Learning Distance Functions for Image Retrieval

Tomer Hertz, Aharon Bar-Hillel, and Daphna Weinshall
School of Computer Science and Engineering and the Center for Neural Computation
The Hebrew University of Jerusalem, Jerusalem, Israel 91904
Email: { tomboy,aharonbh,daphna}@cs.huji.ac.il

Abstract

Image retrieval critically relies on the distance function used to compare a query image to images in the database. We suggest to learn such distance functions by training binary classifiers with margins, where the classifiers are defined over the product space of pairs of images. The classifiers are trained to distinguish between pairs in which the images are from the same class and pairs which contain images from different classes. The signed margin is used as a distance function. We explore several variants of this idea, based on using SVM and Boosting algorithms as product space classifiers. Our main contribution is a distance learning method which combines boosting hypotheses over the product space with a weak learner based on partitioning the original feature space. The weak learner used is a Gaussian mixture model computed using a constrained EM algorithm, where the constraints are equivalence constraints on pairs of data points. This approach allows us to incorporate unlabeled data into the training process. Using some benchmark databases from the UCI repository, we show that our margin based methods significantly outperform existing metric learning methods, which are based on learning a Mahalanobis distance. We then show comparative results of image retrieval in a distributed learning paradigm, using two databases: a large database of facial images (YaleB), and a database of natural images taken from a commercial CD. In both cases our GMM based boosting method outperforms all other methods, and its generalization to unseen classes is superior.

1. Introduction

Image retrieval is often done by computing the distance from a query image to images in the database, followed by the retrieval of nearest neighbors. The retrieval performance mainly depends on two related components: the image representation, and the distance function used. Given a specific image representation, the quality of the distance function used is the main key to a successful system.[1]. In this paper we focus on learning 'good' distance functions, that will improve the performance of content based image retrieval. The quality of an image retrieval system also depends on its ability to adapt to the intentions of the user as in relevance feedback methods [3]. Learning distance functions can be useful in this context for training user dependent distance functions.

Formally, let $\mathcal{X}$ denote the original data space, and assume that the data is sampled from M discrete labels where M is large and unknown. Our goal is to learn a distance function $f : \mathcal{X} \times \mathcal{X} \rightarrow [0, 1]$. In order to learn such a function, we pose a related binary classification problem over product space, and solve it using margin based classification techniques. The binary problem is the problem of distinguishing between pairs of points that belong to the same class and pairs of points that belong to different classes.[2] If we label pairs of points from the same class by 0 and pairs of points belonging to different classes by 1, we can then view the classifier's margin as the required distance function.

The training data we consider is composed of binary labels on points in $\mathcal{X} \times \mathcal{X}$. The labels describe equivalence constraints between datapoints in the original space $\mathcal{X}$. Equivalence constraints are relations between pairs of datapoints, which indicate whether the point in the pair belong to the same category or not. We term a constraint 'positive' when the points are known to be from the same class, and 'negative' in the opposite case. Such constraints carry *less* information than explicit labels of the original images in $\mathcal{X}$, since clearly equivalence constraints can be obtained from M explicit labels on points in $\mathcal{X}$, but **not** vice versa. More importantly, we observe that equivalence constraints are easier to obtain, especially when the image database is very large and contains a large number of categories without pre-defined names.

[1]A distance function is a function from pairs of datapoints to the positive real numbers, usually (but not necessarily) symmetric with respect to its arguments. We do not require that the triangle inequality holds, and thus our distance functions are *not* necessarily metrics.

[2]Note that this problem is closely related to the multi class classification problem: if we can correctly generate a binary partition of the data in product space, we implicitly define a multi-class classifier in the original vector space $\mathcal{X}$. The relations between the learnability of these two problems is discussed in [2].

To understand this observation, ask yourself how can you obtain training data for a large facial images database? You may ask a single person to label the images, but as the size of the database grows this quickly becomes impractical. Another approach is the *distributed learning* approach [9]: divide the data into small subsets of images and ask a number of people to label each subset. Note that you are still left with the problem of coordinating the labels provided by each of the labellers, since these are arbitrary. To illustrate the arbitrariness of tags, imagine a database containing all existing police facial images. While in one folder all the pictures of a certain individual may be called 'Insurance Fraud 205', different pictures of the same individual in another folder may be called 'Terrorist A'. In this distributed scenario, full labels are hard to obtain, but 'local' equivalence information can be easily gathered. [3]

Learning binary functions with margins over an input space is a well studied problem in the machine learning literature. We have explored two popular and powerful classifiers which incorporate margins: support vector machines (SVM's) and boosting algorithms. However, experiments with several SVM variants and Boosting decision trees (C4.5) have led us to recognize that the specific classification problem we are interested in has some unique features which require special treatment.

1. The product space binary function we wish to learn has some unique structure which may lead to 'unnatural' partitions of the space between the labels. The concept we learn is an indicator of an equivalence relation over the original space. Thus the properties of transitivity and symmetry of the relation place geometrical constraints on the binary hypothesis. Obviously, traditional families of hypotheses, such as linear separators or decision trees, are not limited to equivalence relation indicators, and it's not easy to enforce the constraints when such classifiers are used.

2. In the learning setting we have described above, we are provided with N datapoints in $\mathcal{X}$ and with equivalence constraints (or labels in product space) over some pairs of points in our data. We assume that the number of equivalence constraints provided is much smaller than the total number of equivalence constraints $O(N^2)$. We therefore have access to large amounts of unlabeled data, and hence semi-supervised learning seems an attractive option. However, classical SVM and boosting methods are trained using labeled data only.

[3]Inconsistencies which arise due to different definitions of distinct categories by different teachers are more fundamental, and are not addressed in this paper. Another way to solve the problem of tag arbitrariness is to use pre-defined category names, like letters or digits. Unfortunately this is not always possible, especially when the number of categories in the database is large and the specific categories are unknown apriori.

These considerations led us to the development of the *DistBoost* algorithm, which is our main contribution in this paper. *DistBoost* is a distance learning algorithm which attempts to address all of the issues discussed above. It learns a distance function which is based on boosting binary classifiers with a confidence interval in product space, using a weak learner that learns in the *original* feature space (and not in product space). We suggest a boosting scheme that incorporates unlabeled data points. These unlabeled points provide a density prior, and their weights rapidly decay during the boosting process. The weak learner we use is based on a constrained EM algorithm, which computes a Gaussian mixture model, and hence provides a partition of the original space. The constrained EM procedure uses unlabeled data and equivalence constraints to find a Gaussian mixture that complies with them. A product space hypothesis is then formed based on the computed partition.

There has been little work on learning distance functions in the machine learning literature. Most of this work has been restricted to learning Mahalanobis distance functions of the form $(x-y)^T A(x-y)$. The use of labels for the computation of the weight matrix A has been discussed in [10]; the computation of A from equivalence constraints was discussed in [14, 18]. Yianilos [19] has proposed to fit a generative Gaussian mixture model to the data, and use the probability that two points were generated by the same source as a measure of the distance between them. Schemes for incorporating unlabeled data into the boosting process were introduced by Ambroise et. al [4, 8]. We discuss the relation between these schemes and our *DistBoost* algorithm in Section 3.

We have experimented with the *DistBoost* algorithm as well as other margin based distance learning algorithms, and compared them to perviously suggested methods which are based on Mahalanobis metric learning. We used several datasets from the UCI repository [16], the yaleB facial image dataset, and a dataset of natural images obtained from a commercial image CD. The results clearly indicate that our margin based distance functions provide much better retrieval results than all other distance learning methods. Furthermore, on all these datasets the *DistBoost* method outperforms all other methods, including our earlier margin based methods which use state of the art binary classifiers.

2. Learning in the product space using traditional classifiers

We have tried to solve the distance learning problem over the product space using two of the most powerful margin based classifiers. The first is a support vector machine, that tries to find a linear separator between the data examples in a high dimensional feature space. The second is the AdaBoost algorithm, where the weak hypotheses are decision trees learnt using the C4.5 algorithm. Both algorithms had to be slightly adapted to the task of product space learning, and we have empirically tested possible adaptations using data sets from the UCI repository. Specifically, we had to deal with the following technical issues:

- Product space representation: A pair of original space points must be converted into a single which represents this pair in the product space. The simplest representation is the concatenation of the two points. Another intuitive representation is the concatenation of the sum and difference vectors of the two points. Our empirical tests indicated that while the SVM works better with the first representation, the C4.5 boosting achieves its best performance with the 'sum and difference' representation.

- Enforcing symmetry: If we want to learn a symmetric distance function satisfying $d(x, y) = d(y, x)$, we have to explicitly enforce this property. With the first representation this can be done simply by doubling the number of training points, introducing each constrained pair twice: as the point $[x, y]$ and as the point $[y, x]$. In this setting the SVM algorithm finds the global optimum of a symmetric Lagrangian and the solution is guaranteed to be symmetric. With the second represetation we found that modifying the representation to be symmetrically invariant gave the best results. Specifically, we represent a pair of points x, y using the vector $[x + y, sign(x_1 - y_1) * (x - y)]$, where x_1, y_1 are the first coordinates of the points.

- Preprocessing transformation in the original space: We considered two possible linear transformation of the data before creating the product space points: the whitening transformation, and the RCA transformation [1] which uses positive equivalence constraints. In general we found that pre-processing with RCA was most beneficial for both the SVM and C4.5 boosting algorithms.

- Parameter tuning: for the SVM we used the polynomial kernel of order 4, and a trade-off constant of 1 between error and margin. The boosting algorithm was run for 50 rounds, and the decision trees were built with a stopping criterion of train error smaller than 0.05 in each leaf.

These design issues were decided based on the performance over the UCI datasets, and all settings remained fixed for all further experiments.

3. Boosting original space partitions using *DistBoost*

Our *DistBoost* algorithm builds distance functions based on the weighted majority vote of a set of original space soft partitions. The weak learner's task in this framework is to find plausible partitions of the space, which comply with the given equivalence constraints. In this task, unlabeled data can be of considerable help, as it allows to define a prior on what are 'plausible partitions'. In order to incorporate the unlabeled data into the boosting process, we augmented an existing boosting version. The details of this augmentation are presented in Section 3.1. The details of our weak learner are presented in Section 3.2.

3.1. Semi supervised boosting in product space

Our boosting shceme is an extension of the Adaboost algorithm with confidence intervals [12] to handle unsupervised data points. As in Adaboost, we use the boosting process to maximize the margins of the labeled points. The unlabeled points only provide a decaying density prior for the weak learner. The algorithm we use is sketched in Fig. 1. Given a partially labeled dataset $\{(x_i, y_i)\}_{i=1}^N$ where $y_i \in \{1, -1, *\}$, the algorithm searches for a hypothesis $f(x) = \sum_{i=1}^{k} \alpha_k h(x)$ which minimizes the following loss function:

$$\sum_{\{i|y_i=1,-1\}} \exp(-y_i f(x_i)) \tag{1}$$

Note that the unlabeled points do not contribute to the minimization objective of the product space boosting in (1). Rather, at each boosting round they are given to the weak learner and supply it with some (hopefully useful) information regarding the domain density. The unlabeled points effectively constrain the search space during the weak learner estimation, giving priority to hypotheses which both comply with the pairwise constraints and with the density information. Since the weak learner's task becomes harder in later boosting rounds, the boosting algorithm slowly reduces the weight of the unlabeled points given to the weak learner. This is accomplished in step 4 of the algorithm (see Fig. 1).

In product space there are $O(N^2)$ unlabeled points, which correspond to all the possible pairs of original points, and the number of weights is therefore $O(N^2)$. However, the update rules for the weight of each unlabeled point are identical, and so all the unlabeled points can share the same weight. Hence the number of updates we effectively do in each round is proportional to the number of labeled pairs only. The weight of the unlabeled pairs is guaranteed to decay at least as fast as the weight of any labeled pair. This

Algorithm 1 Boosting with unlabeled data

```
Given $(x_1, y_1), ..., (x_n, y_n);\ \ x_i \in X\ ,\ y_i \in \{-1, 1, *\}$
Initialize $D_1(i) = 1/n\ \ i = 1, .., n$

For $t = 1, .., T$

    1. Train weak learner using distribution $D_t$
    2. Get weak hypothesis $h_t : X \to [-1, 1]$ with $r_t = \sum_{i=1}^{n} D_t(i) h_t(i) > 0$.
       If no such hypothesis can be found, terminate the loop and set $T = t$.
    3. Choose $\alpha_t = \frac{1}{2} \ln(\frac{1+r}{1-r})$
    4. Update:
       $D_{t+1}(i) = \begin{cases} D_t(i)\exp(-\alpha_t y_i h_t(x_i)) & y_i \in \{-1, 1\} \\ D_t(i)\exp(-\alpha_t) & y_i = * \end{cases}$
    5. Normalize: $D_{t+1}(i) = D_{t+1}(i)/Z_{t+1}$
       where $Z_{t+1} = \sum_{i=1}^{n} D_{t+1}(i)$
    6. Output the final hypothesis $f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$
```

immediately follows from the update rule in step 4 of the algorithm (Fig. 1), as each unlabeled pair is treated as a labeled pair with maximal margin of 1.

We note in passing that it is possible to incorporate unlabeled data into the boosting process itself, as has been suggested in [4, 8]. Their idea was to extend the margin concept to unlabeled data points. The algorithm then tries to minimize the total (both labeled and unlabeled) margin cost. The problem with this framework is that a hypothesis can be very certain about the classification of unlabeled points , and hence have large margins, even when it classifies these points incorrectly. Indeed, we have empirically tested some variants of these algorithms and found poor generalization performance in our context.

3.2. Mixtures of Gaussians as product space weak hypotheses

The weak learner in *DistBoost* is based on the constrained EM algorithm presented in [9]. This algorithm learns a mixture of Gaussians over the original data space, using unlabeled data and a set of positive and negative constraints. In this section we briefly review the basic algorithm, and then show how it can be extended to incorporate weights on sample data points. We describe how to translate the boosting weights from product space points to original data points, and how to generate a product space hypothesis from the soft partition found by the EM algorithm.

A Gaussian mixture model (GMM) is a parametric statistical model which assumes that the data originates from a weighted sum of several Gaussian sources. More formally, a GMM is given by $p(x|\Theta) = \Sigma_{l=1}^{M} \alpha_l p(x|\theta_l)$, where α_l denotes the weight of each Gaussian, θ_l its respective parameters, and M denotes the number of Gaussian sources in the GMM. EM is a widely used method for estimating the parameter set of the model (Θ) using unlabeled data [5]. In the constrained EM algorithm *equivalence constraints* are introduced into the 'E' (Expectation) step, such that the expectation is taken only over assignments which comply with the given constraints (instead of summing over *all* possible assignments of data points to sources).

Assume we are given a set of unlabeled i.i.d. sampled points $X = \{x_i\}_{i=1}^{N}$, and a set of pairwise constraints over these points Ω. Denote the index pairs of positively constrained points by $\{(p_j^1, p_j^2)\}_{j=1}^{N_p}$ and the index pairs of negatively constrained points by $\{(n_k^1, n_k^2)\}_{k=1}^{N_n}$. The GMM model contains a set of discrete hidden variables H, where the Gaussian source of point x_i is determined by the hidden variable h_i. The constrained EM algorithm assumes the following joint distribution of the observables X and the hiddens H:

$$p(X, H|\Theta, \Omega) = \frac{1}{Z} \prod_{i=1}^{n} \alpha_{h_i} p(x_i|\theta_{h_i}) \prod_{j=1}^{N_p} \delta_{h_{p_j^1} h_{p_j^2}} \prod_{k=1}^{N_n} (1 - \delta_{h_{n_j^1} h_{n_j^2}}) \tag{2}$$

The algorithm seeks to maximize the data likelihood, which is the marginal distribution of (2) with respect to H.

The equivalence constraints create complex dependencies between the hidden variables of different data points. However, the joint distribution can be expressed using a Markov network, as seen in Fig. 1. In the 'E' step of the algorithm the probabilities $p(h_i|X, \Theta, \Omega)$ are computed by applying a standard inference algorithm to the network. Such an inference is feasible if the number of negative constraints is $O(N)$, and the network is sparsely connected. The model parameters are then updated based on the computed probabilities. The update of the Gaussian parameters $\{\theta_l\}$ can be done in closed form, using rules similar to the standard EM update rules. The update of the cluster weights $\{\alpha_l\}_{l=1}^{M}$ is more complicated, since these parameters appear in the normalization constant Z in (2), and it requires a gradient descent procedure. The algorithm finds a local maximum of the likelihood, but the partition found is not guaranteed to satisfy any specific constraint. However, since the boosting procedure increases the weights of points which belong to unsatisfied equivalence constraints, it is most likely that any constraint will be satisfied in some partitions.

We have incorporated weights into the constrained EM procedure according to the following semantics: The algo-

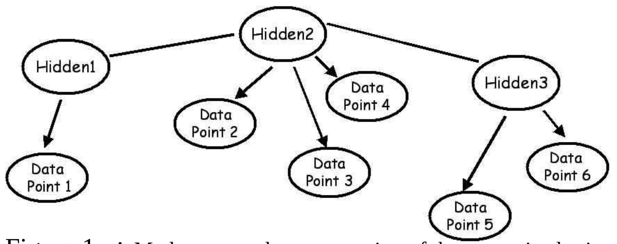

Figure 1: A Markov network representation of the constrained mixture setting. Each observable data node has a discrete hidden node as a father. Positively constrained nodes have the same hidden node as father. Negative constraints are expressed using edges between the hidden nodes of negatively constrained points.Here points 2,3,4 are known to be together, and point 1 is known to be from a different class.

rithm is presented with a virtual sample of size N_v. A training point x_i with weight w_i appears $w_i N_v$ times in this sample. All the repeated tokens of the same point are considered to be positively constrained, and are therefore assigned to the same source in every evaluation in the 'E' step. In all of our experiments we have set N_v to be the actual sample size.

While the weak learner accepts a distribution over original space points, the boosting process described in 3.1 generates a distribution over the sample product space in each round. The product space distribution is converted to a distribution over the sample points by simple summation. Denoting by w^p_{ij} the weight of pair (i, j), the weight w^s_i of point i is defined to be

$$w^s_i = \sum_j w^p_{ij} \tag{3}$$

In each round, the mixture model computed by the constrained EM is used to build a binary function over the product space and a confidence measure. We first derive a partition of the data from the Maximum A Posteriori (MAP) assignment of points. A binary product space hypothesis is then defined by giving the value 1 to pairs of points from the same Gaussian source, and -1 to pairs of points from different sources. This value determines the sign of the hypothesis output.

This setting further supports a natural confidence measure - the probability of the pair's MAP assignment which is:

$$\max_i p(h_1 = i|x_1, \Theta) \cdot \max_i p(h_2 = i|x_2, \Theta)$$

where h_1, h_2 are the hidden variables attached to the two points. The weak hypothesis output is the signed confidence measure in $[-1, 1]$, and so the weak hypothesis can be viewed as a 'weak distance function'.

4. Learning distance functions: comparative results

In this section we compare our *DistBoost* algorithm with other distance learning techniques, including our two other proposed methods for learning in product space (SVM and boosting decision trees). We begin by introducing our experimental setup. We then show results on several datasets from the UCI repository, which serve as benchmark to evaluate the different distance learning methods.

4.1. Experimental setup

Gathering equivalence constraints: We simulated a *distributed learning* scenario [9], where labels are provided by a number of uncoordinated independent teachers. Accordingly, we randomly chose small subsets of data points from the dataset and partitioned each of the subsets into equivalence classes.

The size of each subset k in these experiments was chosen to be $2M$, where M is the number of classes in the data. In each experiment we used l subsets, and the amount of partial information was controlled by the *constraint index* $P = k \cdot l$; this index measures the amount of points which participate in at least one constraint. In our experiments we used $P = 0.3, 0.5$. However, it is important to note that the number of equivalence constraints thus provided typically includes only a small subset of all possible pairs of datapoints, which is $\mathcal{O}(N^2)$.[4]

Evaluated Methods: we compared the performance of the following distance learning methods:

- Our proposed *DistBoost* algorithm.
- Mahalanobis distance learning with Relevant Component Analysis (RCA) [1].
- Mahalanobis distance learning with non-linear optimization [18].
- SVM for direct discrimination in product space.
- Boosting decision trees in product space.

In order to set a lower bound on performance, we also compared with the whitened Mahalanobis distance, where the weight matrix A is taken to be the data's global covariance matrix.

We present our results using ROC curves and *cumulative neighbor purity* graphes. *Cumulative neighbor purity* measures the percentage of correct neighbors up to the Kth neighbor, averaged over all the queries. In each experiment we averaged the results over 50 different equivalence constraint realizations. Both *DistBoost* and the decision tree boosting algorithms were run for a constant number of boosting iterations $T = 50$. In each realization all the algorithms were given the exact same equivalence constraints.

[4]It can be readily shown that by wisely selecting $\mathcal{O}(NM)$ equivalence constraints, one can label the entire dataset. This follows from the transitive nature of positive equivalence constraints.

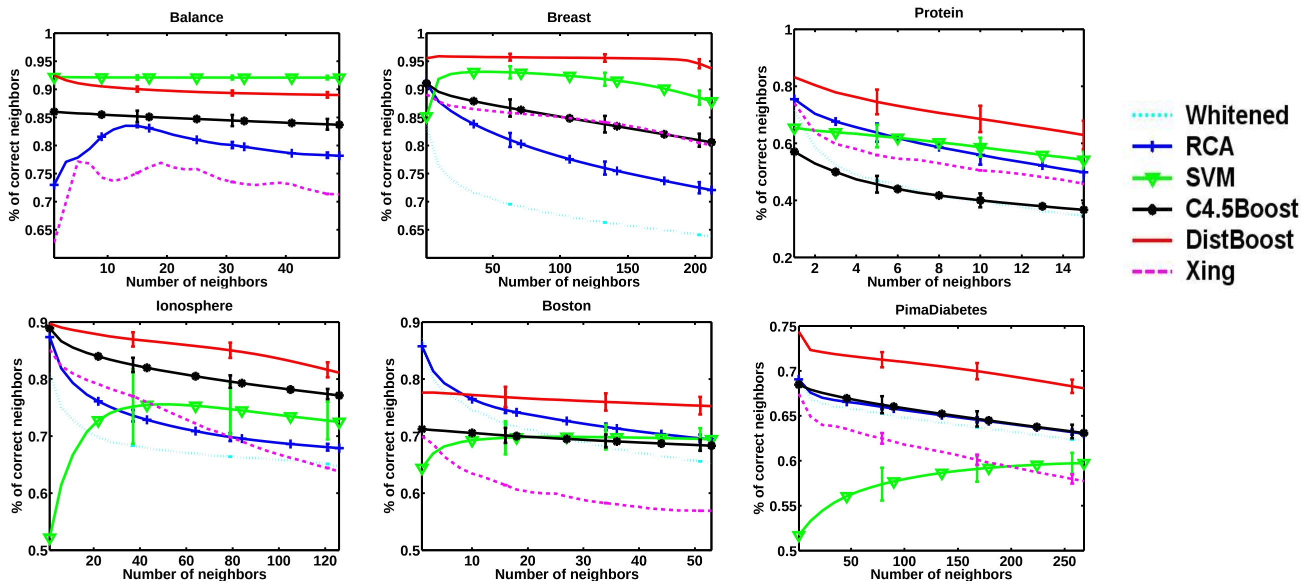

Figure 2: Cumulative neighbor purity plots over 6 data sets from the UCI repository. The UCI results were averaged over 50 realizations of constraints, and 1-std error bars are shown. The percentage of data in constraints was 50% in all cases.

4.2. Results on UCI datasets

We selected several standard datasets from the UCI data repository and used the experimental setup above to evaluate our proposed methods. The comulative purity was computed using all the points in the data as queries.

Fig. 2 shows neighbor purity plots for each of these data sets. As can be readily seen, *DistBoost* achieves significant improvements over Mahalanobis based distance measures, and also outperforms all other product space learning methods (except SVM in the 'balance' dataset).

5. Experiments on image retrieval

We ran experiments on two image retrieval tasks: facial image retrieval using the YaleB dataset, and color based image retrieval using pictures from a commercial image CD. The evaluated methods are described in Section 4.1.

In our experiments we randomly selected from each dataset a subset of images, to be the retrieval database, and this subset was used as the training set. We then followed the same experimental setup of distributed learning (described in Section 4.1) for the generation of equivalence constraints, and trained all methods on the selected data. Retrieval performance was measured using test images which were not presented during training.

5.1 Facial image retrieval - YaleB

As an image retrieval example with known ground-truth and a clear definition of categories, we used a subset of the YaleB facial image database [7]. The dataset contains a total of 1920 images, including 64 frontal pose images of 30 different subjects. In this database the variability between images of the same person is mainly due to different lighting conditions. We automatically centered all the images using optical flow. Images were then converted to vectors, and each image was represented using its first 60 PCA coefficients.

¿From each class, we used 22 images (a third) as a data base training set, and 42 images were used as test queries. In order to check different types of generalization, we used a slightly modified policy for constraint sampling. specifically, constraints were drawn from 20 out of the 30 classes in the dataset, and in the constrained classes p was set to 1 (which means that all the training points in these classes were divided between uncoordinated labellers). When testing the learnt distance functions measurements were done separately for test images from the first 20 classes and for the last 10. Notice that in this scenario images from the 10 unconstrained classes were not helpful in any way to the traditional algorithms, but they were used by *DistBoost* as unlabeled data. On the left in Fig. 3 we present the ROC curves of the different methods on test data from the constrained classes. We can see that the margin based distance functions give very good results, indicating an adaptation of the distance function to these classes. On the right we present the ROC curves when the queries are from unconstrained classes. It can be seen that the performance of SVM and C4.5Boost severely degrades, indicating strong overfit behavior. The *DistBoost*, on the other hand, degrades gracefully and is still better then the other alternatives.

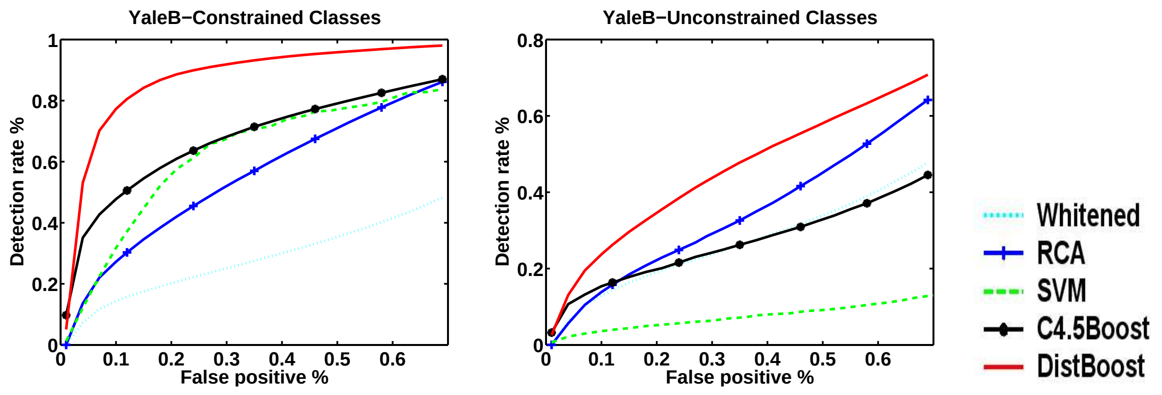

Figure 3: ROC curves of different methods on the YaleB facial image database. Left: retrieval performance on classes on which constraints we are provided. Right: retrieval performance on classes on which no constraints were provided. Results were averaged over 80 realizations

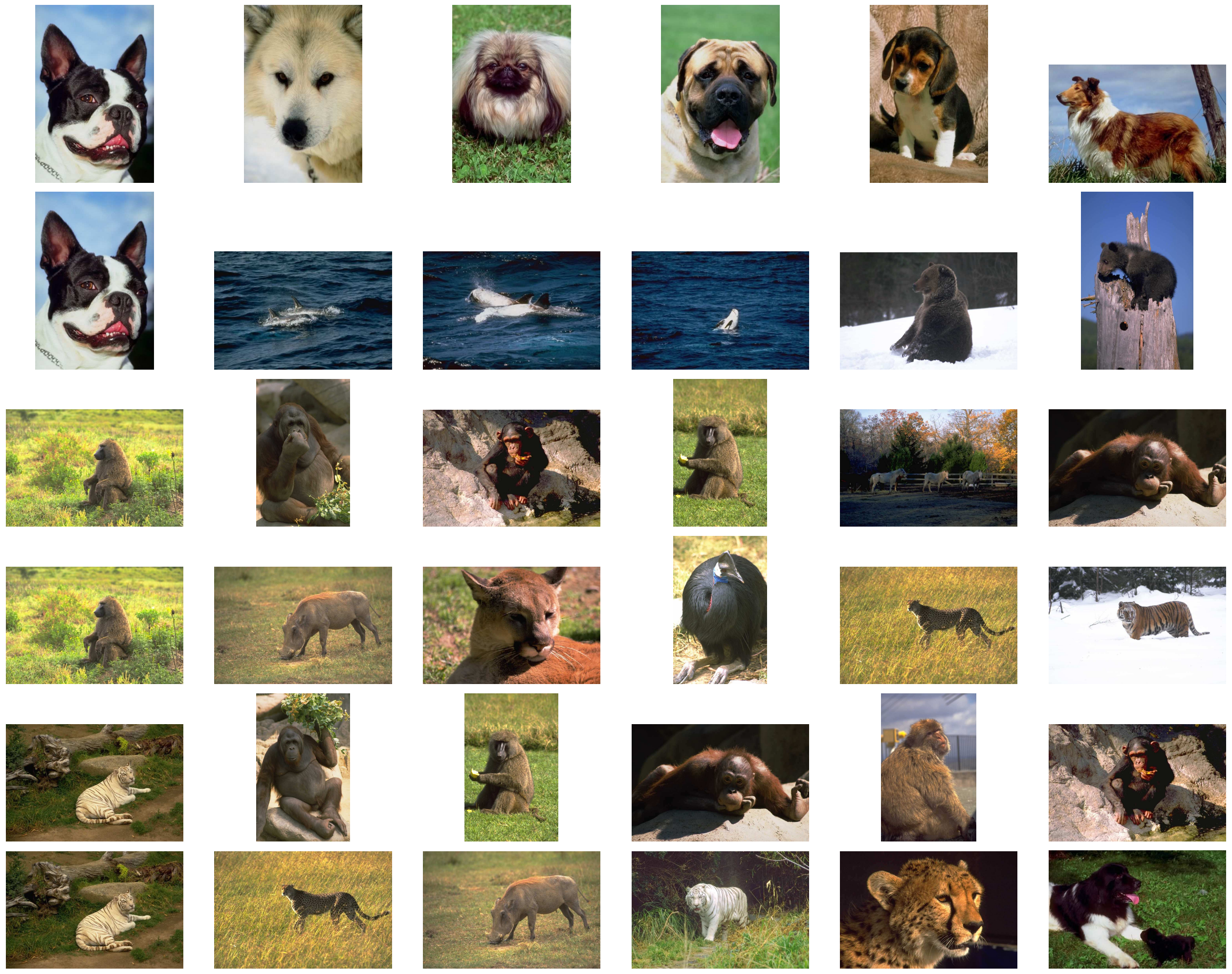

Figure 4: Typical retrieval results on the Animal image database. Each row presents a query image and its first 5 nearest neighbors comparing DistBoost and normalized $L1$ CCV distance (baseline measure). Results appear in pairs of rows: Top row: DistBoost results, Bottom row: normalized $L1$ CCV distance. Results are best seen in color.

5.2 Color based image retrieval

We created a picture database which contained images from 16 animal classes taken from a commercial image CD. The retrieval task in this case is much harder then in the facial retrieval application. We used 70% of the data from each class as our dataset (training data), and the remaining 30% as our test data. We experimented with two scenarios vary-

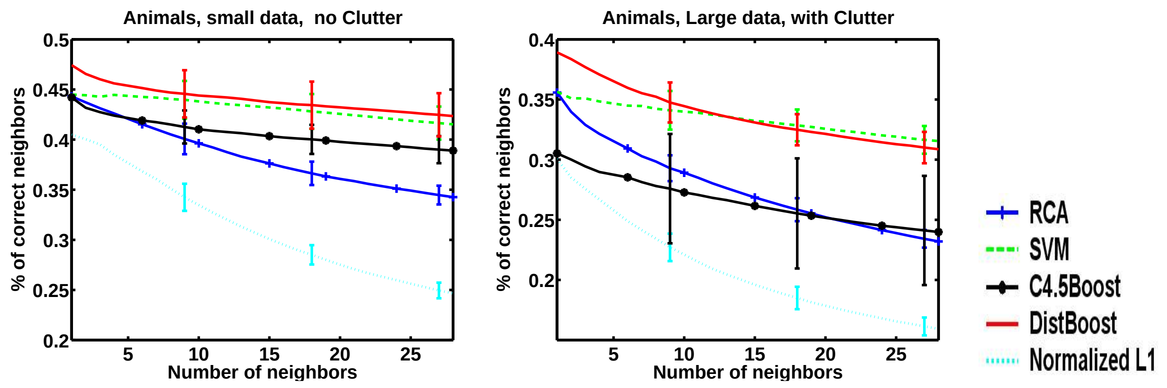

Figure 5: Neighbor purity results on color animal database. Left: results when on a database of 10 classes, 405 images and no clutter. Right: results with 16 classes, 565 images and 600 clutter images. The clutter was added to the database after the training stage. Results were averaged over 50 realizations

ing in their difficulty level. In the first scenario we used 10 classes with a total of 405 images. In the second scenario the database contained 16 classes with 565 images, and 600 'clutter' images from unrelated classes were added to the data base. The clutter included non-animal categories, such as 'landscapes' and 'buildings'.

The original images were heavily compressed jpg images. The images were represented using Color Coherence Vectors [11] (CCV's). This representation extend the color histogram, by capturing some crude spatial properties of the color distribution in an image. Specifically, in a CC vector each histogram bin is divided into two bins, representing the number of 'Coherent' and 'Non-Coherent' pixels from each color. 'Coherent' pixels are pixels whose neighborhood contains more than τ neighbors which have the same color. Following [11] we represented the images in HSV color space quantized the images to $4*2*4 = 32$ color bins, and computed the CCV of each image - a 64 dimensional vector - using $\tau = 25$.

Fig. 5 shows neighbor purity plots of all different distance learning methods. As our baseline measure, we used the normalized $L1$ distance measure suggested in [11]. Clearly the *DistBoost* algorithm and our product space SVM methods outperformed all other distance learning methods. The C4.5Boost performs less well, and it suffers from a relatively high degradation in performance when the task becomes harder. Retrieval results are presented in Fig 4 for our proposed *DistBoost* algorithm (Top row) and for the baseline normalized $L1$ distance over CCV's (bottom row). As can be seen our algorithm seems to group images which do not appear trivially similar in CCV space.

References

[1] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall. Learning Distance Functions using Equivalence Relations. In Proc. of ICML, 2003.

[2] A. Bar-hillel , D. Weinshall Learning with Equivalence Constraints, and the relation to Multiclass Classification In The Sixteenth Annual Conference on Learning Theory (COLT) , 2003

[3] Cox, I.J., Miller, M.L., Minka, T.P., Papathornas, T.V., Yianilos, P.N. The Bayesian Image Retrieval System, PicHunter: Theory, Implementation, and Psychophysical Experiments. In IEEE Tran. On Image Processing, Volume 9, Issue 1, pp. 20-37, Jan. 2000.

[4] F. d'Alche-Bue, Y. Grandvalet, and C. Ambroise. Semi supervised margin boost. in Proc. of Nips, 2001.

[5] A. P. Dempster, N. M. Laird and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. In *J. Royal Statistical Society*, B, 39, 1-38, 1977.

[6] K. Fukunaga. Introduction to statistical pattern recognition. Academic press, 1990.

[7] Georghiades, A.S. and Belhumeur, P.N. and Kriegman, D.J., From Few To Many: Generative Models For Recognition Under Variable Pose and Illumination", IEEE Int. Conf. on Automatic Face and Gesture Recognition, page 277-284, 2000.

[8] Y. Grandvalet, F. d'Alche-Bue, and C. Ambroise. Boosting mixture models for semi supervised learning in ICANN 2001,Vienne, Austria, 41-48, Springer 2001.

[9] T. Hertz, N. Shental, A. Bar-Hillel, and D. Weinshall. Enhancing Image and Video Retrieval: Learning via Equivalence Constraints. In Proc. of CVPR, 2003.

[10] D. G. Lowe. Simlarity metric learning for a variable-kernel classifier. Neural Computation 7:72-85, 1995.

[11] G. Pass, R. Zabih, and J. Miller. Comparing Images Using Color Coherence Vectors. In *ACM Multimedia*, 65-73, 1996.

[12] R. E. Schapire and Y. Singer Improved Boosting Algorithms using Confidence-Rated Predictions. In proc. of COLT , 80-91, 1998.

[13] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee Boosting the margin: a new explanation for the effectiveness of voting methods. Proc. of 14th International Conference on Machine Learning,322-330, 1997.

[14] N. Shental, T. Hertz, D. Weinshall, and M. Pavel. Adjustment learning and relevant component analysis. In *Proc. of ECCV*, Copenhagen, 2002.

[15] K. Tieu and P. Viola. Boosting image retrieval. In Proc. of CVPR, 2000.

[16] http://www.ics.uci.edu/ mlearn/MLRepository.html

[17] V. Vapnik Statistical learning theory Wilay, Chichester, GB 1998.

[18] E. P. Xing, A. Y. Ng, M. I. Jordan and S. Russell. Distance Metric learnign with application to clustering with side-information. In *Proc. of NIPS*, 2002.

[19] P. N. Yianilos. Metric learning via normal mixtures. *NECRI TR*, 1995.