# Improving the Initial Image Retrieval Set by Inter-Query Learning with One-Class SVMs

Iker Gondra[1], Douglas R. Heisterkamp[1], and Jing Peng[2]

[1]Department of Computer Science, Oklahoma State University
Stillwater, Oklahoma 74078, USA
{doug, gondra}@cs.okstate.edu

[2]Department of Electrical Engr. & Computer Science, Tulane University
New Orleans, Louisiana 70118, USA
jp@eecs.tulane.edu

**Abstract.** Relevance Feedback attempts to reduce the semantic gap between a user's perception of similarity and a feature-based representation of an image by asking the user to provide feedback regarding the relevance or non-relevance of the retrieved images. This is intra-query learning. However, in most current systems, all prior experience is lost whenever a user generates a new query thus inter-query information is not used. In this paper, we focus on the possibility of incorporating prior experience (obtained from the historical interaction of users with the system) to improve the retrieval performance on future queries. We propose learning one-class SVMs from retrieval experience to represent the set memberships of users' query concepts. Using a fuzzy classification approach, this historical knowledge is then incorporated into future queries to improve the retrieval performance. In order to learn the set membership of a user's query concept, a one-class SVM maps the relevant or training images into a nonlinearly transformed kernel-induced feature space and attempts to include most of those images into a hyper-sphere. The use of kernels allows the one-class SVM to deal with the non-linearity of the distribution of training images in an efficient manner, while at the same time, providing good generalization. The proposed approach is evaluated against real data sets and the results obtained confirm the effectiveness of using prior experience in improving retrieval performance.

## 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 [12]. 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 [12].

In order to alleviate some of these problems, Content-Based Image Retrieval (CBIR) was proposed. Some early systems include [9, 5]. 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 a feature 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 [12].

Relevance feedback 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 next set of images is retrieved 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 relevance feedback have been proposed [11, 13]. Recently, Support Vector Machines (SVM) have been applied to CBIR systems with relevance feedback to significantly improve retrieval performance [3]. However, in most current systems, all prior experience based on past queries is lost whenever a user generates a new query. That is, the system is adapting to the current user without using any long-term, inter-query learning.

A few approaches [16, 8, 6] attempt inter-query learning. That is, relevance feedback of past queries are used to improve the retrieval for a current query. Both [16] and [8] take the approach of complete memorization of prior history. Then the correlation between past image labeling is merged with low-level features to rank images for retrieval. In [16] the extra inter-query information is efficiently encoded as virtual features. In [6] Latent Semantic Analysis was used to provide a generalization of past experience. The initial results from the three 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 interactions needed to satisfy a query.

In this paper, we propose using one-class SVMs to capture users' query concepts and utilize them as previous experience to be used in future queries. In order to learn the set membership of a user's query concept, a one-class SVM maps the relevant or training images into a nonlinearly transformed kernel-induced feature space and performs risk minimization by attempting to include most of those images into a hyper-sphere of minimum size. The use of kernels allows the one-class SVM 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 one-class SVM 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 one-class SVMs in detail. A description of our proposed approach for improving retrieval performance by using SVMs to capture historical information and fuzzy classification to incorporate it into the relevance feedback method is presented in Section 3. In Section 4, we report experimental results which confirm the effectiveness of our approach. Concluding remarks are presented in Section 5.

## 2 Support Vector Machines

A Support Vector Machine (SVM) is a system for training linear learning machines in a kernel-induced feature space efficiently 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 $\{x_1, x_2, \ldots, x_n\}$ that are vectors in some space $X \in \Re^d$ and their corresponding class labels $\{y_1, y_2, \ldots, y_n\}$ where $y_i \in \{-1, 1\}$. The task of a learning machine would be to learn the mapping $x_i \rightarrow y_i$. The machine is defined by a set of possible mappings $x \rightarrow f(x, \alpha)$, where the functions $f(x, \alpha)$ are labeled by the adjustable parameters $\alpha$ [2]. If there are no restrictions on the family of functions $f(x, \alpha)$ from which we choose our trained machine $f$, even though $f$ may have zero error on the training data, it may not generalize well on unseen 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 class of separating hyperplanes $(x \cdot w) + b = 0$, where $w \in \Re^n$ and $b \in \Re$, corresponding to decision functions $f(x) = \text{sign}((x \cdot w) + b)$. 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 hyperplanes, a constrained quadratic optimization problem is solved. The solution has an expansion $w = \sum_i \alpha_i x_i$. Those points for which $\alpha_i > 0$ are called “support vectors” and lie on one of the separating hyperplanes. 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 $f(x) = \text{sign}(\sum_i \alpha_i (x \cdot x_i) + b)$.

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 $\Phi: \Re^n \rightarrow F$. Because 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 $\Phi(x_i) \bullet \Phi(x_j)$), if we are given a "kernel function" $K$ such that $K(x_i, x_j) = \Phi(x_i) \bullet \Phi(x_j)$, we could just use $K$ without even having to know what $\Phi$ is [2]. 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) is available. For instance, when a user labels some images as "relevant" and others as "non-relevant", information about one class (i.e., the one corresponding to the user's query concept) is given by the "relevant" images. However, the "non-relevant" images do not provide any class information since they can belong to any class. Thus, in one-class classification, the task is to create a boundary around the target class such that most of the target data is included while, at the same time, minimizing the risk of accepting outliers (i.e., data that does not belong to the target class) [15].

The strategy that we will follow to capture a user's query concept (i.e., the target class) is to map the "relevant" images (i.e., the training data) to a higher dimensional feature space and then try to include most of those images into a hyper-sphere. That is, given training data $\{x_1, x_2, \ldots, x_n\}$ that are vectors in some space $X \in \Re^d$, we have to find the smallest hyper-sphere (so that the risk of including outliers is minimized) that includes most of the training data. Thus, the task is to minimize the following objective function (in primal form):

$$\min_{R\in\Re,\ \xi\in\Re^n,\ a\in F} \quad R^2 + C \textstyle\sum_i \xi_i$$

$$\text{such that } \|\Phi(x_i) - a\|^2 \le R^2 + \xi_i, \qquad \xi_i \ge 0, \text{ for } \forall i$$

where $R$ and $a$ are the radius and center of the hyper-sphere, $C$ gives the tradeoff between the radius of the hyper-sphere and the number of training data that can be included, and $\Phi: \Re^n \rightarrow F$. By setting partial derivatives to 0 in the corresponding Lagrangian we obtain the following expression for the center of the hyper-sphere:

$$a = \textstyle\sum_i \alpha_i \Phi(x_i)$$

Replacing partial derivatives into the Lagrangian and noticing that the center $a$ is defined as a linear combination of $\Phi(x_i)$, which allows us to use a kernel, we obtain the following objective function (in dual form):

$$\min \quad \textstyle\sum_{i,j} \alpha_i \alpha_j K(x_i, x_j) - \sum_i \alpha_i K(x_i, x_i) \quad \text{such that } 0 \le \alpha_i \le C, \quad \sum_i \alpha_i = 1$$

This is a quadratic programming problem and the optimal $\alpha$'s can be obtained using a quadratic programming method [15]. In order to determine the ranking of an image x in the database (in terms of belonging to a particular query concept), the following function can be used [15]:

$$\begin{aligned} f(\mathrm{x}) &= R^2 - \|\Phi(\mathrm{x}) - a\|^2 \\ &= R^2 - \sum_{\mathrm{i,j}} \alpha_{\mathrm{i}}\alpha_{\mathrm{j}}\, K(\mathrm{x}_{\mathrm{i}}, \mathrm{x}_{\mathrm{j}}) + 2 \sum_{\mathrm{i}} \alpha_{\mathrm{i}}\, K(\mathrm{x}_{\mathrm{i}}, \mathrm{x}) - K(\mathrm{x}, \mathrm{x}) \end{aligned}$$

where images with higher values are closer to the hyper-sphere and thus, are more likely to belong to the same query concept.

In [3] image retrieval performance is successfully improved by using a one-class SVM for intra-query learning. Scholkopf proposed another approach [14] in which a largest margin hyperplane is used to separate the training data from the origin. When the training data has unit norm this is identical to the approach taken by Tax [15] (explained above). In this paper, as in [3], we will follow the approach developed by Tax [15].

## 3 Proposed Approach

Using one-class SVMs we obtain set membership knowledge (about previous users' query concepts), which can be visualized as hyper-spheres in feature space. In order to integrate this prior experience with a user's current query, we do a fuzzy classification of the user's query concept. When a query is submitted, we determine whether it falls into one of the existing one-class SVMs. Because it is very common for an image to be ascribed into many different concepts, we expect to have queries that fall into many hyper-spheres. One possible way of selecting the images that will be presented to the user would be to perform a hard classification by retrieving the KNN images closest to the nearest center (i.e., KNN images to the closest prototype). However, this purely exploitative approach is not a very good strategy since the query is considered not to be an outlier by several one-class SVMs. It may as well be ascribed to the concept corresponding to any one of the other hyper-spheres. Furthermore, the query may be ascribed to a combination of different concepts. Therefore, instead, we use the ideas from possibilistic cluster analysis [7] and assign a degree of membership to each one of the one-class SVMs (i.e., to each cluster) according to the degree by which the query can be ascribed to its particular concept. A possibilistic cluster analysis drops the probabilistic constraint that the sum of the degrees of membership of each image to all one-class SVMs is equal to one [7]. Therefore, possibilistic cluster partitions are especially useful in the classification of images because it is often the case that an image cannot be assigned to any one of the existing clusters [7].

Let's introduce some terminology. Let $D = \Re^{\mathrm{d}}$ be the *data space* of *d*-dimensional image vectors, $C = \{c_1, c_2, \ldots, c_{\mathrm{n}}\}$ be the *concept space* (i.e., the set of concepts corresponding to all existing one-class SVMs), and $R = \{\{c\} \mid c \in C\}$ be the *result space*. The result of a data analysis is a mapping $\mu: X \rightarrow \{c\}$, where $X \subseteq D$ and $c \in C$. Then, $A(D, R) = \{\mu \mid \mu: X \rightarrow K, X \neq \phi, K \in R\}$ is called an *analysis space* [7]. The *fuzzy set* of $X$ is a mapping $\mu: X \rightarrow [0,1]$ and the set containing all fuzzy sets of $X$ is denoted by $F(X) = \{\mu \mid \mu\colon X \rightarrow [0,1]\}$. Then, $A_{\mathrm{fuzzy}}(D, R) = A(D, \{F(K) \mid K \in R\})$ is the *fuzzy analysis space* for $A(D, R)$. A result of an analysis $\mu\colon X \rightarrow F(K)$ is called a *possibilistic cluster partition* if $\forall k \in K\colon \sum_{x \in X} \mu(x, k) > 0$, where $\mu(x, k)$ is interpreted as the degree of membership of $x \in X$ into the cluster $k \in K$ [7].

The fuzzy *c*-means algorithm carries out a data analysis by minimizing an objective function. It searches for an optimal set of spheres (i.e., clusters) of *d*-dimensional points in a feature space. The clusters are represented by their corresponding centers (i.e., by their prototypes) and Euclidean distance is used as the measure of distance between a point $x$ and a prototype $a_i$ [7]. In our case, the set of clusters (in the form of one-class SVMs) is formed by the historical interaction of users with the system. We use the following membership function to assign degrees of membership to the $n$ hyper-spheres into which a query x falls [7]:

$$\mu(x, a_i) = 1 / \sum_{j=1}^{n} \left(\|\Phi(x) - a_i\|^2 / \|\Phi(x) - a_j\|^2\right) \text{ for } \forall i$$

where $a_i$ is the center of the $i^{th}$ hyper-sphere and $\Phi\colon \Re^n \rightarrow F$. Therefore, the degree of membership of a query into a one-class SVM is based on the relative distances between the query and the centers of all hyper-spheres into which it falls. Suppose a query x falls into $n$ hyper-spheres. If $c_i$ denotes the concept that is embodied by the $i^{th}$ hyper-sphere then the belief (or our degree of confidence) that x is delivering concept $c_i$ is equal to $\mu(x, a_i)$.

Our approach for selecting the set of images that will be presented to the user (i.e, the retrieval set) is based on combining exploitation and exploration while maximizing the number of relevant images in the retrieval set. 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 very good performance. However, for complex queries (i.e., queries that can be ascribed to more than one concept), there is a tradeoff between faster learning of the user's query concept and the delivery of, in our case, more relevant images. In other words, for a complex query, we may be able to maximize the number of relevant images presented to the user by selecting images that can be ascribed to the concept with the largest $\mu(x, a_i)$ (i.e., by pure exploitation). However, we may not be able to learn the user's query concept unless some exploration is also done. The approach that we take is to combine exploitation and exploration while, at the same time, attempting to maximize the number of relevant images that are presented to the user.

To fully exploit the relevance feedback information provided by the current user, we set $\mu(x, a_{\mathrm{current}}) = w$, where $a_{\mathrm{current}}$ is the center of a hyper-sphere formed by using all the feedback provided by the current user (or the original query image when no relevance feedback iterations have been performed yet) and $0 \leq w \leq 1$ signifies our confidence that $a_{\mathrm{current}}$ captures the user's query concept. To form the retrieval set, sample representative images from each hyper-sphere in which the query falls are included. The number of representatives that a particular concept $c_i$ has in the retrieval set is proportional to $\mu(x, a_i)$. 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 $\mu(x, a_i) < \mu(x, a_j)$. Because a query may fall into many hyper-spheres but only a fixed number of images is to be retrieved, priority is given to hyper-spheres with higher $\mu(x, a_i)$ and, after the fixed number of images is reached, the remaining hyper-spheres with smaller $\mu(x, a_i)$ are ignored. This allows us to perform exploitation and exploration while, at the same time, maximizing the possibility of presenting relevant images to the user. The search of the historical one-class SVMs could be done efficiently by building an M-Tree in feature space [10].

At system startup there is no historical interaction of users with the system and thus, no prior experience. A query is then considered to be the center of its own cluster, $w = 1$, and the retrieval set is formed by the same steady-state procedure outlined above. Similarly when there is prior historical information but the query does not fall into any hyper-sphere. The images marked as relevant by the user during the relevance feedback iterations are used as training data and a one-class SVM is used to learn the set membership (in the sense of a user's query concept). Figure 1 shows a block diagram of the system.

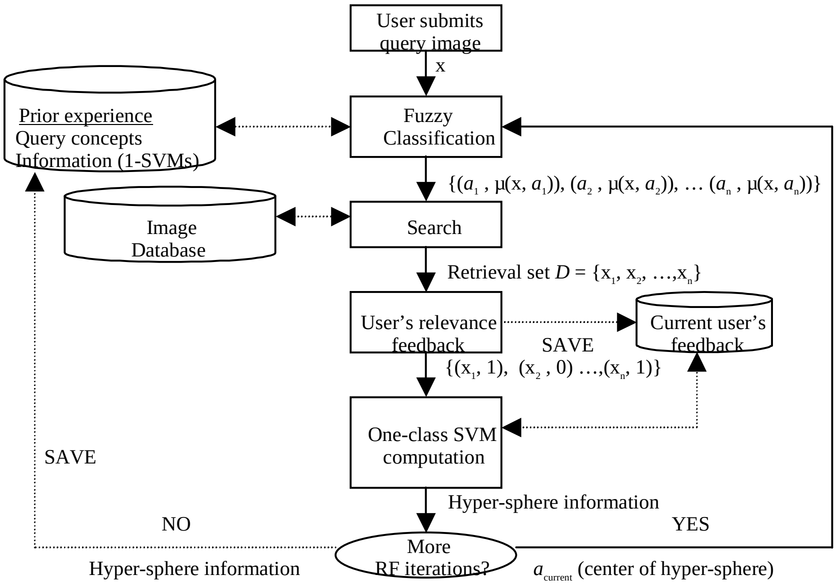

Figure 1. System diagram

## 4 Experimental Results

We compare the performance of our approach against that of relevance feedback methods that do not use historical information. The response of our technique with respect to different amounts of experience (data level) and with respect to quality (noise level) of historical information is also investigated. The retrieval performance is measured by *precision*, which is defined as

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

The *Texture* database, obtained from MIT Media Lab, consists of 640 images of size 128x128. There are 15 classes (corresponding to different textures) and each image is represented by a 16-dimensional feature vector. To determine the free parameters, a ten-fold cross-validation was performed. Our approach was then evaluated with different amounts of experience (data level) using a Gaussian kernel with width $s$ = 0.1, and a misclassification penalty $C = 1/(p*n)$, where $n$ is the number of training images, with $p$ = 0.001. The weight for exploration of historical information is (1-$w$), where $w$ = 0.25. The number of images in the retrieval set $k$ = 20. Figure 2a shows the precision of the initial retrieval (i.e., with no relevance feedback iterations) with respect to different data levels. The values reported are the average of the ten tests. The level of data is the number of hyperspheres relative to the number of images in the database. We can observe that, with low data levels (less than 1/3), precision is less than that obtained by simply retrieving the KNN images in input or feature space, which is the approach to create the initial retrieval set taken by relevance feedback methods that do not use historical information.

In order to avoid the initial decrease in performance we could adaptively change the value of the parameter $w$ so that at the beginning, when there is little historical information, $w$ is large and, as experience accumulates, $w$ becomes increasingly smaller (i.e., more exploration is done). Thus by doing more exploitation at the beginning we would avoid the initial drop in performance. Nevertheless, with the increasing experience, the precision becomes higher and, with significant data levels, there is a large gain in performance.

In order to investigate the robustness of our approach with respect to poor historical information, simulated noise was added. Figure 2b shows the precision of the initial retrieval at different amounts of noise. The percentage of noise is the probability that a users' feedback for each image is flipped (i.e., the probability that an image marked as “relevant” by the user will be marked “non-relevant”, and vice versa). We can observe that, with reasonable amounts of noise, data level is the dominating factor on the performance of our approach and quality of historical information has a small effect.

The *Segmentation* database, which contains 2310 outdoor images, was taken from the UCI repository at http://www.ics.uci.edu/~mlearn/MLRepository.html. There are 7 classes (each with an equal number of instances) and a 19-dimensional feature vector is used to represent each image. A one-fold cross validation was conducted to determine the free parameters. Figure 2c shows the precision of the initial retrieval at different data levels with $s$ = 25, $p$ = 0.001, $w$ = 0.25, and $k$ = 20. Similarly, we can observe that with significant data levels (more than 1/2 in this case), our approach performs better than relevance feedback methods that do not use historical information. From Figure 2d we can observe that our method outperforms an approach based on using a one-class SVM for intra-query learning only (i.e., with data level 0). These results support the efficacy of our method.

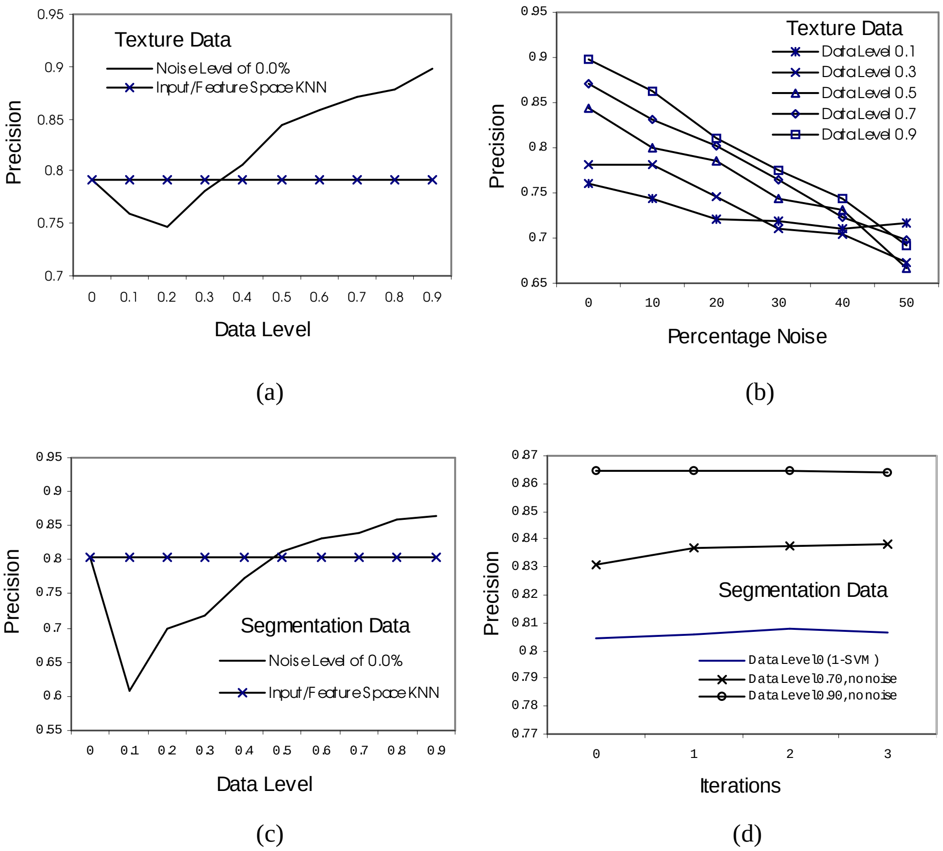

Figure 2. Initial Retrieval Set: (a),(c) Precision vs Data Level, (b) Precision vs Noise Percentage, (d) Precision vs Iteration of Relevance Feedback

## 5 Conclusions

This paper presented an approach for incorporating historical information into a relevance feedback system to improve image retrieval performance. By training one-class SVMs with users' feedback, we can learn users' query concepts and accumulate retrieval experience. Using a fuzzy classification approach we exploit both current and historical information to improve retrieval performance. Initial investigation suggests that our approach improves retrieval in the initial set where a traditional intra-query approach requires an iteration of relevance feedback to provide improvement. Therefore, our method reduces user interaction by reducing the number of iterations needed to satisfy a query. Furthermore, it is robust to poor historical information. Our future research will focus on methods for combining or merging hyper-spheres (i.e., users' concepts). This may be desirable when the amount of historical information is very large. Also, we will investigate some systematic scheme for adaptively changing *w* so that the amount of exploration done is proportional to the amount of experience.

## References

[1] M. Balabanovic. Exploring versus exploiting when learning user models for text recommendation. *User Modeling and User-Adapted Interaction*, 8(1-2):71-102, 1998.

[2] C. J .C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121-167, 1998.

[3] Y. Chen, X. Zhou, and T. S. Huang. One-Class SVM for Learning in Image Retrieval. In *Proceedings of IEEE International Conference on Image Processing*, Thessaloniki, Greece, October 2001.

[4] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[5] C. Faloutsos, M. Flickner, and et al. *Efficient and effective querying by image content*. Technical Report, IBM Research Report, 1993.

[6] D. Heisterkamp. Building a latent-semantic index of an image database from patterns of relevance feedback. In *Proceedings of 16th International Conference on Pattern Recognition*, Quebec City, Canada, August 2002.

[7] F. Hoppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis*. John Wiley & Sons, Inc., New York, 1999.

[8] M. Li, Z. Chen, and H. J. Zhang. Statistical correlation analysis in image retrieval. *Pattern Recognition*, 35(12):2687-2693, 2002.

[9] W. Niblack, R. Barber, and et al. The QBIC project: querying images by content using color, texture and shape. In *Proceedings of SPIE Storage and Retrieval for Image and Video Databases*, pages 173-187, February 1993.

[10] J. Peng, B. Banerjee, and D. Heisterkamp. Kernel index for relevance feedback retrieval in large image databases. In *9th International Conference on Neural Information Processing*, 2002.

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

[12] Y. Rui and T. Huang. Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):644-655, 1998.

[13] Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proceedings of IEEE International Conference on Image Processing*, Santa Barbara, California, pages 815-818, October 1997.

[14] B. Scholkopf, R. Williamson, A. Smola, and J. Shawe-Taylor. SV Estimation of a Distribution's Support. In *Proceedings of Neural Information Processing Systems*, Denver, Colorado, pages 582-588, November 1999.

[15] D. M. J. Tax. *One-class classification*. PhD thesis, Delft University of Technology, http://www.ph.tn.tudelft.nl/~davidt/thesis.pdf, June 2001.

[16] P. Yin, B. Bhanu, K. Chang, and A. Dong. Improving retrieval performance by long-term relevance information. In *Proceedings of the 16th International Conference on Pattern Recognition*, Quebec City, Canada, 3:533-536, 2002.