On combining one-class classifiers for image database retrieval

Carmen Lai[1], David M.J. Tax[2], Robert P.W. Duin[3]
Elżbieta Pękalska[3], Pavel Paclík[3]

[1] DIEE, University of Cagliari, Sardinia, Italy
carmen@ph.tn.tudelft.nl
[2] Fraunhofer Institute FIRST.IDA, Berlin, Germany
davidt@first.fhg.de
[3] Pattern Recognition Group, TU Delft, The Netherlands
{duin,ela,pavel}@ph.tn.tudelft.nl

Abstract. In image retrieval system, images can be represented by single feature vectors or by clouds of points. A cloud of points offers a more flexible description but suffers from class overlap. We propose a novel approach for describing clouds of points based on support vector data description (SVDD). We show that combining SVDD-based classifiers improves the retrieval precision. We investigate the performance of the proposed retrieval technique on a database of 368 texture images and compare it to other methods.

1 Introduction

In the problem of image database retrieval, we look for a particular image in a huge collection of images. If an example, or a query image is available, we would like to find images, similar to the query, according to our (human) perception. Making an automated system for such a search, would, therefore, require advanced matching methods in order to approximate this. In the paper, we discuss two approaches how images may be represented in an image retrieval system. We propose to represent images by support vector data description (SVDD) for clouds of feature vectors. We show that combining the SVDD representations helps to find good description of the data.

A number of approaches has been investigated how to represent images for image database retrieval [3, 5, 1] Usually, an image is encoded in a single feature vector containing different color-, texture-, or shape-based information about the whole image. This feature vector is computed for all images in the database. To retrieve images resembling the query image, a suitable distance measure between the image feature vectors is needed. The images with smaller distances are then considered to be more similar to the query. This method provides a global description, which does not take into account possible image substructures.

The other, more robust, way to represent images is to encode an image as a set of feature vectors (or a cloud of pixel objects). Usually, simple features like average intensities in small image patches are used. Each image patch is again encoded by a feature vector, storing information about color and texture. The complete image is represented by a set of vectors. We propose to describe this cloud of points by the SVDD method. In order to find the resembling images in the database, a boundary around the image cloud is fitted. Images, whose pixel clouds lie within this boundary, are then the most resembling ones.

Although in this cloud representation the storage and computation costs are much higher, it is much simpler to detect substructures in the original images. Two clearly distinct objects in the image (for instance, a sculpture and a background) will appear as two separate clouds in the feature space. In the 'single-vector' representation it is much harder to detect substructures in the original image.

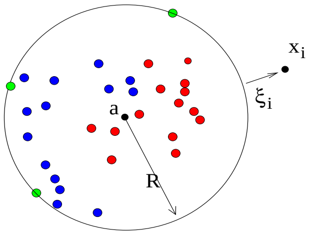


Fig. 1. Graphical representation of the (hyper)sphere around some training data. One object $\mathbf{x}_i$ is rejected by the description (i.e. an error).

Another complication of the proposed approach is, that the pixel clouds of two different images may overlap. It might even happen, that one of the clouds is completely covered by another cloud. Although all the pixels lie within the description of the query image, their distribution is completely different. For similar images, the fraction of pixels lying outside and within the the boundary, will be (roughly) the same.

The SVDD method, employed to represent the cloud of points, is explained in section 2. In section 3, we present the image retrieval problem and two approaches to image representation. The first one uses single feature vectors, while the second method is based on a cloud of points. Later, the combination of individual SVDD one-class classifiers is described. In section 4, the experiments on texture images are presented. Conclusion are summarized in section 5.

2 Support vector data description

First, we give a short derivation of the SVDD [8]. To describe the domain of a dataset, we enclose the data with a hypersphere with minimum volume. By minimizing the volume of the captured feature space, we hope to minimize the chance of accepting outlier objects. Assume we have a dataset containing M data objects, $\{\mathbf{x}_i, i = 1, .., M\}$ and that the hypersphere is described by the center $\mathbf{a}$ and the radius R. A graphical representation is shown in figure 1.

To allow the possibility of outliers in the training set, the distance from $\mathbf{x}_i$ to the center $\mathbf{a}$ must not be strictly smaller than R^2, but larger distances should be penalized. Therefore, we introduce slack variables ξ_i which measure the distance to the boundary, if an object is outside the description. An extra parameter C has to be introduced for the trade-off between the volume of the hypersphere and the errors. Now, we minimize an error L containing the volume of the hypersphere and the distance from the boundary of the outlier objects. We constrain the solution with the requirement that (almost) all data is within the hypersphere:

$$L(R, \mathbf{a}, \boldsymbol{\gamma}) = R^2 + C \sum_i \xi_i \tag{1}$$

$$\|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2 + \xi_i, \qquad \forall_i \tag{2}$$

The constraints (2) can be incorporated in the error (1) by applying Lagrange multipliers [2] and optimizing the Lagrangian. This allows to determine the center as $\mathbf{a} = \sum_i \alpha_i \mathbf{x}_i$ with $0 \leq \alpha_i \leq C$, $\forall i$, and the problem can be changed into maximizing the Lagrangian with respect to $\boldsymbol{\alpha}$:

$$L = \sum_i \alpha_i (\mathbf{x}_i \cdot \mathbf{x}_i) - \sum_{i,j} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j), \quad 0 \leq \alpha_i \leq C \text{ and } \sum_i \alpha_i = 1 \tag{3}$$

This error function is in a standard quadratic form, and combined with the constraints, it gives rise into a quadratic optimization problem. In practice, it appears that a large fraction of the α_i becomes zero. For a small fraction, $\alpha_i > 0$, and the corresponding objects are called *support objects*. These objects appear to lie on the boundary (in figure 1 these are the three light gray objects on the boundary). Therefore, the center of the hypersphere depends just on a few support objects. The objects with $\alpha_i = 0$ can be disregarded in the description of the data. An object $\mathbf{z}$ is then accepted by the description when:

$$\|\mathbf{z} - \mathbf{a}\|^2 = (\mathbf{z} \cdot \mathbf{z}) - 2 \sum_i \alpha_i (\mathbf{z} \cdot \mathbf{x}_i) + \sum_{i,j} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \leq R^2, \tag{4}$$

where the radius R can be determined by calculating the distance from the center $\mathbf{a}$ to a support vector $\mathbf{x}_i$ on the boundary.

Here, the model of a hypersphere is assumed and this will not be satisfied in the general case. Analogous to the method of Vapnik [10], we can replace the inner products $(\mathbf{x} \cdot \mathbf{y})$ in equations (3) and in (4) by kernel functions $K(\mathbf{x}, \mathbf{y})$ which gives a much more flexible method. When we replace the inner products by Gaussian kernels, for instance, we obtain:

$$(\mathbf{x} \cdot \mathbf{y}) \rightarrow K(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2 / s^2) \tag{5}$$

Equation (3) now changes into:

$$L = 1 - \sum_i {\alpha_i}^2 - \sum_{i \neq j} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{6}$$

The maximization of (6), gives $\boldsymbol{\alpha}$, which are used in the computation of the center. It can now be checked if a new object $\mathbf{z}$ lies within the boundary (from (4)):

$$\sum_i \alpha_i K(\mathbf{z}, \mathbf{x}_i) \leq \frac{1}{2} \left(1 - R + \sum_{i,j} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) \tag{7}$$

This Gaussian kernel contains one extra free parameter, the width parameter s in the kernel (from definition (5)). For small values of s the SVDD resembles a Parzen density estimation, while for large s the original hypersphere solution is obtained [9]. As shown in [9], this parameter can be set by setting a priori the maximal allowed rejection rate of the target set, i.e. the error on the target set.

Secondly, we also have the trade-off parameter C. We can define a new variable $\nu = \frac{1}{MC}$, which describes an upper bound for the fraction of objects outside the description [7]. When the user specifies beforehand a fraction of the target objects which can be rejected by the description, just one of the parameters s or ν can be determined. In this paper, we choose, therefore, to set ν to a fixed, small value of 1%. The value of s is optimized such that the user-specified fraction of the data is rejected.

3 Image database retrieval

Let us denote by I_D an image database with N images I_i, $i = 1, ..., N$. The image retrieval problem is formulated as a selection of a subset of images, similar to a given query image Q. In our application, images in the database can be assigned to classes, which describe images coming from the same origin, e.g. grain textures, sky images, images with flowers etc. Therefore, whenever we speak about a class, we mean a group of similar images. In this way, an image retrieval strategy can be tested in a more objective way. Such a strategy is defined in two steps: image representation and a similarity measure between the query image and images stored in the database.

3.1 Image representation

For the sake of image discrimination, images should be represented in a feature space such that the class differences could be emphasized. A convenient way to extract good features is to apply a bank of filters to each image in a database. These filters may be, for example, wavelets, Gabor filters or other texture detectors. In many cases, the filters will give response values which are incomparable to each other. To avoid that one filter with large variance will dominate, the data is preprocessed by weighting individual features on the basis of a dataset mean and standard deviation. We use a scaling that emphasizes differences between individual images in the database.

Assume we have constructed a dataset F containing N K-dimensional feature vectors, representing all images in the database. The weight vector $\mathbf{w}$ is computed element-wise in the following way:

$$w_k = \frac{1}{\mathrm{mean}(F_k)} \ \mathrm{std}\left(\frac{F_k}{\mathrm{mean}(F_k)}\right), \tag{8}$$

where F_k is the k-th feature in the dataset F. All features of all images are rescaled according to this weight vector.

3.2 Single pixel or cloud representation

If we choose to represent one image by one feature vector, the filter responses have to be converted, in one way or another, into a single feature vector. This can be, for example, the average of the filter response over the whole image. All images are then represented by points in a feature space. The similarity between a query image Q and the image I_i from a database may be defined in various ways. For example, Rui *et al.* [6] proposed to use a cosine similarity:

$$Sim(Q, I_i) = \frac{\boldsymbol{x}_Q^T \, \boldsymbol{x}_{I_i}}{||\boldsymbol{x}_Q|| \, ||\boldsymbol{x}_{I_i}||}, \tag{9}$$

where $\boldsymbol{x}_Q$ and $\boldsymbol{x}_{I_i}$ are vector representations of the query and the image I_i respectively and $||\cdot||$ is the L_2-norm. The large Sim value for two vectors in the feature space, the more similar the corresponding images.

Depending on the conversion from an image to a feature vector, it is very hard to retain the individual characteristics of substructures present in the image. For instance, when the original image contains sky and sand in two different parts, the image feature vector will represent the average of the sand and sky characteristics. Only for homogeneous images, the single feature will capture the structure well.

A more flexible image representation can be defined by using a cloud of points, instead. A cloud C_i, representing the image I_i, consists of M_i feature vectors, storing the information on M_i single points or patches in the image. The more compact the cloud, the simpler its separation from the other clouds (images). Such a representation becomes more robust to noise when image patches are used instead of raw pixel intensities.

Such a cloud of points can be used in a number of ways for the image retrieval. For instance, if the assumption of normality holds approximately, the Mahalanobis distance can serve to estimate the similarity between clouds of points. For a good performance, this approach requires also images, homogeneous in the structure.

An another possibility, proposed by us, is to fit the SVDD around the cloud of points. As explained in section 2, the user has to define the percentage of target objects (points) that will lie on the boundary. Given this fraction, a one-class classifier is constructed for the query cloud.

To be more specific, let us introduce the formal notation. Let $\mathcal{C}^i_{\text{SVDD}}$be a one-class classifier constructed for the image I_i. For a vector $\boldsymbol{x}$, coming from the cloud of points C_i, representing the image I_i, i.e. $\boldsymbol{x} \in C_i$, it is defined as:

$$\mathcal{C}^i_{\text{SVDD}}(\boldsymbol{x}) = \mathcal{I}\,(\boldsymbol{x} \text{ is accepted by the SVDD}), \tag{10}$$

where $\mathcal{I}$ is the indicator function (i.e. $\mathcal{I}(A) = 1$ if the condition A is true and is equal to 0, otherwise) and the acceptance of the vector $\boldsymbol{x}$ is defined by formulae (4) or (7), depending on the kernel used. It means that

$$\mathcal{C}^i_{\text{SVDD}}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \text{ is accepted by the SVDD} \\ 0 & \text{if } \boldsymbol{x} \text{ is rejected by the SVDD} \end{cases}$$

This classifier is trained such that the fraction of $p = 0.2$ target vectors lie on the boundary, i.e.:

$$\text{Prob}\left(\mathcal{C}^i_{\text{SVDD}}(\boldsymbol{x}) = 0 \;\;\&\;\; \boldsymbol{x} \text{ is on the boundary } \mid \boldsymbol{x} \in C_i\right) = 0.2, \tag{11}$$

which means that the boundary vectors are here considered to be outliers.

An image I_j is classified by the $\mathcal{C}^i_{\mathrm{SVDD}}$, taking into account the fraction of vectors from the cloud representation C_j, which are rejected by the description, i.e. the fraction S_i of the retained outliers:

$$S_i\left(I_j\right) = \frac{1}{M_j} \sum_{\boldsymbol{x} \in C_j} (1 - \mathcal{C}^i_{\mathrm{SVDD}}(\boldsymbol{x})), \tag{12}$$

where M_j is the size of the cloud C_j. So, the clouds representing other images in the database can now be classified by this one-class classifier, counting the number of outliers for each of them. The smaller the percentage of outliers, the more similar the two images.

3.3 Image similarity by combining one-class classifiers

If only one classifier is used, the performance may suffer from a large overlap between individual clouds of points. For instance, if one cloud completely contains another one, originating from a different class, the percentage of outliers can still be zero. Such an image is then considered to be more similar to the query image than to other images from the same class. This, of course, lowers the performance of the whole image retrieval system.

To prevent such inconvenient situations, we propose to use a set of one-class classifiers, i.e. a combined classifier profile. This profile is then built for the query Q as follows:

$$\boldsymbol{S}(Q) = [S_1(Q), S_2(Q), \ldots, S_N(Q)], \tag{13}$$

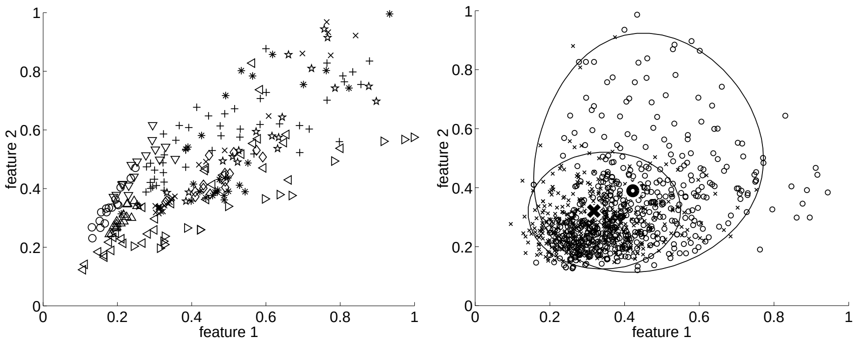
which is the vector of N individual SVDD's responses S_i, defined by (12) for the image Q. Now, our proposal is to compare the query profile with the profiles of the images in the database, on the basis of their similarity. For this purpose, different dissimilarity measures can be used, for instance the Euclidean distance $D_E(Q, I_i) = ||\boldsymbol{S}(Q) - \boldsymbol{S}(I_i)||$, $i = 1, \ldots, N$. In this way, the responses of the individual one-class classifiers are combined to express the dissimilarity between the query image and the images in the database. The images, most similar to the query image, are then retrieved by ranking the dissimilarities $D_E(Q, I_i)$ for $i = 1, \ldots, N$. In this way, the combination rule of the individual SVDD's becomes the trained nearest neighbor combiner. This approach is similar to the decision based on multiple classifiers, proposed by Kuncheva *et al.* [4] where the decision templates are created by averaging over all training objects in a class. In our approach, individual classifiers are constructed for a single image in the database.

4 Experiments

In this section, we describe a set of experiments performed on a dataset of texture images. Our dataset is based on 23 images obtained from MIT Media Lab[4]. Each original image is cut into 16 128×128 non-overlapping pieces. These represent a single class. Therefore, we use a database with 23 classes and 368

[4] `ftp://whitechapel.media.mit.edu/pub/VisTex/`

images. Note that these images are mostly homogeneous and should represent one type of a texture. In this case, it is to be expected that the single feature vector representation performs well.



(a) Single feature vectors in 2D feature space

(b) Two clouds of points and corresponding SVDD boundaries. Bold markers denote corresponding single feature vectors

Fig. 2. Two different image representations.

The images are, one by one, considered as queries. The retrieval precision is computed using all 368 images. The presence of the query image in the training set leads to a slightly optimistic performance estimate. We decided for this approach because it allowed us to work with the complete distance matrices. For each query image, 16 most similar images are found. The retrieval precision for each query is then defined as the percentage of returned images, originating from the same class as the query. The total precision of the retrieval method is then the average precision of all 368 individual queries, i.e.:

$$P = \frac{1}{368} \sum_{I \in I_D} \frac{\#\,\text{images of the same class as I in the first 16 retrieved}}{16} \cdot 100\% \tag{14}$$

The absolute values of responses of 10 different Gabor filters are used as features. These 10 features were chosen by a backward feature selection from the larger set of 48 Gabor filters with different smoothing, frequency and direction parameters. We have used the retrieval precision computed on the vector representation as the feature selection criterion. We used the same set of 10 Gabor filters for all experiments presented in this paper.

4.1 Experiment 1: Image representation by a single feature vector

In this experiment we investigate as a reference the performance of the image retrieval system representing images by single feature vectors. Each vector is computed as the average vector of the corresponding Gabor filter responses.

The data is weighted as described in section 3.1. For an illustration, the scatter-plot of the first two features is shown in figure 2(a). Each point corresponds to a single image; classes are denoted by different markers. As it can be observed, images of the same class are often grouped together. Two dissimilarity measures: cosine distance (9) and Euclidean distance are used for the image retrieval, for which the total precision is presented in the first two rows of table 4.2.

4.2 Experiment 2: Image representation by a cloud of points

In the second set of experiments, we investigate a retrieval system, where the images are represented by clouds of points. An image is described by the average intensities in 9 × 9 pixel neighborhoods. Each cloud consists of 500 patches randomly selected from the image. The choice of 500 is a compromise between a higher standard deviation (noise sensitive) for small number of patches, and a computational complexity. An example of clouds of points in a 2D space is given in figure 2(b). Different markers are used to denote images originating from different classes.

We built an SVDD for the cloud of points, setting 20% of points to the boundary; see (11). Exemplar resulting boundaries in the 2D case are shown in figure 2(b), for which a clear difference in distribution characteristics of the two clouds can be observed.

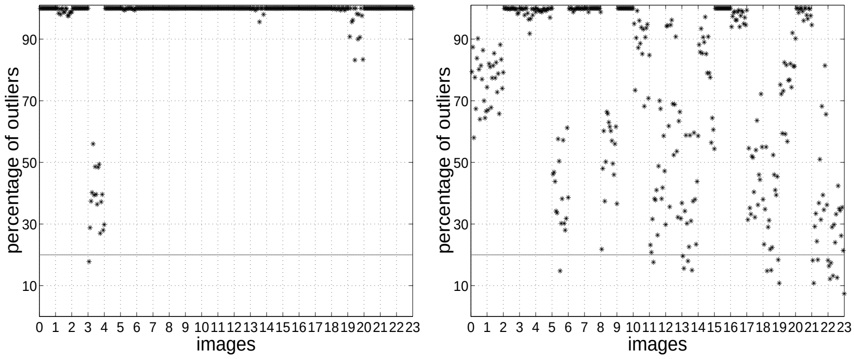


Fig. 3. Percentage of outliers of a query classifier applied to all images. Image from the class 4 is used in the left and image from the class 9 in the right graph.

First, we use a single SVDD, trained on the query cloud and we apply it to other images, represented by clouds. It follows from table 4.2 that the total precision is 67.85% which is worse than 73.44%, obtained by using the single feature vector representation. This can be explained by a heavy overlap between clouds of points, as illustrated in figure 3. This figure shows the percentage of outliers for a classifier trained by a particular query image and applied to all 368 images. The left graph presents the case when a class, containing the query image, is separated from all the other classes. In the right graph, the classifier, trained on the query image (from the class 9), entirely overlaps with several other classes. We judge that, in such cases, combining the responses of a number of one-class classifiers may improve the overall retrieval precision.

The classifier responses form a combined classifier profile, as described in section 3.3. Different approaches may be used to measure the similarity between the query image and other images from a database. In the most obvious attempt, the most similar images to the query Q can be directly found by ranking the elements of the query profile vector. This is achieved by sorting the $\boldsymbol{S}(Q)$ and choosing those individual SVDD's for which the responses are the smallest. This heavily relies on the performance of the single SVDD, which, for highly overlapping classes, is not the optimal approach. Our experiments confirm that, in fact, the total precision is 56.76%.

This motivated us to combine these SVDD's further, by using the trained nearest neighbor combiner, as described in section 3.3. This leads to a decision based on the (dis)similarities between combined classifier profiles for the query and other images. Two different distance measures are considered here: the Euclidean distance and the cosine distance. For the query Q and the image I_i, the latter, based on the inner product between classifier profiles, is computed as $D_{cos} = \frac{1}{2}\left(1 - Sim(\boldsymbol{S}(Q), \boldsymbol{S}(I_i))\right)$, where Sim is defined by (9).

Image representation	Method	Precision [%]
Single feature vector	Euclidean dist.	67.44
	cosine dist.	73.44
Cloud of points	Mahalonobis	19.06
SVDD	target classifier	67.85
SVDD combined	Euclidean dist.	79.11
	cosine dist.	79.40
	ranking	56.76

Table 1. Experimental results: Precisions of different retrieval methods.

5 Summary and Conclusions

The performance of image retrieval systems depends on the selection of an appropriate representation of image data. Usually, an image is represented by a single feature vector. It is an efficient, but sometimes oversimplifying way of information encoding. This type of representation is averaging out details in the images. Other, more complex image representations may be defined, e.g. such as a cloud of points. This is more robust to noise and, at the same time, sensitive to substructures in the data.

To apply this type of a representation, a convenient way of measuring similarity between images must be defined. It should take into account possible multimodality of the data. We have found out that simple methods, such as Mahalonobis distance between clouds of points, suffer because the corresponding assumptions are not fulfilled.

We propose to describe a cloud of points by the support vector data description (SVDD) method. On the contrary to other methods based on the probabilistic approach, SVDD describes the data domain. By this approach, images can be easily matched, based on the fraction of the points rejected by the description (the smaller, the better). It appears that this type of image representation is a flexible tool for image retrieval. However, the retrieval performance of a single SVDD classifier is often badly affected by a large overlap between clouds. To overcome this problem, we propose to combine the one-class classifiers of the database images into a profile of classifiers' responses. An image retrieval is then based on a trained nearest-neighbor combiner.

We have performed a set of experiments on a dataset of 368 texture images. It appears, that a representation by single feature vectors leads to a good retrieval performance, which was expected because of relatively homogeneous images in our database.

In our study, we have investigated different ways of using the SVDD, describing a cloud of points. We have found out that for a single SVDD used, the retrieval performance is worse than for a single vector representation. Therefore, our proposal was to combine the information given by different one-class classifiers, encoded in a vector of their individual responses. Direct ranking in the query profile gives a poor performance, because the outcome is again based on a single pair of clouds. We have found that computing distances between complete classifier profiles is a better strategy. It follows from our experiments that this method outperforms single feature vector-based methods. Moreover, it employs more flexible image representation.

References

1. S. Antani, R. Kasturi, and R. Jain. Pattern recognition methods in image and video databases: past, present and future. In *Advances in Pattern Recognition, Proceedings of SPR'98 and SSPR'98*, pages 31–53, Berlin, 1998. IAPR, Springer-Verlag.
2. Christopher M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.
3. T. Huang, Y. Rui, and S.-F. Chang. Image retrieval: Past, present, and future. In *International Symposium on Multimedia Information Processing*, 1997.
4. Ludmila I Kuncheva, James C Bezdek, and Robert P W Duin. Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recognition*, 34(2):299–314, 2001.
5. K. Messer and J. Kittler. A region-based image database system using colour and texture. *Pattern Recognition Letters*, 20:1323–1330, 1999.
6. Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in MARS, 1997.
7. B. Schölkopf, P. Bartlett, A.J. Smola, and R. Williamson. Shrinking the tube: A new support vector regression algorithm. M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, Advances in Neural Information Processing Systems, 1999.
8. D.M.J. Tax. *One-class classification.* PhD thesis, Delft University of Technology, http://www.ph.tn.tudelft.nl/~davidt/thesis.pdf, June 2001.
9. D.M.J. Tax and R.P.W Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, December 1999.
10. Vladimir N. Vapnik. *Statistical Learning Theory.* John Wiley & Sons., 1998.