Summarizing Inter-Query Learning in Content-Based Image Retrieval via Incremental Semantic Clustering

Iker Gondra, Douglas R. Heisterkamp
Department of Computer Science
Oklahoma State University
Stillwater, Oklahoma 74078
{gondra, doug}@cs.okstate.edu

Abstract

In previous work, we developed a novel Relevance Feedback (RF) framework that learns One-class Support Vector Machines (1SVM) from retrieval experience to represent the set memberships of users' high level semantics. By doing a fuzzy classification of a query into the regions of support represented by the 1SVMs, past experience is merged with short-term (i.e., intra-query) learning. However, this led to the representation of long-term (i.e., inter-query) learning with a constantly growing number of 1SVMs in the feature space. We present an improved version of our earlier work that uses an incremental k-means algorithm to cluster 1SVMs. The main advantage of the improved approach is that it is scalable and can accelerate query processing by considering only a small number of cluster representatives, rather than the entire set of accumulated 1SVMs. Experimental results against real data sets demonstrate the effectiveness of the proposed method.

1. Introduction

We can distinguish two different types of information provided by Relevance Feedback (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 Content Based Image Retrieval (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.

A few approaches [3, 4, 7, 12] perform inter-query learning. In [4] Latent Semantic Analysis is used to provide a generalization of past experience. Both [7] and [12] perform a complete memorization of prior history and the correlation between past image labeling is merged with low-level features to rank images for retrieval. The initial results from those approaches for inter-query learning show an enormous benefit in the initial and first retrieval iterations. Therefore, inter-query learning has a great potential for decreasing the amount of user feedback by reducing the number of interactions needed to satisfy a query.

In previous work [3], we presented a novel RF framework that uses One-Class Support Vector Machines (1SVM) to model set membership knowledge about users' high level concepts. For each query, the resulting RF is used to train a 1SVM, which is then saved. Thus, inter-query learning is accumulated in the form of 1SVMs. 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. However, this way of storing inter-query learning results in a constantly increasing number of (possibly overlapping) clusters (i.e., 1SVMs) in the feature space. In this paper, we build on that work by incorporating an implicit cluster-merging process to incrementally summarize the derived inter-query learning. The similarity measure that is used for clustering 1SVMs takes both distance in feature space and a probabilistic perceptual closeness (based on users' RF) into consideration. The main advantage of the improved approach is that it is scalable and it can accelerate query processing by considering only a small number of cluster representatives, rather than the entire set of accumulated 1SVMs.

2. The proposed Approach

Figure 1 shows a diagram of the proposed approach. At the first stage, the user submits a query image $\mathbf{q}$ to the system. Let k be the size of the retrieval set and $0 \leq w_{intra} \leq 1$. The retrieval set that is returned to the user is formed by

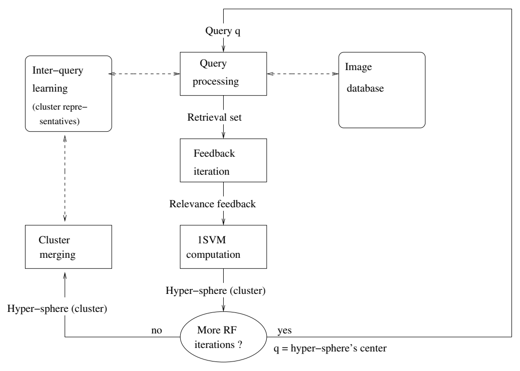

Figure 1. Proposed Approach

including $(w_{intra})k$ nearest neighbor images to $\mathbf{q}$ in feature space and (1-w_{intra})k nearest neighbor images to the cluster representative that is closest to $\mathbf{q}$. Thus, the ratio of intra to inter-query learning that is used in processing a query is w_{intra}:(1-w_{intra}). The M-tree [2] data structure is used for the efficient search of nearest neighbor images. For details, see [2]. At the next stage, the user evaluates the relevance of images in the retrieval set. The images marked as relevant are used as training data for a 1SVM. The center of the resulting hypersphere (i.e., the 1SVM) becomes the new query for the second round and this process continues until the user is satisfied with the results or quits. When the session is over, an implicit cluster-merging process takes place. This process determines, from a fixed number of cluster representatives, the most similar one to the resulting 1SVM and combines both. Thus, inter-query learning is summarized by a small number of cluster representatives.

2.1. 1SVM Computation

We show how the computation of the 1SVM from the user-labelled 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 i^{th} user-labelled 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 $D \geq d$. We want to map the training data to the higher dimensional feature space and obtain a hypersphere that is as small as possible while, at the same time, including most of the training data. That is, the task is to minimize the following objective function (in primal form)

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

$$s.t.\ ||\Phi(\mathbf{x_i}) - \mathbf{a}||^2 \leq R^2 + \zeta_i,\ \zeta_i \geq 0,\ i = 1, 2, \ldots, n$$

where R and $\mathbf{a}$ are the radius and center of the hypersphere, respectively. The parameter $0 \leq C \leq 1$ gives the tradeoff between R and the number of training data that can be included inside the hypersphere. By setting partial derivatives to 0 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 into the Lagrangian it can be noticed that $\mathbf{a}$ is a linear combination of the training data. This allows us to use a kernel function. A kernel K calculates the dot product in the feature space of the images of 2 points from input space, $K(\mathbf{x_i}, \mathbf{x_j}) = <\Phi(\mathbf{x_i}) \cdot \Phi(\mathbf{x_j})>$. We obtain the following objetive function (in dual form)

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

$$s.t.\ 0 \leq \alpha_i \leq C,\ \sum_{i=1}^{n}\alpha_i = 1$$

where K is an appropriate Mercer kernel. We use the Gaussian kernel $K(\mathbf{x_i}, \mathbf{x_j}) = e^{-||\mathbf{x_i}-\mathbf{x_j}||^2/\sigma^2}$, where σ is the width of the Gaussian function. A quadratic programming method is used to find the optimal α values in the objective function [11].

2.2. Similarity Measure

During the intra-query learning process, the system presents a set of images to the user at each RF iteration. The user then labels each image as either relevant (i.e., 1) or non-relevant (i.e., -1). The system then refines the query by performing the 1SVM computation on the set of all relevant images and the process continues until the user is satisfied with the results or quits. Let the accumulated intra-query learning at the end of the RF iterations for the q^{th} query be the following set of 3-tuples

$$IQK^q = \{(id_1^q, rf_1^q, \alpha_1^q), \ldots, (id_m^q, rf_m^q, \alpha_m^q)\}$$

where m is the number of images retrieved (over all RF iterations), $id_i^q \in N$ is the index of an image in the database and refers to the i^{th} retrieved image, $rf_i^q \in \{1, -1\}$ is its corresponding RF, and $\alpha_i^q \in \Re$ is its weight as calculated by the 1SVM computation ($0 \leq \alpha_i^q \leq 1$ if $rf_i^q = 1$, 0 otherwise). The center $\mathbf{a^q}$ of the corresponding hypersphere (1SVM) is then

$$\mathbf{a^q} = \sum_{i=1}^{m}\alpha_i^q\Phi(\mathbf{x}_{id_i^q})$$

where $\mathbf{x}_{id_i^q}$ is the feature vector of the image with index id_i^q in the database. Let the accumulated inter-query learning H be summarized by a fixed number r of cluster representatives. The j^{th} cluster representative C^j is a 2-tuple defined as follows

$$\begin{aligned} C^j &= (\mathbf{p^j}, words^j) \\ words^j &= \{(id_1^j, rf_1^j, w_1^j), \ldots, (id_s^j, rf_s^j, w_s^j)\} \end{aligned}$$

where $\mathbf{p^j}$ is the pre-image of C^j's center in feature space, which is computed as explained in the next section. Thus, the center of C^j in feature space is $\mathbf{c^j} = \Phi(\mathbf{p^j})$. The s images (each with corresponding "semantic weight" $w_i^j \in \Re$) in $words^j$ contribute to C^j's concept. Intuitively, $words^j$ describes the high-level semantics (i.e., the concept) associated with C^j and denoted by $\Psi(C^j)$. For a database image $\mathbf{x}_{id}$, let the set O be defined as follows

$$O \ = \ \bigcup_{h=1}^{r} \{w_i^h | (id, 1, w_i^h) \in words^h\}$$

That is, O contains the "semantic weights" of $\mathbf{x}_{id}$ from all cluster representatives in which $\mathbf{x}_{id}$ appears as a relevant image. Given that the user has labelled $\mathbf{x}_{id}$ as a relevant image and given H, we define the single-image probability that the user's concept is $\Psi(C^j)$ as follows

$$p(\Psi(C^j)|id, H) = \frac{w^j}{\sum_{w \in O} w}$$

where $w^j = w_i^j$ if $w_i^j \in O$, 0 otherwise. Thus, among all $\Psi(C^j)$ in which $\mathbf{x}_{id}$ is relevant, the $\Psi(C^j)$ in which $\mathbf{x}_{id}$ has the largest "semantic weight" has the highest probability of matching the user's concept. The total probability for each $\Psi(C^j)$ is obtained by summing the single-image probabilities of images that are co-occurring and relevant in both $words^j$ and IQK^q. Therefore, the $\Psi(C^j)$ that has the largest semantic overlap will have the highest probability of coinciding with the user's concept. Thus, given IQK^q and H, we define the overall probability that the user's concept is $\Psi(C^j)$ as follows

$$P(\Psi(C^j)|IQK^q, H) = \frac{\sum_{id \in S} p(\Psi(C^j)|id, H)}{|M|}$$

where

$$\begin{aligned} S \ &= \ \{id | (id, 1, *) \in (IQK^q \cap words^j)\} \\ M \ &= \ \{id | (id, *, *) \in (IQK^q \cap words^j) \\ &\quad\ \wedge (id, -1, *) \notin (IQK^q \cap words^j)\} \end{aligned}$$

where $*$ is a "dont-care" symbol indicating that the corresponding tuple element is ignored when determining the intersection. For each cluster representative C^j, we compute its distance to IQK^q with the following similarity measure

$$\begin{aligned} Dist(C^j, IQK^q) \ &= \ (1 - 2P(\Psi(C^j)|IQK^q, H))\Delta \\ &\quad + ||\mathbf{c^j} - \mathbf{a^q}||^2 \end{aligned}$$

where $0 \leq \Delta \leq 1$ is a distance adjustment. Thus, the distance between $\mathbf{c^j}$ and $\mathbf{a^q}$ in feature space is adjusted based on the probability of $\Psi(C^j)$. Therefore,the proposed similarity measure adjusts the distance between the hypersphere and the cluster representatives based on an estimate of their conceptual similarity, which is derived from both the current intra-query and accumulated inter-query learning.

2.3. Pre-Image Computation

As previously shown, the center of a hypersphere (1SVM) is expressed as an expansion in terms of its corresponding support vector images. The center $\mathbf{c^j}$ of a cluster representative C^j is the mean of the centers of all the hyperspheres that have been merged with C^j. Therefore, its location in feature space would have to be expressed in terms of the support vector image of all of those hyperspheres' centers. However, the complexity of distance computations scales with the number of support vectors. Thus, this would result in a system that is both considerably slower and not scalable since the memory needed for storing cluster representatives's centers would continually increase as more queries are processed. This fact motivated us to use pre-images for approximating cluster representatives' centers.

The pre-image problem is to find a point $\mathbf{x} \in \Re^d$ in input space such that, for a given $\vartheta \in \Re^D$ in feature space, $\vartheta = \Phi(\mathbf{x})$. However, since the map Φ into the feature space is nonlinear, this is often impossible (i.e., the pre-image $\mathbf{x}$ does not exist). Therefore, instead, we can find an approximate pre-image $\mathbf{p} \in \Re^d$ such that $||\vartheta - \Phi(\mathbf{p})||^2$ is minimized [10]. Traditional methods [1, 9] solve this optimization problem by performing iteration and gradient descent. The disadvantage of those methods is that the optimization procedure can be expensive and may result in finding a local optimum [10]. The basic idea of the approach presented in [6] is to use distance constraints in the feature space to approximate the location of the pre-image. That is, distances between ϑ and its neighbors in feature space are found. Then, the corresponding input-space distances are computed and used to constraint the location of the pre-image [6]. In this paper, we apply this method to our problem.

Let d_{ci} be the feature-space distance between a cluster representative's center $\mathbf{c^j}$ and a database image $\mathbf{x_i}$. Using the Gaussian kernel defined in Section 2.1 we solve for the corresponding input-space distance $\hat{d}_{ci}$ between $\mathbf{c^j}$ and $\mathbf{x_i}$

$$\hat{d}_{ci} = -\sigma^2 log(1 - \frac{d_{ci}}{2})$$

Let $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k}\}$ be the k nearest neighbor images to $\mathbf{c^j}$ in feature space. Each image $\mathbf{x_i}$ in the database is represented by an n-dimensional feature vector, $\mathbf{x_i} = \{x_{i1}, x_{i2}, \ldots, x_{in}\}$. Then, the problem is to find the least-squares solution $\mathbf{c^j} = \{c_{j1}, c_{j2}, \ldots, c_{jn}\}$ to the system of equations

$$||\mathbf{c^j} - \mathbf{x_i}||^2 = \hat{d}_{ci}, \quad i = 1, \ldots, k$$

After expanding, grouping like terms, and substracting the kth equation from the rest we obtain a system of the form $A\mathbf{x} = \mathbf{b}$, where A is a (k-1) by n matrix with row vectors

$$[2(x_{k1} - x_{i1}) \quad \ldots \quad 2(x_{kn} - x_{in})], \quad i = 1, \ldots, k$$

$\mathbf{b}$ is a (k-1) by 1 vector with rows

$$[-(\mathbf{x_i} \cdot \mathbf{x_i}) + (\mathbf{x_k} \cdot \mathbf{x_k}) - \hat{d}_{ck} + \hat{d}_{ci}], \quad i = 1, \ldots, k$$

and $\mathbf{x} = [c_{j1}, c_{j2}, \ldots, c_{jn}]^T$. We then use the Singular Value Decomposition of A to solve this least-squares problem.

2.4. Cluster Merging

The merging of clusters (i.e., hyperspheres) is the core of our approach. It is used to accelerate query processing by considering only a small number of cluster representatives rather than the entire set of hyperspheres. The k-means (KM) [8] algorithm is one of the simplest and most commonly used clustering algorithms. It starts with a random partitioning of patterns to clusters and keeps reassigning patterns to clusters based on their similarity to cluster centers until there is no reassignment of any pattern from one cluster to another or a convergence criterion is met [8]. We use a modified KM algorithm in which training is done incrementally one pattern (i.e., one hypersphere) at a time as successive queries are processed. The cluster merging algorithm is as follows:

1. Choose k cluster representatives to coincide with the first k hyperspheres.
2. For each input hypersphere IQK^q
   For all cluster representatives $C^j,\ j = 1, \ldots, r$
   (a) Compute $Dist(C^j, IQK^q)$
3. Set $C^{winner} = arg\min_{C^j} Dist(C^j, IQK^q)$
4. Move C^{winner} towards IQK^q
   (a) Move $\mathbf{c^{winner}}$ towards $\mathbf{a^q}$
   (b) Update $\Psi(C^{winner})$ towards $\Psi(IQK^q)$

Therefore, the proposed method for merging a hypersphere with the closest cluster representative is composed of two stages: move the cluster's center in feature space towards the hypersphere's center and update the cluster's concept so that it is more similar to the hypersphere's semantics. At the first stage, a weighted average between the support vector images that make up the hypersphere's center and the cluster's pre-image is taken. Then, the pre-image of the cluster center's new location in feature space is computed. At the second stage, the union between images in IQK^q and $words^{winner}$ is taken. Then, the "semantic weight" of co-occurring relevant images is increased. Similarly, the "semantic weight" of images with opposite feedback is decreased. For any cluster representative C^j, only a fixed number z of images is kept in $words^j$. Thus, when $|words^j| > z$, images with lowest "semantic weight" are deleted.

3. Experimental Results

We compare the performance of the proposed method (i.e., with cluster-merging) against that of the original system in terms of retrieval performance. The goal is to determine whether high retrieval performance can be obtained when summarizing inter-query learning. We have also implemented the Statistical Correlation (SC) approach [7].

The original system presented in [3] keeps the entire set of accumulated 1SVMs. When a query $\mathbf{q}$ is submitted, the retrieval set is formed by including $w_{intra}k$, $0 \leq w_{intra} \leq 1$, nearest neighbor images in feature space to $\mathbf{q}$. The remaining $(1 - w_{intra})k$ images are determined by doing a fuzzy classification of $\mathbf{q}$ into all the 1SVMs into which it falls. Possibilistic cluster analysis [5] is used to assign a degree of membership to each of the 1SVMs (i.e., to each cluster) according to the degree by which $\mathbf{q}$ can be ascribed to that particular concept. A number of images, which is proportional to the degree of membership, is then retrieved from each of the 1SVMs into which $\mathbf{q}$ falls.

The retrieval performance is measured by precision which is the percentage of relevant images in relation to the number of images retrieved. The following data sets were used for evaluation:

Texture - the texture data set, obtained from MIT Media Lab. 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 128x128. Thus, there are 640 images in the database. The images are represented by 16 dimensional feature vectors. We use 16 Gabor filters (2 scales and 4 orientations).

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 2 capital letters O and Q. The images are based on 20 different fonts with randomly distorted letters.

To determine the free parameters, a ten-fold cross-validation was performed for the *Texture* and *Letter* data sets. The values reported are the average of the ten tests.

Figures 2 and 3 show the precision of the initial retrieval set (i.e., with no RF iterations) with respect to different data levels. The data level is the number of processed queries relative to the number of images in the data set. These figures also show the performance obtained by running the proposed method without using pre-images to approximate cluster representatives's centers (i.e., by keeping their full expansions).

Based on these figures, we can observe that the performance loss that results from using pre-images to approximate cluster representatives' centers is small. Furthermore, this loss of performance is well justified by the resulting gain in speed and scalability. We also make the following observations about the advantages of the proposed cluster-

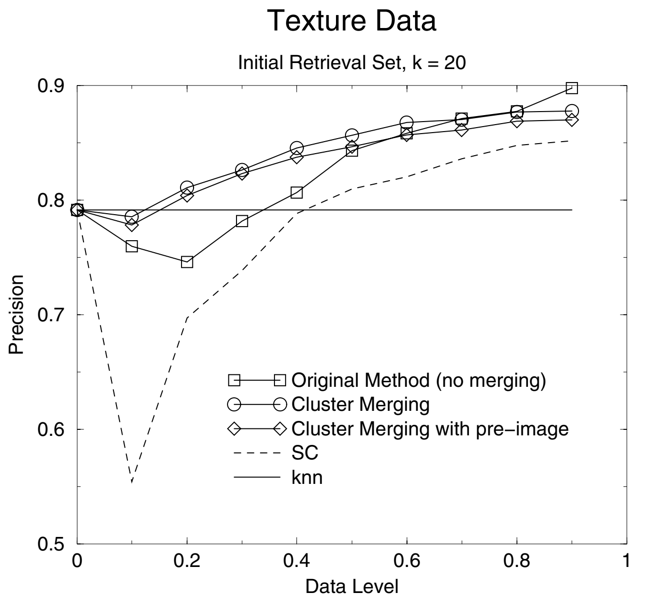

Figure 2. Initial Retrieval Performance

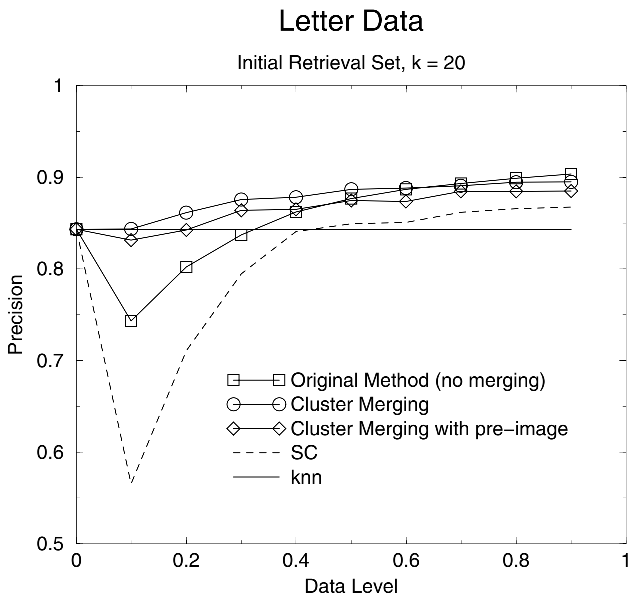

Figure 3. Initial Retrieval Performance

merging approach over the original method: Precision does not degrade with low data levels. It is higher on low data levels and slightly smaller with high levels of data. Also, there is a large gain in execution time; memory requirements are small and do not increase with time.

4. Conclusions

By learning 1SVMs from retrieval experience, it is possible to maintain a database of user's query semantics. In this paper, we proposed using an incremental k-means algorithm to cluster 1SVMs. We used a similarity measure which takes both distance in feature space and a probabilistic perceptual closeness into consideration. The main advantage of the proposed approach is that it is scalable and it can accelerate query processing by considering only a small number of cluster representatives, rather than the entire set of accumulated 1SVMs. Initial investigation suggests that a comparable (or better) retrieval performance can be obtained when summarizing inter-query learning with a small number of cluster representatives.

References

[1] C. Burges. Simplified support vector decision rules. In *Proceedings of the 13th International Conference on Machine Learning*, pages 71–77, July 1996.

[2] P. Ciaccia, M. Patella, and P. Zezula. M-tree: an efficient access method for similarity search in metric spaces. In *Proceedings of the 27th International Conference on Very Large Databases*, pages 34–37, October 2001.

[3] I. Gondra, D. Heisterkamp, and J. Peng. Improving the initial image retrieval set by inter-query learning with one-class svms. In *Proceedings of the 3rd International Conference on Intelligent Systems Design and Applications*, pages 393–402, August 2003.

[4] D. Heisterkamp. Building a latent-semantic index of an image database from patterns of relevance feedback. In *Proceedings of the 16th International Conference on Pattern Recognition*, pages 132–135, August 2002.

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

[6] J. Kwok and I. Tsang. Finding the pre-images in kernel principal component analysis. In *NIPS'2002 Workshop on Kernel Machines*, December 2002.

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

[8] J. McQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[9] S. Mika, B. Scholkopf, A. Smola, K. Muller, M. Scholz, and G. Ratsch. Kernel pca and de-noising in feature spaces. *Advances in Neural Information Processing Systems 11*, pages 536–542, 1999.

[10] B. Scholkopf, S. Mika, C. Burges, P. Knirsch, K. Muller, G. Ratsch, and A. Smola. Input versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.

[11] D. Tax. One-class classification. In *PhD Thesis*. Delft University of Technology, June 2001.

[12] 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*, pages 533–536, August 2002.