Image database clustering with SVM-based class personalization

Bertrand Le Saux and Nozha Boujemaa

IMEDIA Research Group, INRIA, Domaine de Voluceau, Rocquencourt, Le Chesnay, France

ABSTRACT

To allow efficient browsing of large image collections, we have to provide a summary of its visual content. We present in this paper a robust approach to organize image databases: the Adaptive Robust Competition (ARC). This algorithm relies on a non-supervised database categorization, coupled with a selection of prototypes in each resulting category. This categorization is performed using image descriptors, which describe the visual appearance of the images. A principal component analysis is performed for every feature to reduce dimensionality. Then, clustering is performed in challenging conditions by minimizing a Competitive Agglomeration objective function with an extra noise cluster to collect outliers. The competition is improved to be adaptive to clusters of various densities. In a second step, we provide the user with tools to correct possible misclassifications and personalize the image categories. The constraints to deal with for such a system are the simplicity of the user feedback and the rapidity to propose a new category based on the user's criteria.

Keywords: Image Database, Image Retrieval, Relevance Feedback, Unsupervised Clustering, Support Vector Machine Classification

1. INTRODUCTION

Content-based Image Retrieval (CBIR) aims at indexing images by automatic description, which only depends on their objective visual contents. In such a field, to find rapidly a target image in a large collection is a crucial problem. A good organization of the image collection is a key asset to let the user browse it efficiently and retrieve her target image. We propose to first find the main categories of the database and then build a summary by picking key images in each category. This categorization is performed using image signatures, which represent the visual appearance of the image. The main issues of the problem are the unknown number of categories, the high-dimensionality of the feature space, and the complexity of the natural clusters, which are often overlapping.

Our objective was to be able to cope with the "page zero" problem, i.e. how to handle an unknown image collection to formulate a query. Existing systems often start either by a panel of randomly chosen images or by a keyword query. The first option leads to absent categories and may present redundant images, whereas the second one requires to annotate images, often manually, which is a severe limitation to deal with large image volumes.

A few methods have been proposed to browse an image database. The first approach is based on visualization[1,2] techniques, and aims at mapping the image space in a 2D plane. But image clusters may exist in a higher dimensional space than these 2d-manifolds. The second approach is based on database[3,4] techniques, and organizes the images using hierarchical clustering. But the choice of the algorithm (for instance quad-trees) determines the final database organization more than the data themselves.

We developed a fuzzy clustering method able to categorize the image feature space so as to group visually similar images. Prototype-based clustering algorithms are a popular way to find partitions in complex data. Early versions like Fuzzy C-Means (FCM)[5] or density mixture estimations with the EM algorithm[6] have been improved to deal with noisy data either by adding a noise cluster[7] or by the use of M-estimators.[8] However, these methods were not really flexible since the number of clusters had to be fixed before the clustering. Several attempts were made in the last decade to propose methods which automatically estimate the number of clusters in the data-set, in particular the competitive agglomeration[9] which proceeds by reducing the number of clusters over iterations, or the estimation[10] of the number of densities in a melange based on Minimum Message Length.

In a second step, we provide a tool to help the user personalize the image categories according to his own criteria. It aims to address two issues. First, to correct possible misclassifications. This can be due either to the unsupervised way we obtain a first partition, with only a few a priori informations, or to the difficulty of descriptors to distinguish the images. Such a system has to deal with constraints of simplicity of the user feedback and rapidity to propose a new category based on the user's criteria.

We proposed to consider the relevance feedback process as a classification task and to use a Support Vector Machine (SVM) to deal with it. In order to fasten the processing of user's feedback, we propose to modify the standard classification function of the SVM by shifting its bias term. The new bias term is estimated by considering the data to classify and not only the data labeled by the user.

The paper is organized as follows : §2 presents our approach to find categories in image databases: the way we describe images, the clustering method, and some results we obtain. In §3 we introduce the relevance feedback approach used to modify categories according to the user, and explain the shifted-bias classification function. 4 summarizes and evaluates the proposed methods.

2. UNSUPERVISED CLASSIFICATION

2.1. Image description

Images may be described according to different features. Each of them brings a different information. To build the image description space, we use the following features and descriptors:

- color: weighted histograms;
- texture: Fourier power spectrum;
- shape and structure: edge-orientation histograms.

Those descriptors are chosen to be complementary to each other (two color descriptors provide less useful information than a color descriptor used with a texture one). Besides we try to pick for each feature the most discriminant descriptor we can have. But if a new state-of-the-art descriptor is made available, we want to be able to switch to use it in our system. To achieve this, the clustering algorithm should be generic enough to handle any kind of descriptor. Hence we will simply consider descriptors as elements of a vector space, and will not associate specific meaning to them.

Such a feature space leads to challenging conditions. The first of them is its high dimensionality (a simple color histogram may have more than 200 components). In such a space, distances are likely not to correspond to the geometric intuition. Moreover the clustering may become computationally expensive.

We first reduce the dimensionality of the feature space by performing a Principal Component Analysis. For each feature, only the first principal components are kept. Fig. 1 shows the variance of the image descriptor set for each principal component. It appears the variance of the data is concentrated on the first axis. For each descriptor, the reduced dimension is chosen to keep 75% of the variance, which usually leads to keep between 5 and 10 dimensions.

2.2. Clustering method

The purpose of partitional clustering algorithms is to find a partition of the data which minimizes a given objective function. In this case, clusters are modeled by a set of parameters, or prototype. In particular, methods of the FCM family aim to minimize quadratic error functions which measure the distances between data and cluster prototypes. They produce a fuzzy partition of the data, i.e. each data may belong to a cluster with a degree of membership in $[0, 1]$.

The Competitive Agglomeration was introduced by Frigui & Krishnapuram[9] to be able to estimate on the fly the number of clusters of the final partition. It controls the complexity of the model (here the number of clusters) through the use of regularization which involves the addition of a penalty term Ω to the error function E:

$$\tilde{E} = E + \alpha\Omega \tag{1}$$

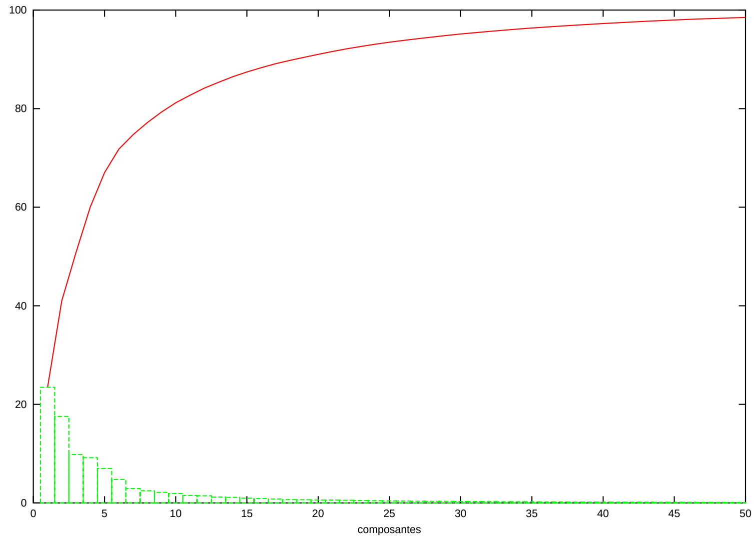

Figure 1. variance and cumulated variance per each principal component of the feature space, for a generic database.

The parameter α has to be chosen carefully to control the extent to which the penalty term influences the form of the solution. Basically, the algorithm starts with a large number of clusters, and at each iteration discards the ones which do not represent enough data. Hence the clusters have to compete so that enough data belong to them. α allows to tune the fierceness of this competition, and so is called competition factor.

Such an algorithm family offers key assets for an application like ours. First the ability to estimate automatically the number of clusters in the data set allows to face one of the main issues of image database clustering. Then it allows to retrieve clusters with various shapes by using an appropriate distance. Though, it suffers from some major weaknesses. It hardly handles noisy data, and is harmed by clusters of various densities. In such a case, wide-spread clusters are likely to be discarded.

To be able to retrieve clusters of various densities, we define a new objective function with one competition factor for each cluster:

$$E_{\text{arc}}(U,B) = \sum_{j=1}^{C}\sum_{i=1}^{N}(u_{ji})^2 d^2(x_i,\beta_j) - \sum_{j=1}^{C}\alpha_j\left[\sum_{i=1}^{N}(u_{ji})\right]^2 \tag{2}$$

Where the data are represented by a set of N vectors $X = \{x_i \in \mathbb{R}^p \ \mid i \in \{1\cdots N\}\}$ and the set of the prototypes is $B = \{\beta_j \in \mathbb{R}^p \ \mid j \in \{1\cdots C\}\}$, which for instance may be centroïds of the clusters. The α_j will be defined below. U is the set of the memberships. The membership of vector x_i to the cluster defined by β_j is denoted u_{ji} and satisfies the constraints:

$$\begin{aligned} \forall i \in [1,N], \forall j \in [1,C] \qquad & 0 \le u_{ji} \le 1 \\ \forall i \in [1,N] \qquad & \sum_{j=1}^{C} u_{ji} = 1 \end{aligned} \tag{3}$$

The first term of Eq. (2) is the standard FCM objective function and allows to control size and shape of the clusters. It is minimal when each data is in a different cluster. The second one is a modified regularization term (if compared to standard Competitive Agglomeration) which allows to adapt the influence of each cluster in the competition. The minimum of this penalty term is achieved when all data belongs to one cluster.

The whole function is minimized iteratively with a 2-step process at each iteration similar to the Expectation-Maximization algorithm[6]: first, given the parameters of the model, the expected memberships are estimated, and secondly these parameters are updated using the new membership values.

To find the optimal expression to update the memberships and parameters which minimize the objective function upon iterations with respect to the constraints (3), the corresponding Lagrangian function is computed. This yields to have the following expression of the membership:

$$u_{st} = u_{st}^{\text{fcm}} + u_{st}^{\text{bias}} \tag{4}$$

with:

$$u_{st}^{\text{fcm}} = \frac{[1/d^2(x_t, \beta_s)]}{\sum_{j=1}^{C}[1/d^2(x_t, \beta_j)]}, \tag{5}$$

and:

$$u_{st}^{\text{bias}} = \frac{1}{d^2(x_t, \beta_s)}\big(\alpha_s N_s - \frac{\sum_{j=1}^{C}[1/d^2(x_t, \beta_j)]\alpha_j N_j}{\sum_{j=1}^{C}[1/d^2(x_t, \beta_j)]}\big) \tag{6}$$

The first term (5) is the standard membership of the FCM algorithm and considers only the relative distances of the feature point to all clusters. Its value tends to 1 when the point x_t comes closer to the prototype β_s, whereas for for other clusters $j \neq s$ u_{jt}^{fcm} tend to 0. The second term (6) is a bias term which compares the current cluster to the other ones. The comparison is made on the basis of the product $\alpha_s N_s$. If its value for the current cluster is lesser the the average value on all the clusters, the bias will be negative and the memberships (4) will depreciate. On the contrary, if the value of it is larger than average, memberships to this cluster will appreciate. Then the new cluster cardinalities are computed after these memberships, and are used to evaluate the respective importances of the clusters. If a cluster cardinality is below a fixed threshold, it will be discarded. By choosing an appropriate competition factor α_s, we intend to be able to adapt the competition to various cluster natures.

In order to have u_{st}^{bias} comparing different densities, we propose to make α_s proportional to a measure of the size of the cluster. The weighted average distance of members of a cluster to their centroïds gives a good indication of the spareness of this cluster:

$$\forall s \in [1, C] \quad d_{moy}^2(s) = \frac{\sum_{i=1}^{N}(u_{si})^2 d^2(x_i, \beta_s)}{\sum_{i=1}^{N}(u_{si})^2} \tag{7}$$

The average distance over all the data set is also computed for normalization purpose:

$$d_{moy}^2 = \frac{\sum_{j=1}^{C}\sum_{i=1}^{N}(u_{ji})^2 d^2(x_i, \beta_j)}{\sum_{j=1}^{C}\sum_{i=1}^{N}(u_{ji})^2} \tag{8}$$

The competition factor α_s for cluster s is defined as:

$$\forall s \in [1, C] \quad \alpha_s(k) = \frac{d_{moy}^2}{d_{moy}^2(s)}\alpha(k) \tag{9}$$

where $\alpha(k)$ is the standard factor of CA, to benefit from the critical optimization of the CA algorithm. It is carefully chosen so that both terms of the objective function have comparable influences during the minimization.

The bias term can be written as:

$$u_{st}^{\text{bias}} = \frac{d_{moy}^2 \alpha(k)}{d^2(x_t, \beta_s)}\big([N_s/d_{moy}^2(s)] - \frac{\sum_{j=1}^{C}[1/d^2(x_t, \beta_j)][N_j/d_{moy}^2(j)]}{\sum_{j=1}^{C}[1/d^2(x_t, \beta_j)]}\big) \tag{10}$$

The new bias term depends on the difference between a measure of the empirical density of a cluster and the average density on the whole data set. For clusters with empirical densities higher than the average, u_{st}^{bias} is positive, so memberships to these clusters are augmented. Hence the cardinality of such clusters will appreciate and make them more resistant to the discard process.

Image databases are likely to contain images that can not be assigned to a specific class. These outliers will not only be misclassified, but will harm the clustering too. We propose to collect those noise points in a separate cluster.[7] A virtual prototype is defined, located at the same distance of all the data. Let this noise cluster be the first one:

$$\forall i \in [1, N] \quad d^2(x_i, \beta_1) = \delta^2 \tag{11}$$

The constant distance for the noise cluster is defined after the distances on the data set. It will be used as a proximity threshold to estimate outliers: if no real prototype is closer than a ratio of the average distance between points and prototypes, current point will belong to the noise cluster:

$$\delta^2 = \delta_0^2 \frac{\sum_{j=2}^{C} \sum_{i=1}^{N} d^2(x_i, \beta_j)}{N(C-1)} \tag{12}$$

During competition, the noise cluster cannot be discarded. Though, it has an influence on the elimination of other clusters. These ones have better estimated parameters (with no bias due to outliers), and their validity is estimated according to more likely cardinalities.

The distances used in Eq. 12 for the clusters corresponding to real classes aim to combine various features. We define the global distance as a weighted sum of partial distances on each separate feature $1 \leq f \leq F$:

$$d(x_i, \beta_j) = \sum_{f=1}^{F} w_{j,f} d_f(x_i, \beta_j) \tag{13}$$

In each feature subspace, real classes have various shapes which are difficult to estimate. The use of a fuzzy Mahalanobis distance[11] allows us to retrieve hyper-ellipsoidal clusters which could be considered as a good estimation of the main classes. For clusters $2 \leq j \leq C$, partial distances are defined using:

$$d_f(x_i, \beta_j) = |C_{j,f}|^{1/p_f} (x_{i,f} - \beta_{j,f})^T C_{j,f}^{-1} (x_{i,f} - \beta_{j,f}) \tag{14}$$

where $x_{i,f}$ and $\beta_{j,f}$ are the restrictions of x_i and β_j to subspace (of dimension p_f) corresponding to feature f. $C_{j,f}$ is the fuzzy covariance matrix of cluster j for the feature f and is computed according to:

$$C_{j,f} = \frac{\sum_{i=1}^{N} (u_{ji})^2 (x_{i,f} - \beta_{j,f})(x_{i,f} - \beta_{j,f})^T}{\sum_{i=1}^{N} (u_{ji})^2} \tag{15}$$

The weights of the partial distances are updated during the clustering process in order to be able to find on the fly clusters particularly relevant to one of the features. Some images are likely to be similar according to a particular feature when the partial distances of the corresponding descriptor is low. In such a case, a higher weight should be applied to this feature. More formally, the combination is processed according to the following steps:

1. Sort the partial distances by increasing order (for each couple image-prototype)
2. Compute average rank $r_j^{(f)}$ of each descriptor for a given prototype β_j
3. Compute a weight (at iteration k) proportional to the importance of the descriptor for a given prototype according to $w_{j,f}^{(k)} = \frac{2(F - r_j^{(f)})}{F(F+1)}$
4. Update final weight of each partial distance for a given prototype:

$$W_{j,f}^{(k)} = \frac{(k-1)W_{j,f}^{(k-1)} + w_{j,f}^{(k)}}{k} \tag{16}$$

Figure 2. Summary of a ground-truth database[13] used as a *page zero* and a typical category.

2.3. Categorization results

From the fuzzy partition we obtain, we generate some image categories by assigning one image to the cluster to which it has the highest membership degree. In each category, one image is chosen in order to represent the cluster. The average value is computed for each feature (e.g. mean color for a color feature) and the representative is the real image closest to the centroïd of these average features. This is more robust to possible misclassified images.

By gathering these representatives, we are able to provide a summary which can be used as a *page zero* for a CBIR system (cf. Fig. 2). We refer to previous works[12] for comparison with other clustering methods. ARC is shown to provide a good trade-off between both efficient summaries and categories with few false positives: the summary presents less redundant images and provides a good overview of the whole image collection, whereas categories are the more likely to represent the real classes of the data-set, with only a small amount of misclassified images.

3. LEARNING-BASED CLASSIFICATION

3.1. Category relevance feedback

To have only a few misclassified images in the categories obtained in an unsupervised way could be considered as an asset if one considers the few informations needed by the clustering algorithm. However the user would prefer some perfect categories. We provide her with the opportunity to correct the clustering and to modify the database organization. Besides, in the highly-semantical context of image databases, such a tool can be used to group the images according to a subjective criterion and to personalize the categories.

Since the interpretation of images is highly subjective, relevance feedback became popular in image retrieval.[14,15] The usual scenario consists in iterations of the following steps:

- the system provides a first answer to the user's request;

- the user gives pertinence degree for some returned images;
- the system learns from this feedback and proposes a new answer.

Different approaches are used: Bayesian methods estimate the probability of each image to be the target-image according to the history of the request[16,17]; density estimation methods estimate the distribution of positive examples; query re-weighting methods give more importance to a feature which seems particularly relevant to the user[18]; classification methods and in particular SVM separate relevant images from irrelevant ones.[19]

We want a method able to learn from given samples which images are relevant. First the user has to select samples. To be efficient, the selection must be short and intuitive. Then, the processing of user's feedback must satisfy rapidity constraints too. We choose to consider the relevance feedback process as a classification task and to use a SVM[20] to classify the data.

3.2. SVM with shifted bias

Here the data to classify is one of the categories of the database. The user has to label a few images as positive or negative to define the class he is considering as visually relevant. The SVM is trained on the descriptors of these labeled images. In order to use all the available information, the complete descriptors of images are used, and not only the reduced ones like for clustering. The small sizes of both training and test sets let us expect short process times.

However, the standard criterion of the SVM does not give satisfying classification results. Practically, the images are correctly ordered: values of $f(x)$ greater than 1 correspond to relevant images, and values lesser than -1 to types of images labeled as irrelevant. But the sign change performs badly to indicate the change of visual class. Either there are some false positives in the images returned to the user, or some relevant images are discarded (which is more critical since these images are lost for the user).

Two problems occur in this particular case. First the few samples used to train the classifier may be far from the real frontier. Hence a lot of points from the test set may fall in the uncertainty zone between the support vectors. It happens the samples do not represent correctly the test set. Besides, some asymmetry between positive and negative examples is possible. We can assume that relevant images are correctly labeled. But the user may not label all the different types of misclassified images, so several irrelevant images of the same type may fall in the uncertainty zone and lead to wrong results. Usually the decision frontier is chosen as equidistant of the two canonical hyperplanes (the ones supported by the support vectors) but in this case it may be set closer to the positive canonical hyperplane.

The images are first ordered according to the decreasing order of the values of the decision function:

$$f(x) = \sum_j \alpha_j y_j K(x_j, x) + b \tag{17}$$

where the couples (x_j, y_j) are the training data with their corresponding labels, $K(.,.)$ is the chosen kernel function of the SVM, the α_j result from the optimization of the SVM problem (they are null for most training data and non-null for support vectors) and the bias b is computed in order to have a frontier equidistant from both positive and negative vector supports.

The images from the same real class are correctly grouped, so we just have to shift the bias to adjust the frontier to the data. We notice that the decision function presents discontinuities between two distinct classes. We aim to retrieve those " jumps" to be able to detect the class changes. For that purpose, we compute the discrete derivate of the decision function for image of rank r:

$$f'(x_r) = f(x_r) - f(x_{r-1}) \tag{18}$$

Since the discontinuities of the decision function correspond to peaks of the derivate, we are looking for exceptional values of it, i.e. when:

$$-f'(x_r) \geq \mu_{f'} + 3\sigma_{f'} \tag{19}$$

Finally, the frontier is chosen as the rank of the first peak after the last positive support vector:

$$r_{\text{threshold}} = \min_{\forall i \mid x_i \; SV+ \; r>i} \arg(f'(x_r) | f'(x_r) \geq s) \tag{20}$$

3.3. Comparison of both methods

Fig. 3 shows one of the categories, which contain mostly images of the same type, and a few misclassified ones. The user labels only 4 images (2 as positive, 2 as negative). After training the classifier with these samples, the result of the classification of images of the whole category with both decision criterion is shown on Fig. 4. Some misclassified images remain with the standard criterion of the SVM. On the contrary the precise category is correctly estimated when the shifted-bias criterion is used.

See Fig. 5 for the graph of the classification function obtained with these training samples, after the test data have been correctly ordered. With respect to margin-maximization paradigm, it can be assumed the gap between two images from different visual classes is larger than the one between images from the same visual category. This leads to some discontinuities of the function, and hence to peaks of its derivate. The two possible frontiers are displayed as vertical lines. The standard one, corresponding to sign change, leads to the false positives shown on Fig. 4-a. After a shift of the bias to the discontinuity next to the first support vector, the correct class is estimated with precision.

In the standard criterion case, new iterations of the feedback process are required to obtain a good estimation of the category. With regard to our rapidity constraints, time spent by the user to refine the categories is a crucial issue. Fig. 6 summarizes the proportion of iterations required to estimate correct classes. The relevance feedback is processed on all the categories of the ground-truth database. It appears that more than 80 % of classes are corrected in one step with the shifted bias criterion, while three steps are necessary to obtain the same proportion of corrected categories. On average, 2 times more iterations are needed to correct one class.

4. CONCLUSION

We have presented a method to organize an image database in a 2-step process. First categories of visually similar images are estimated by our unsupervised clustering method ARC. This step is off-line and the approach allows to face the main issues of automatic categorization:

- estimate automatically the number of clusters,
- handle noisy data,
- cope with various class densities and shapes.

In a second step, we provide the user with a tool to change the database organization and fit to his own ideas. His feedback on categories is processed through a support vector machine to classify the images as relevant or not with respect to the current category. We have proposed a method to estimate with precision the frontier between classes of visually similar images by shifting the bias according to the data to classify.

However, if the relevance feedback method gives good practical results, the generalization capacity of this classifier is not proved. Further works will investigate how we can move the decision frontier without increasing the guaranteed risk too much. This problem has an elegant formulation in the framework of transductive learning.

REFERENCES

1. R. Brunelli and O. Mich, "Image retrieval by examples," *IEEE Transactions on Multimedia* **2**(3), pp. 164–171, 2000.
2. Y. Rubner, L. J. Guibas, and C. Tomasi, "The earth mover's distance, multi-dimensional scaling, and color-based image retrieval," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 59–66, (Bombay, India), 1998.
3. J.-Y. Chen, C. A. Bouman, and J. C. Dalton, "Hierarchical browsing and search of large image databases," *IEEE Transactions on Image Processing* **9**(3), pp. 442–455, 2000.

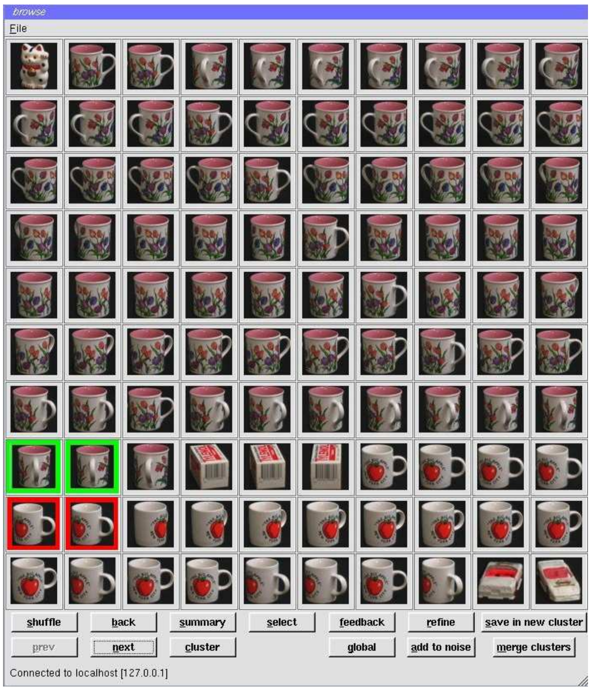

Figure 3. Selection of the training set: two images are labeled as positive, and two as negative.

(a) (b)

Figure 4. Comparison of classification results: (a) with the standard criterion of the SVM and (b) with the shifted bias criterion.

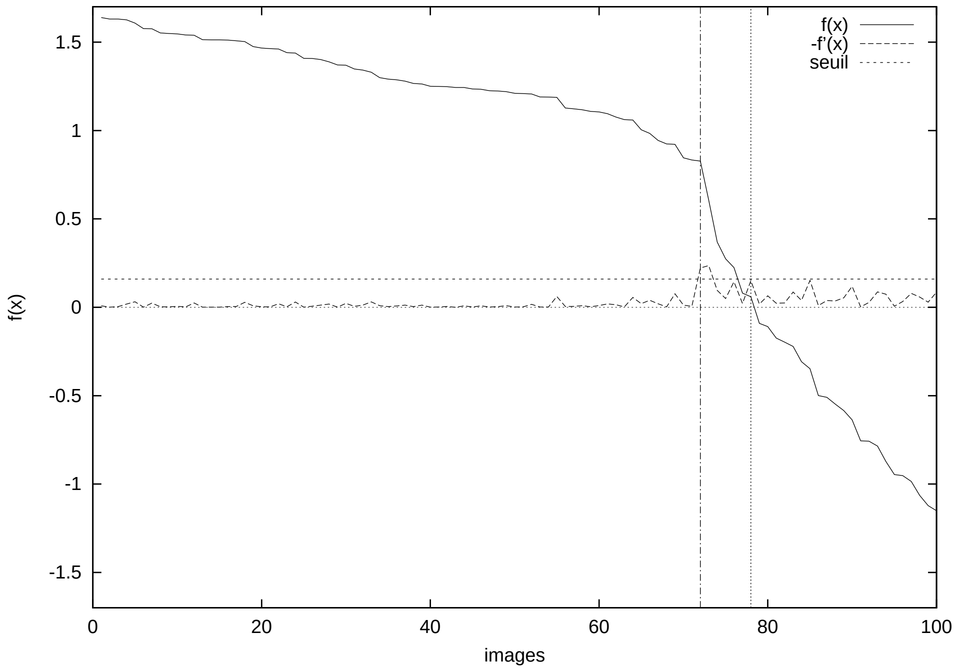

Figure 5. Graph of the classification function of the SVM.

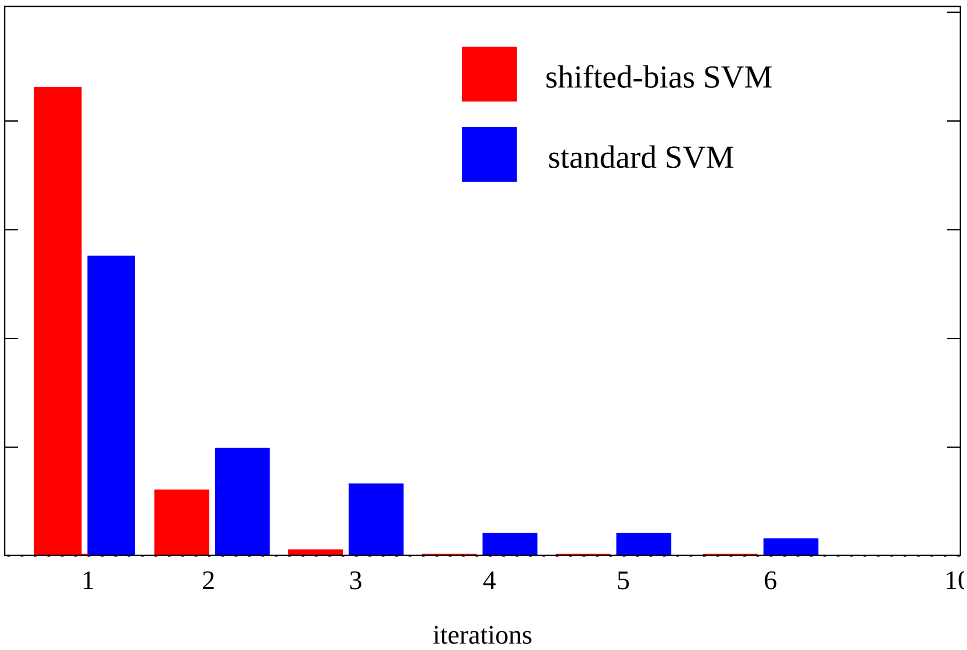

Figure 6. Comparison of the number of iterations required to eliminate all the misclassified images from one category.

4. T. Pun and D. Squire, “Statistical structuring of pictorial databases for content-based image retrieval systems,” *Pattern Recognition Letters* **17**(12), pp. 1299–1310, 1996.
5. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New-York, 1981.
6. A. Dempster, N. Laird, and D. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of Royal Statistical Society* **39**, pp. 1–38, 1977.
7. R. N. Davé, “Characterization and detection of noise in clustering,” *Pattern Recognition Letters* **12**(11), pp. 657–664, 1991.
8. H. Frigui and R. Krishnapuram, “A robust competitive clustering algorithm with applications in computer vision,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **21**(5), pp. 450–465, 1999.
9. H. Frigui and R. Krishnapuram, “Clustering by competitive agglomeration,” *Pattern Recognition* **30**(7), pp. 1109–1119, 1997.
10. M. Figueiredo and A. K. Jain, “Unsupervised learning of finite mixture models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(3), pp. 381–396, 2002.
11. E. E. Gustafson and W. C. Kessel, “Fuzzy clustering with a fuzzy covariance matrix,” in *IEEE CDC*, pp. 761–766, (San Diego, California), 1979.
12. B. Le Saux and N. Boujemaa, “Unsupervised robust clustering for image database categorization,” in *Proceedings of IEEE-IAPR International Conference on Pattern Recognition*, (Quebec, Canada), august 2002.
13. S. A. Nene, S. K. Nayar, and H. Murase, “Columbia object image library (coil-100),” tech. rep., Department of Computer Science, Columbia University, http://www.cs.columbia.edu/CAVE/, 1996.
14. T. Kurita and T. Kato, “Learning of personal visual impression for image database systems,” in *Proceedings of International Conference on Document Analysis and Recognition*, pp. 547–552, 1993.
15. R. W. Picard, T. P. Minka, and M. Szummer, “Modeling user subjectivity in image libraries,” in *IEEE International Conference on Image Processing*, pp. 777–780, (Lausanne, Switzerland), 1996.
16. I. J. Cox, M. L. Miller, T. P. Minka, T. V. Papathomas, and P. N. Yianilos, “The bayesian image retrieval system, PicHunter,” *EEE Transactions on Image Processing* **9**(1), pp. 20–37, 2000.
17. D. Geman and R. Moquet, “A stochastic feedback model for image retrieval,” in *Proceedings of RFIA*, pp. 173–180, (Paris, France), 2000.
18. Y. Rui, T. S. Huang, and S. Mehrotra, “Relevance feedback: A power tool in interactive content-based image retrieval,” *Transactions on Circuits and Systems for Video Technology* **8**(5), pp. 644–655, 1998.
19. S. Tong and E. Chang, “Support vector machine active learning for image retrieval,” in *Proceedings of ACM International Conference on Multimedia*, pp. 107–118, (Ottawa, Canada), 2001.
20. V. Vapnik, *The Nature of Statistical Learning Theory*, Springer Verlag, New-York, 1995.