w_{\mathrm{intra}})k$ images based on $M\}$
5. User marks images in $D$ as relevant or non-relevant
   $T \leftarrow T \cup \{$user-labeled relevant images$\}$
   Use $T$ to compute 1SVM
6. While more RF iterations Do
   $\mathbf{x} \leftarrow$ center of resulting 1SVM
   go to 2
7. Insert 1SVM into concept database
```

The M-tree [5] data structure is used for efficient searching of nearest-neighbor images in feature space. An M-tree is a paged, balanced, and dynamic tree. It provides an efficient platform for the execution of multi-dimensional similarity queries using an arbitrary metric. For details, see [5]. We use M-trees for efficient searching of both historical information and images in the database. The *image* M-tree (I-M-tree) contains all the images in the database and the *history* M-tree (H-M-tree) contains the learned 1SVMs (i.e., the historical hyperspheres).

4 Experimental results

We have implemented the VF approach [20], the statistical correlation technique (SC) [11], and our proposed method (1SVM). Those three approaches exploit inter-query learning. Their response with respect to different amounts of experience (data level) is investigated. We also compare the performance of our method against that of traditional intra-query-learning-only RF approaches. For that purpose, we have also implemented the PFRL [14] method. The retrieval performance is measured by *precision*. *Precision* measures the ability to retrieve only relevant images and is defined as

$$precision = \frac{\text{Number of relevant images retrieved}}{\text{Number of images retrieved}}$$

The following data sets were used for evaluation:

Texture The texture data set, obtained from MIT Media Lab at ftp://whitechapel.media.mit.edu/pub/VisTex. There are 40 different texture images that are manually classified into 15 classes. Each of those images is then cut into 16 non-overlapping images of size 128×128. Thus, there are 640 images in the database. The images are represented by 16-dimensional feature vectors. We use 16 Gabor filters (two scales and four orientations). Sample images are shown in Fig. 5.

Letter The letter data set consists of 20,000 character images, each represented by a 16-dimensional feature vector. There are 26 classes of the two capital letters O and Q. The images are based on 20 different fonts with randomly distorted letters.

To determine the free parameters, a tenfold cross-validation was performed for the *Texture* and *Letter* data sets. Each data set was divided into ten partitions. Each partition in turn was left out and the other nine were used to determine values for the free parameters. The left out partition was then used to test the algorithm. The values reported are the average of the ten tests. We make no claim on using optimal values for *Letter* as the parameters were selected after a very coarse sampling.

Figures 6 and 7 show the precision in the initial retrieval set (i.e., with no iterations of RF) with respect to different data levels. The data level is the amount of accumulated inter-query learning (i.e., number of queries processed) relative to the number of images in the data set. In order to create the initial retrieval set, a traditional intra-query-learning-only RF approach

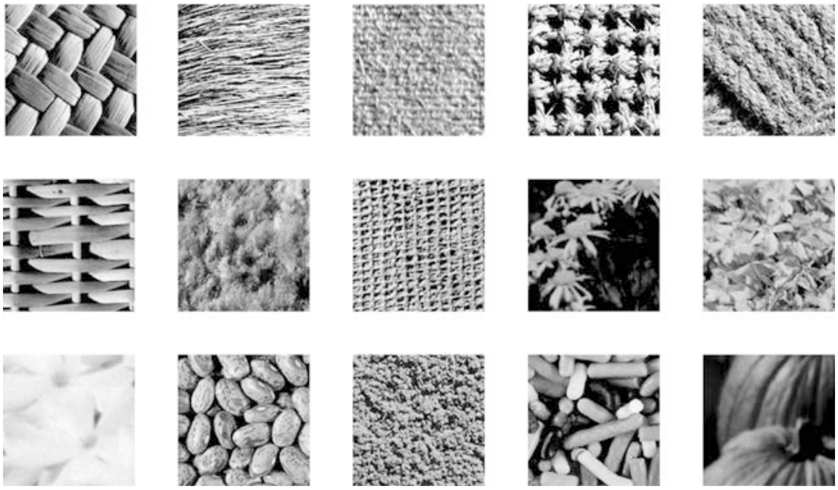

Fig. 5 Sample Images from MIT Texture database

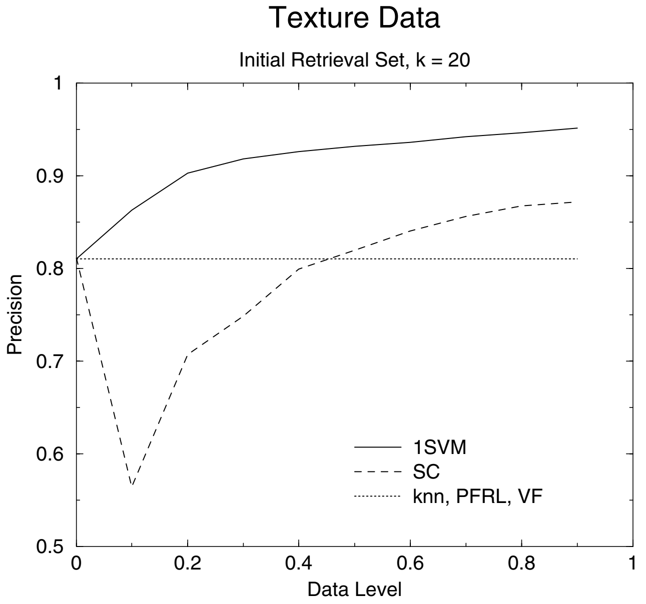

Fig. 6 Precision vs. data level in initial retrieval set. Texture Data

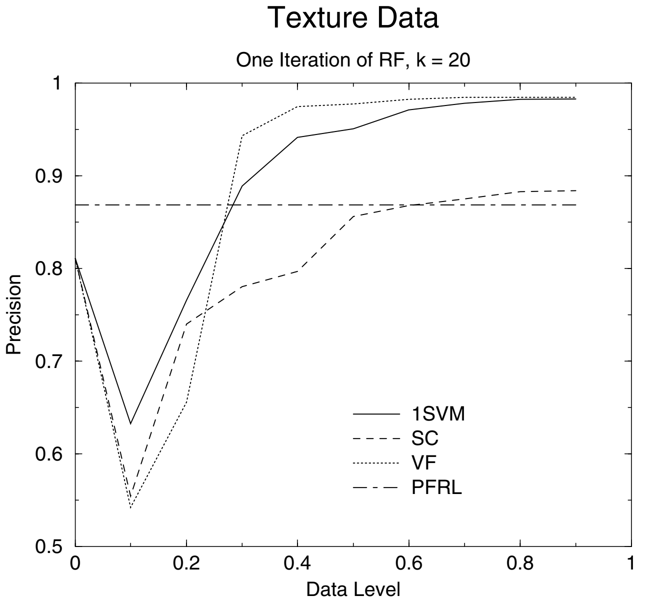

Fig. 8 Precision vs. data level with one RF iteration. Texture Data

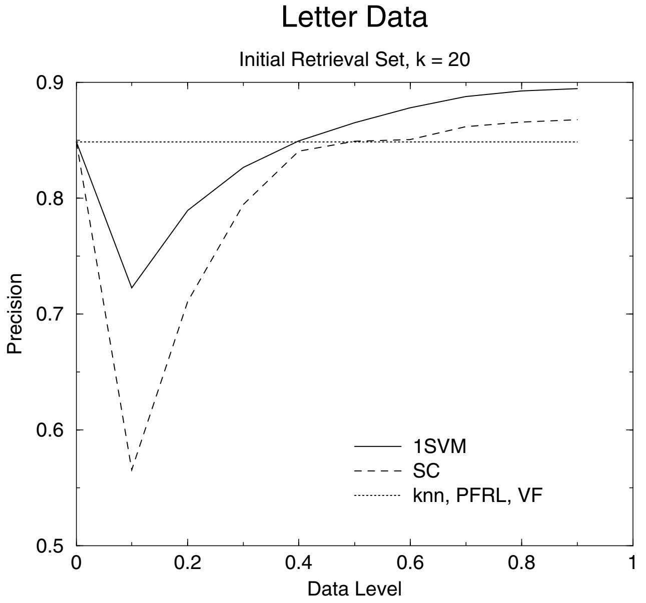

Fig. 7 Precision vs. data level in initial retrieval set. Letter Data

performs a k-nearest-neighbor (knn) search. Both VF [20] and PFRL [14] require at least one iteration of RF. Thus, for the initial retrieval set, they have the same performance as knn. As we can observe from those figures, precision in the initial retrieval set can be drastically improved by integrating inter-query learning. Also, precision keeps improving as the data level increases. This results in reduction of the number of RF iterations needed to satisfy a query. Thus, from the user's point of view, it is very beneficial, because users cannot stand too many RF iterations. On the other hand, if we use solely a knn search, there is no gain on the initial retrieval precision along the number of processed queries. From those figures we can observe that, with low data levels, there may be an initial decrease in precision. In our method, this is because the retrieval set is formed based on a fixed ratio of intra to inter-query learning. Both VF [20] and SC [11] use a similar concept, the "maximal distance adjustment" and the "semantic weight" respectively, which is also based on a fixed weighting of inter-query learning. Intuitively, initially we would like to rely heavily on current intra-query learning since, at the beginning, there is not much historical information. Similarly, we would like to increase the exploitation of inter-query learning as more queries are processed and experience accumulates. Thus, we could adaptively change the ratio of intra to inter-query learning so that at the beginning, when there is little historical information, w_{intra} is large and, as experience accumulates, it becomes increasingly smaller (i.e., we rely more on inter-query learning). In our method, the optimal ratio of intra to inter-query learning was defined as that resulting in highest precision with large data levels and was determined to be 0.25:0.75. Note that choosing 1:0 as the ratio of intra to inter-query learning is the same as using an intra-query-learning-only 1SVM approach. Thus, our method outperforms 1SVM approaches that do not exploit inter-query learning.

Figures 8 and 9 show the precision after one iteration of RF with respect to different data levels. As we can observe from those figures, precision increases after one RF iteration. The amount of improvement obtained when going from one to two RF iterations is much smaller. This is a desired property since users do not want to perform many RF iterations. We can also observe that, with at least one RF iteration, 1SVM and VF [20] have similar performance. On the other hand, our approach can provide improvement in the initial retrieval set. It can also be seen that, as the data level

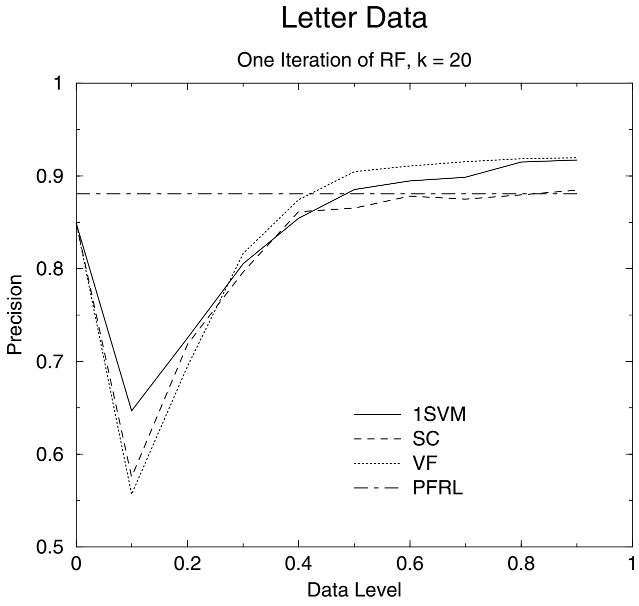

Fig. 9 Precision vs. data level with one RF iteration. Letter Data

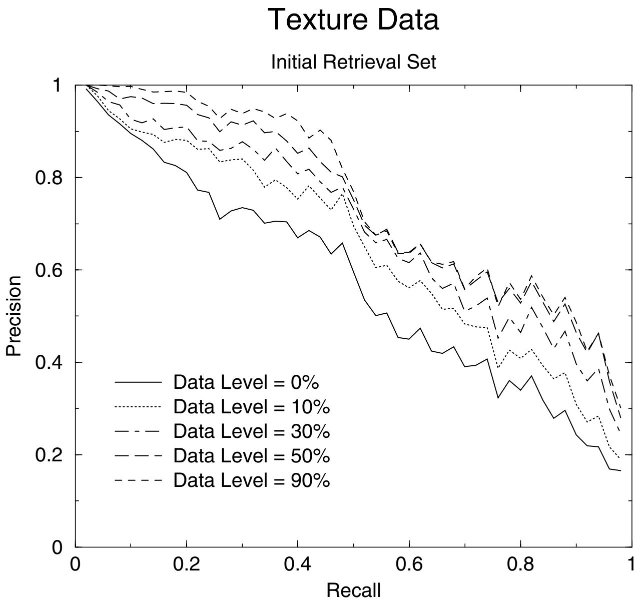

Fig. 10 *Precision* vs. *recall* at different data levels. Texture Data

increases, both methods result in a very significant performance improvement over PFRL [14]. On the other hand, with PFRL [14], there is no gain in retrieval precision along the number of processed queries. As a result, the precision stays at a fixed value. This demonstrates that methods which exploit both short and long-term information perform better than intra-query-learning-only techniques.

We also use *recall* to investigate the performance of our method against different amounts of experience. *Recall* measures the ability to retrieve all relevant images in the database and is defined as:

$$recall = \frac{\text{Number of relevant images retrieved}}{\text{Number of relevant images in database}}$$

Figure 10 shows the *precision–recall* graph of our proposed method for different data levels. Both high recall and high precision are desired, though not often obtainable. The values are the average over 64 random queries from *Texture*. From that figure we can observe that increasing the data level has the desirable effect of pulling the *precision–recall* curve towards the upper right.

As a last illustration, Figure 11 shows a particular retrieval result obtained by performing a nearest-neighbor search in feature space on random query from the *Texture* data set. A retrieval precision of 0.25 is achieved. This shows the inconsistency between content-based and semantic similarity. In contrast, Figure 12 shows the retrieval results obtained with our method. In this case, a retrieval precision of 0.95 is achieved. This illustrates that exploiting inter-query learning can dramatically help to reduce the semantic gap and improve retrieval performance.

We can learn from these results that the image retrieval performance is constantly improved by integration of inter-query learning. Furthermore, performance can be improved in the initial retrieval set where a traditional intra-query-learning-only approach would require at least one iteration of RF to provide some improvement. Thus, user interaction can be reduced by reducing the number of iterations that are needed to satisfy a query.

5 Conclusions and future work

This paper presented a new approach for incorporating inter-query learning into an RF system to improve image retrieval performance. By learning 1SVM from retrieval experience we can represent the set memberships of users' high level concepts and store them in a "concept database". The "concept database" provides a mechanism for accumulating inter-query learning obtained from previous queries. By doing a fuzzy classification of a query into the regions of support represented by the 1SVMs, past experience is merged with current intra-query learning. The geometric view of 1SVM allows a straightforward interpretation of the density of past interaction in a local area of the feature space and thus allows the decision of exploiting past information only if enough past exploration of the local area has occurred. Experimental results on real data sets demonstrate the effectiveness of our proposed method. To extend our research, some challenging problems need to be further investigated:

- How to combine or merge hyperspheres (i.e., user's high level concepts)? Currently, inter-query learning is represented by a constantly growing number of (possibly overlapping) 1SVMs (i.e., regions) in the feature space. Thus, clustering and summarizing

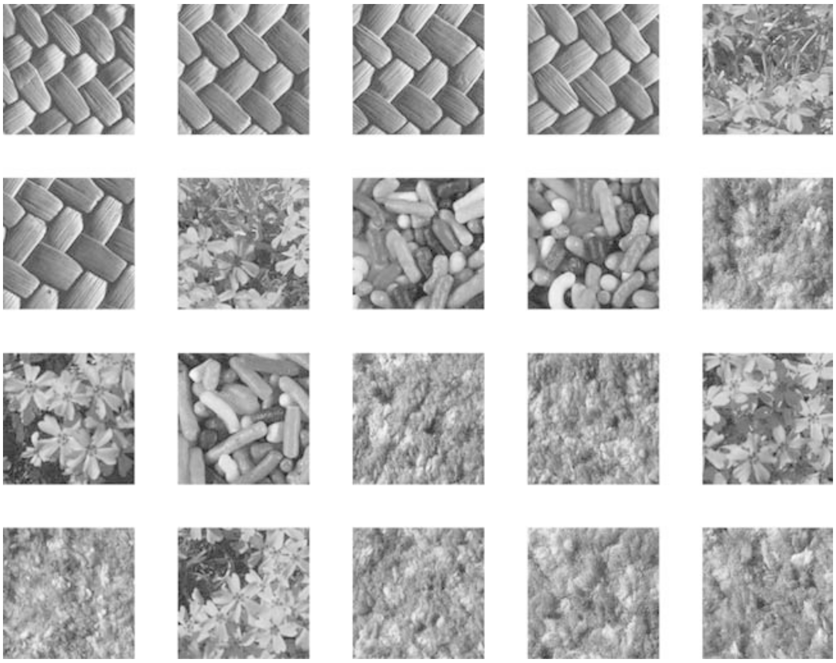

Fig. 11 Retrieval results after performing a nearest-neighbor search in feature space on a random query from the Texture data set. The *top leftmost image* represents the query image. The images are sorted based on their similarity to the query. The ranks descend from left to right and from top to bottom. Retrieval precision is 0.25

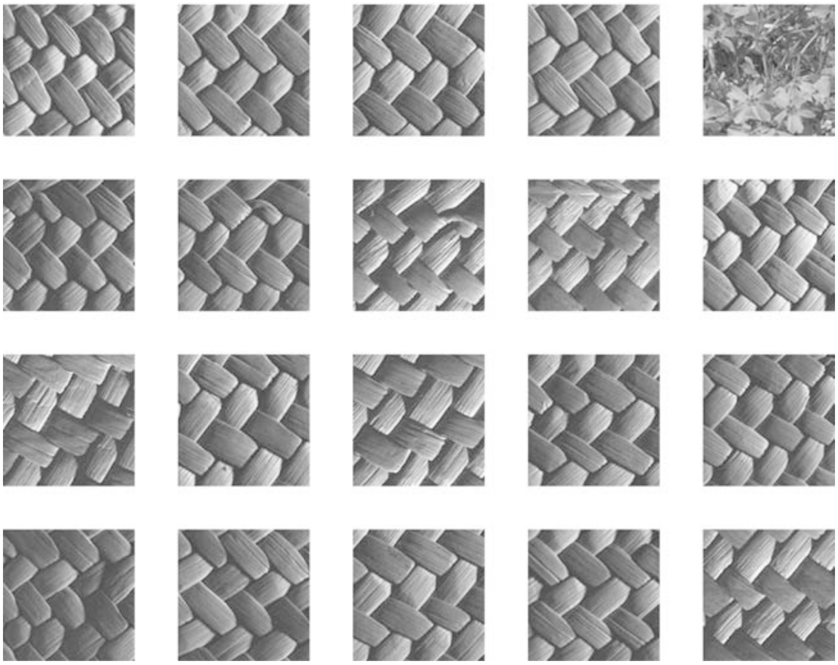

Fig. 12 Retrieval results with our method on a random query from the Texture database. The top leftmost image represents the query image. The images are sorted based on their similarity to the query. The ranks descend from left to right and from top to bottom. Retrieval precision is 0.95

may be desirable when the amount of inter-query learning (i.e., the size of the ‘‘concept database’’) is very large.

- How to adaptively change the ratio of intra to inter-query learning? As was observed, a fixed ratio for combining intra and inter-query learning may result in a drop in performance at some data level.

We plan to study these problems in our future research.

References

1. Balabanovic M (1998) Exploring versus exploiting when learning user models for text recommendation. User Model User Adapted Inter 8(1–2):71–102
2. Burges C (1998) A tutorial on support vector machines for pattern recognition. Data Mining Knowl Discov 2(2):121–167
3. Chen Y, Zhou X, Huang T (2001) One-class svm for learning in image retrieval. In: Proceedings of the 8th IEEE international conference on image processing, Thessaloniki, pp 34–37

4. Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge
5. Ciaccia P, Patella M, Zezula P (1997) M-tree: an efficient access method for similarity search in metric spaces. In: Proceedings of the 23rd international conference on very large databases, pp 426–435
6. Deerwester S, Dumais S, Landauer T, Furnas G, Harshman R (1990) Indexing by latent semantic analysis. J Am Soc Inf Sci 41(6):391–407
7. Faloutsos C, Flocker M, Niblack W, Petkovic D, Equitz W, Barber R (1993) Efficient and effective querying by image content. Technical Report RJ 9453 (83074), IBM Research Report
8. Heisterkamp D (2002) Building a latent-semantic index of an image database from patterns of relevance feedback. In: Proceedings of the 16th international conference on pattern recognition, Quebec City, pp 132–135
9. Hong P, Tian Q, Huang T (2000) Incorporate support vector machines to content-based image retrieval with relevance feedback. In: Proceedings of the 7th IEEE international conference on image processing, Vancouver, pp 750–753
10. Hoppner F, Klawonn F, Kruse R, Runkler T (1999) Fuzzy cluster analysis. Wiley, New York
11. Li M, Chen Z, Zhang H (2002) Statistical correlation analysis in image retrieval. Pattern Recognit 35(12):2687–2693
12. Niblack W, Barber R, Equitz W, Flickner M, Glasman E, Petkovic D, Yanker P, Faloutsos C, Taubin G (1993) The QIBC project: querying images by content using color, texture, and shape. In: Proceedings of storage and retrieval for image and video databases, pp 173–187
13. Peng J, Banerjee B, Heisterkamp D (2002) Kernel index for relevance feedback retrieval in large image databases. In: Proceedings of the 9th international conference on neural information processing, Singapore
14. Peng J, Bhanu B, Qing S (1999) Probabilistic feature relevance learning for content-based image retrieval. Comput Vis Image Understanding 75(1/2):150–164
15. Rui Y, Huang T (1998) Relevance feedback: a power tool for interactive content-based image retrieval. IEEE Trans Circuits Syst Video Technol 8(5):644–655
16. Rui Y, Huang T, Mehrotra S (1997) Content-based image retrieval with relevance feedback in mars. In: Proceedings of the 4th IEEE international conference on image processing, Santa Barbara, pp 815–818
17. Scholkopf B, Williamson R, Smola A, Shawe-Taylor J (1999) SV estimation of a distribution's support. In: Proceedings of the 13th neural information processing systems, Denver, pp 582–588
18. Tax D (2001) One-class classification. PhD Thesis, Delft University of Technology
19. Tong S, Chang E (2001) Support vector machine active learning for image retrieval. In: Proceedings of the 9th ACM international conference on multimedia, Ottawa, pp 107–118
20. Yin P, Bhanu B, Chang K, Dong A (2002) Improving retrieval performance by long-term relevance information. In: Proceedings of the 16th international conference on pattern recognition, Quebec City, pp 533–536