Locally Smooth Metric Learning with Application to Image Retrieval*

Hong Chang
Xerox Research Centre Europe
6 chemin de Maupertuis, Meylan, France
hong.chang@xrce.xerox.com

Dit-Yan Yeung
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
dyyeung@cse.ust.hk

Abstract

In this paper, we propose a novel metric learning method based on regularized moving least squares. Unlike most previous metric learning methods which learn a global Mahalanobis distance, we define locally smooth metrics using local affine transformations which are more flexible. The data set after metric learning can preserve the original topological structures. Moreover, our method is fairly efficient and may be used as a preprocessing step for various subsequent learning tasks, including classification, clustering, and nonlinear dimensionality reduction. In particular, we demonstrate that our method can boost the performance of content-based image retrieval (CBIR) tasks. Experimental results provide empirical evidence for the effectiveness of our approach.

1. Introduction

Metric learning plays a crucial role in machine learning research, as many machine learning algorithms are based on some distance metrics. Metric learning for classification tasks has a long history that can be dated back to some early work for nearest neighbor classifiers more than two decades ago [12]. More recent work includes learning locally adaptive metrics [3, 4, 6] and global Mahalanobis metrics [5, 13]. Metric learning methods have also been proposed for semi-supervised clustering tasks, with supervisory information available in the form of either limited labeled data [15] or pairwise constraints [1, 2, 14]. Instead of devising metric learning methods for specific classification or clustering algorithms, it is also possible to devise generic metric learning methods that can be used for both classification and clustering tasks, as in [15].

An advantage of many metric learning methods that learn global Mahalanobis metrics is that learning can often be formulated as convex optimization problems with no local optima and they can be solved using efficient algorithms. For example, Xing et al. [14] formulated semi-supervised clustering with pairwise constraints as a convex optimization problem. Bar-Hillel et al. [1] proposed a simpler, more efficient method called *relevant component analysis* (RCA) to solve essentially the same problem. More recently, Weinberger et al. [13] proposed a metric learning method for nearest neighbor classifiers, called *large margin nearest neighbor* (LMNN) classification, that is based on a margin maximization principle similar to that for support vector machines (SVM). The metric learning problem can be formulated as a semi-definite programming (SDP) problem. Although local information is used for metric learning in LMNN making the problem more tractable, the Mahalanobis metric learned is still global in the sense that the same linear transformation is applied to all data points globally. For many real-world applications, however, globally linear transformations are inadequate. One possible extension is to perform locally linear transformations that are globally nonlinear, so that different local metrics are applied at different locations of the input space. For example, Chang and Yeung [2] explored this possibility for semi-supervised clustering tasks. However, the optimization problem suffers from local optima and the intra-class topological structure of the input data cannot be preserved well during metric learning. Other algorithms that exploit similar local transformation ideas also experience similar computational difficulties.

In this paper, our goal is to get the best of both worlds. Specifically, we want to devise a more flexible metric learning method that can perform locally linear transformations, yet it is as efficient as algorithms for learning globally linear metrics. We refer to our method as *locally smooth metric learning* (LSML), which is inspired by the method of moving least squares for function approximation from pointwise scattered data [8]. However, the optimization problem for LSML is significantly different from that for previous methods, because we formulate the problem under a regularization framework with which unlabeled data can also play a role. This is particularly crucial when labeled data are

*This research has been supported by research grants HKUST621305 and N-HKUST602/05 from the Research Grants Council (RGC) of Hong Kong and the National Natural Science Foundation of China (NSFC).

scarce. The advantages of LSML are summarized as follows: (1) It allows different local metrics to be learned at different locations of the input space. (2) The local metrics vary smoothly over the input space so that the intra-class topological structure of the data can be preserved during metric learning. (3) It is as efficient as global linear metric learning methods. (4) It is a generic metric learning method that can be used for various semi-supervised learning tasks, including classification, clustering, and nonlinear dimensionality reduction.

The rest of this paper is organized as follows. We present our LSML method in Section 2. We formulate the optimization problem as solving a set of regularized moving least squares problems and then propose an efficient method to solve them. Section 3 presents some experimental results for different semi-supervised learning tasks. We then apply our metric learning method to content-based image retrieval (CBIR) in Section 4. Finally, Section 5 gives some concluding remarks.

2. Our Method

2.1. Locally Smooth Metrics

Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_l, \mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$ denote n data points in a d-dimensional input space $\mathcal{R}^d$. For semi-supervised learning, we assume that supervisory side information is available for the first l data points. The side information, represented by a set $\mathcal{S}$, can be in the form of labels for data points or pairwise constraints between data points. Intuitively, we want to perform local transformations on the data points such that the points that belong to the same class or are considered similar according to the similarity constraints will get closer after they are transformed. Similar to [2], we resort to locally linear transformations. More specifically, for each data point $\mathbf{x}_i \in \mathcal{X} \subset \mathcal{R}^d$, we define the corresponding locally linear transformation as:

$$\mathbf{f}_i(\mathbf{x}) = \mathbf{A}_i\mathbf{x} + \mathbf{b}_i, \tag{1}$$

where $\mathbf{A}_i$ denotes a $d \times d$ transformation matrix and $\mathbf{b}_i$ a d-dimensional translation vector. After metric learning, $\mathbf{x}$ is transformed to $\mathbf{f}_i(\mathbf{x})$.

Note that a different locally linear transformation $\mathbf{f}_i(\cdot\,; \mathbf{A}_i, \mathbf{b}_i)$ is associated with each input data point $\mathbf{x}_i$. It would be desirable if $\mathbf{f}_i(\cdot\,; \mathbf{A}_i, \mathbf{b}_i)$ changes smoothly in the input space so that the transformed data points after metric learning can preserve the intra-class topological structure of the original data. Since there are n locally linear transformations, one for each input data point, we have a large number of parameters to determine. In the next two subsections, we propose an efficient method for solving this problem.

2.2. Regularized Moving Least Squares

To compute the locally linear transformations for all input data points, we formulate metric learning as a set of n optimization problems. We extend the method of moving least squares [8] to *regularized moving least squares* under a regularization framework. For each input data point $\mathbf{x}_i$, we compute its local affine transformation by minimizing the following objective function:

$$J(\mathbf{A}_i, \mathbf{b}_i) = \sum_{j=1}^{l} \theta_{ij} \|\mathbf{f}_i(\mathbf{x}_j) - \mathbf{y}_j\|^2 + \lambda \sum_{j=1}^{n} w_{ij} \|\mathbf{f}_i(\mathbf{x}_i) - \mathbf{f}_j(\mathbf{x}_j)\|^2. \tag{2}$$

The first term in the objective function is the moving least squares term. As in [11], we define the moving weight function that depends on both $\mathbf{x}_i$ and $\mathbf{x}_j$ as $\theta_{ij} = 1/\|\mathbf{x}_i - \mathbf{x}_j\|^2$. For the first l data points, the target location of $\mathbf{x}_j$ after metric learning is denoted as $\mathbf{y}_j$ which can be computed from the input data and the side information. Although it has been demonstrated that the method of moving least squares works very well for interpolation and the learned functions $\mathbf{f}_i(\cdot\,; \mathbf{A}_i, \mathbf{b}_i)$ vary smoothly [8], we notice that performance degrades when the data points with side information are not evenly distributed in the input space. The second term in Equation (2) is to address this problem through regularization by incorporating unlabeled data. It restricts the degree of local transformation to preserve local neighborhood relationships. Like graph-based semi-supervised learning methods [16], we consider the n points as corresponding to a graph consisting of n nodes, with a weight between every two nodes. Since the weights w_{ij} are similar to the edge weights in graph-based methods, we also define w_{ij} as $w_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2)$ with $\sigma > 0$ specifying the spread. To combine the two terms, the parameter $\lambda > 0$ controls the relative contribution of the regularization term in the objective function.

The target points $\mathbf{y}_j$ $(j = 1, \ldots, l)$ are set differently for different learning tasks. For classification tasks with partial label information, one possibility is to set $\mathbf{y}_j$ to the mean of all $\mathbf{x}_k$'s with the same label as $\mathbf{x}_j$.For clustering tasks with some given pairwise similarity constraints, we may set $\mathbf{y}_j$ by pulling similar points towards each other. We will show some examples on different learning tasks in the next section.

2.3. Optimizing the Objective Functions

Note that the objective function $J(\mathbf{A}_i, \mathbf{b}_i)$ for the ith local transformation involves the parameters of all n local transformations, $\mathbf{f}_j(\cdot\,; \mathbf{A}_j, \mathbf{b}_j)$ $(j = 1, \ldots, n)$. Since $J(\mathbf{A}_i, \mathbf{b}_i)$ is quadratic in $\mathbf{A}_i$ and $\mathbf{b}_i$, in principle it is possible to obtain a closed-form solution for the parameters of all n transformations by solving a set of n equations. This approach is undesirable, though, as it requires inverting a pos-

sibly large $n \times n$ matrix. We propose here a more efficient alternative approach for obtaining an approximate solution.

More specifically, we estimate $\mathbf{f}_i(\cdot\,; \mathbf{A}_i, \mathbf{b}_i)$ by minimizing a modified form of Equation (2), with the regularization term replaced by one that incorporates only the $i-1$ local transformations already estimated in the previous steps. We first estimate the local transformations for the first l data points with side information available. For the remaining $n-l$ local transformations, we order them such that the transformations for the data points closer to the first l points are estimated before those that are farther away. This may be seen as a process of propagating the changes from the labeled points to the unlabeled points. Let $\hat{\mathbf{y}}_j = \hat{\mathbf{f}}_j(\mathbf{x}_j) = \hat{\mathbf{A}}_j\mathbf{x}_j + \hat{\mathbf{b}}_j$ denote the new location of $\mathbf{x}_j$ after transformation. Substituting Equation (1) into the modified form of Equation (2), we have the following approximate objective function:

$$J(\mathbf{A}_i, \mathbf{b}_i) = \sum_{j=1}^{l} \theta_{ij} \|\mathbf{A}_i\mathbf{x}_j + \mathbf{b}_i - \mathbf{y}_j\|^2 + \lambda \sum_{j=1}^{i-1} w_{ij} \|\mathbf{A}_i\mathbf{x}_i + \mathbf{b}_i - \hat{\mathbf{y}}_j\|^2. \quad (3)$$

We can obtain a closed-form solution for $\hat{\mathbf{A}}_i$ and $\hat{\mathbf{b}}_i$ as:

$$\hat{\mathbf{A}}_i = (\mathbf{D}_i - \bar{\mathbf{y}}_i\bar{\mathbf{x}}_i^T)(\mathbf{C}_i - \bar{\mathbf{x}}_i\bar{\mathbf{x}}_i^T)^+, \quad (4)$$

$$\hat{\mathbf{b}}_i = \bar{\mathbf{y}}_i - \mathbf{A}_i\bar{\mathbf{x}}_i, \quad (5)$$

where $^+$ denotes the pseudoinverse and $\bar{\mathbf{x}}_i$, $\bar{\mathbf{y}}_i$, $\mathbf{C}_i$ and $\mathbf{D}_i$ are defined as:

$$\bar{\mathbf{x}}_i = \frac{\sum_{j=1}^{l} \theta_{ij}\mathbf{x}_j + \lambda \sum_{j=1}^{i-1} w_{ij}\mathbf{x}_i}{\sum_{i=1}^{l} \theta_{ij} + \lambda \sum_{j=1}^{i-1} w_{ij}},$$

$$\bar{\mathbf{y}}_i = \frac{\sum_{ji=1}^{l} \theta_{ij}\mathbf{y}_j + \lambda \sum_{j=1}^{i-1} w_{ij}\hat{\mathbf{y}}_i}{\sum_{i=1}^{l} \theta_{ij} + \lambda \sum_{j=1}^{i-1} w_{ij}},$$

$$\mathbf{C}_i = \frac{\sum_{j=1}^{l} \theta_{ij}\mathbf{x}_j\mathbf{x}_j^T + \lambda \sum_{j=1}^{i-1} w_{ij}\mathbf{x}_i\mathbf{x}_i^T}{\sum_{i=1}^{l} \theta_{ij} + \lambda \sum_{j=1}^{i-1} w_{ij}},$$

$$\mathbf{D}_i = \frac{\sum_{j=1}^{l} \theta_{ij}\mathbf{y}_j\mathbf{x}_j^T + \lambda \sum_{j=1}^{i-1} w_{ij}\hat{\mathbf{y}}_j\mathbf{x}_i^T}{\sum_{i=1}^{l} \theta_{ij} + \lambda \sum_{j=1}^{i-1} w_{ij}}.$$

For each local affine transformation, the main computation is to find the pseudoinverse of a $d \times d$ matrix. Thus the algorithm is very efficient as long as d is not exceptionally large.

2.4. Algorithm and an Illustrative Example

We summarize the LSML algorithm as follows:

Input: data set $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, side information $\mathcal{S}$;
Compute target points: $\mathbf{y}_i$ $(i = 1, \ldots, l)$;
Metric learning:
  sort the data points in $\{\mathbf{x}_{l+1}, \ldots, \mathbf{x}_n\}$;
  For $i = 1$ to n **do**
    compute local affine transformation using (4), (5);
  End
Output: $\hat{\mathbf{A}}_i, \hat{\mathbf{b}}_i$ $(i = 1, \ldots, n)$.

There are only two free parameters in the LSML algorithm, σ and λ. We set σ^2 to be the average squared Euclidean distance between nearest neighbors in the input space and λ to some value in $(0, 1]$ which can be determined by cross-validation.

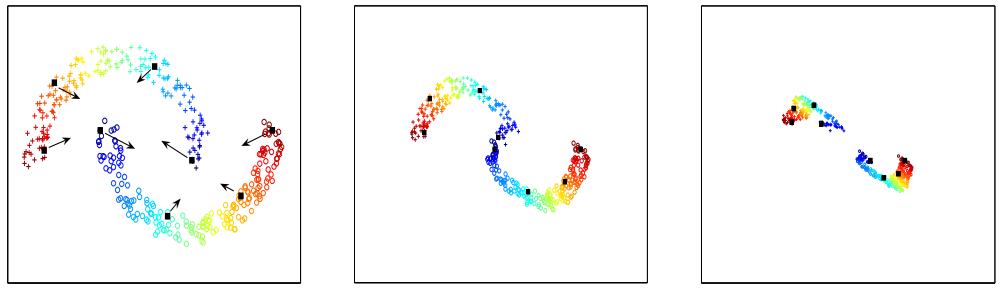
Let us first look at an illustrative example using our proposed LSML method on the well-known 2-moon data set, as shown in Figure 1(a). The solid black squares represent the labeled data points. We set the goal of metric learning as moving the points with the same label towards their center (as indicated by the arrows) while preserving the moon structure. We first compute the locations of the target points, as shown with black squares in Figure 1(b) and (c). For illustration purpose, we set the target locations to be some points along the way to the class centers. In Figure 1(b), the data set after metric learning has shorter moon arms. If the target points are closer to the class centers, the data set after metric learning forms more compact classes, as shown in Figure 1(c), which are desirable for classification and clustering tasks. From the color code, we can see that the intra-class topological structure is well preserved after metric learning, showing that the learned functions $\mathbf{f}_i(\cdot\,; \mathbf{A}_i, \mathbf{b}_i)$ $(i = 1, \ldots, n)$ vary smoothly along the arms of the moons. Figure 1(d) and (e) demonstrate the effect of the regularization term when only a single labeled point is available for each class. The data set after metric learning with $\lambda = 0$ (without regularization) is shown in Figure 1(e) and that with $\lambda = 0.5$ in Figure 1(f). As we can see, the regularization term plays an important role in preserving the structure of the data especially when the side information available is scarce and unevenly distributed.

3. Experiments on Semi-Supervised Learning Tasks

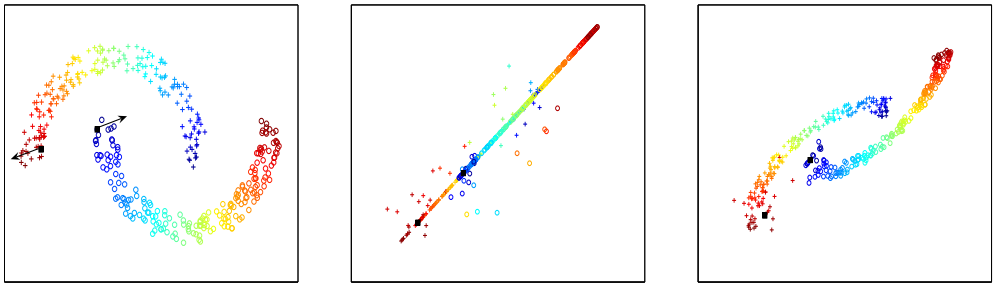
In this section, we illustrate through experiments how the proposed LSML method can be used for different semi-supervised learning tasks, including classification and clustering. The quality of metric learning is assessed indirectly via the performance of the respective learning tasks.

3.1. Semi-Supervised Nearest Neighbor Classification

We use LSML with a nearest neighbor classifier and compare its performance with that of LMNN [13], which learns a global Mahalanobis metric for nearest neighbor



(a) Original data set (b) Transformed data (c) Transformed data



(d) Original data set (e) Transformed data (f) Transformed data

Figure 1. Locally smooth metric learning on the 2-moon data set. (a) and (d) original data set; (b), (c), (e) and (f) transformed data after metric learning.

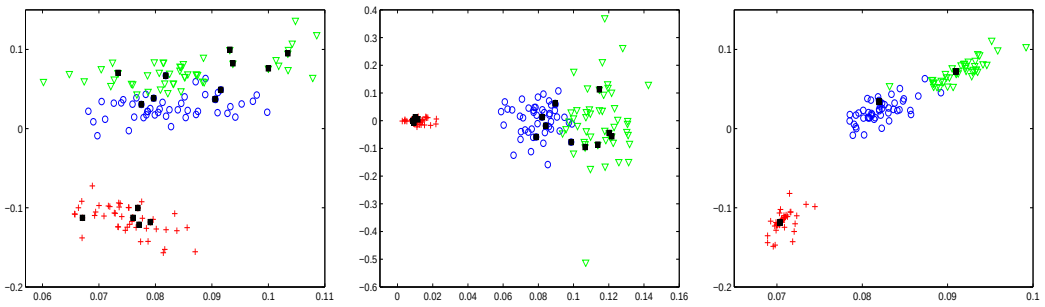
classification with the goal that data points from different classes are separated by a large margin. The MATLAB code for LMNN is provided by its authors.

We first perform some experiments on the Iris plant data set from the UCI Machine Learning Repository. There are 150 data points from three classes with 50 points per class. The dimensionality of the input space is 4. For visualization, we plot the data points in a 2-D space before and after metric learning based on the two leading principal components of the data, as shown in Figure 2. Figure 2(a) shows the original data set. Data points with the same point style and color belong to the same class. As we can see, one class (marked with red '+') is well separated from the other two classes which are very close to each other. Similar to the 2-moon data set in Figure 1, the labeled points are shown as solid black squares. Figure 2(b) shows the metric learning result obtained by LMNN. We can see that there is less overlap between the two nearby classes. This is expected to lead to better classification results. Using LSML, Figure 2(c) shows even higher separability between the two nearby classes. The data points with the same class label have been transformed to the same location. Using 20 randomly generated training sets with each set having five labeled points per class, the average classification rates for a 3-nearest neighbor classifier are 91.52% for LMNN and 94.63% for LSML. With the training sets increased in size to 10 points per class, the average classification rates for LMNN and LSML increase to 95.83% and 95.08%, respectively. This shows that LSML is particularly good when labeled data are scarce.

We further perform more classification experiments on two larger data sets. One data set contains handwritten digits from the MNIST database.[1] The digits in the database have been size-normalized and centered to 28×28 gray-level images. In our experiments, we randomly choose 2,000 images for digits '0'–'4' from a total of 60,000 digit images in the MNIST training set. Another data set is the Isolet data set from the UCI Machine Learning Repository, which contains 7,797 isolated spoken English letters belonging to 26 classes with each letter represented as a 617-dimensional vector. For both data sets, since the original features are highly redundant, we first reduce the dimensionality by performing principal component analysis (PCA) to keep the first 100 principal components.

The 3-nearest neighbor classification results based on different metrics are summarized in Table 1 below. For each metric and training set size, we show the mean classification rate and standard deviation over 10 random runs corresponding to different randomly generated training sets. We can see that both LMNN and LSML are significantly better than the Euclidean metric, with LSML being slightly better.

[1] http://yann.lecun.com/exdb/mnist/



(a) Iris plant data set (b) LMNN result (c) LSML result

Figure 2. Metric learning using the Iris plant data set. (a) 4-D data set projected onto a 2-D space; data set after metric learning based on (b) LMNN and (c) LSML.

Table 1. 3-nearest neighbor classification results based on three different distance metrics.

	MNIST		Isolet	
% labeled data	10%	20%	10%	20%
Euclidean	0.3770 ($\pm$0.030)	0.5095 ($\pm$0.006)	0.7054 ($\pm$0.008)	0.7629 ($\pm$0.005)
LMNN	0.8711 ($\pm$0.009)	0.9270 ($\pm$0.004)	0.9111 ($\pm$0.005)	0.9337 ($\pm$0.006)
LSML	0.8959 ($\pm$0.010)	0.9315 ($\pm$0.014)	0.9336 ($\pm$0.011)	0.9480 ($\pm$0.006)

3.2. Semi-Supervised Clustering with Pairwise Similarity Constraints

We next perform some experiments on semi-supervised clustering with pairwise similarity constraints as side information in $\mathcal{S}$. We compare LSML with two previous metric learning methods for semi-supervised clustering: a globally linear method called RCA [1] and a locally linear but globally nonlinear method called *locally linear metric adaptation* (LLMA) [2]. Euclidean distance without metric learning is used for baseline comparison. These four metrics are used with k-means clustering and their performance is compared.[2] As in [1, 2, 14], we use the Rand index [10] as the clustering performance measure. For each data set, we randomly generate 20 different $\mathcal{S}$ sets for the similarity constraints and report the average Rand index over 20 runs.

We use six data sets from the UCI Machine Learning Repository: Protein $(116/20/6/15)$, Iris plants $(150/4/3/30)$, Wine $(178/13/3/20)$, Ionosphere $(351/34/2/30)$, Boston housing $(506/13/3/40)$, and Balance $(625/4/3/40)$. The numbers inside the brackets $(n/d/c/t)$ summarize the characteristics of the data sets, including the number of data points n, number of features d, number of clusters c, and number of randomly selected point pairs for the similarity constraints.

Figure 3 shows the clustering results using k-means with different distance metrics. From the results, we can see that LLMA and LSML, as nonlinear metric learning methods, generally outperform RCA. Moreover, LSML is comparable to or even better than LLMA.

Besides semi-supervised classification and clustering, LSML can also be used for semi-supervised embedding tasks, where we directly specify the target point $\mathbf{y}_j$ as the location of $\mathbf{x}_j$ in some low-dimensional space. We have obtained some promising results for such embedding tasks, showing that LSML can also be extended for visualization and other machine learning applications. Due to space limitation, however, such experimental results are omitted in this paper.

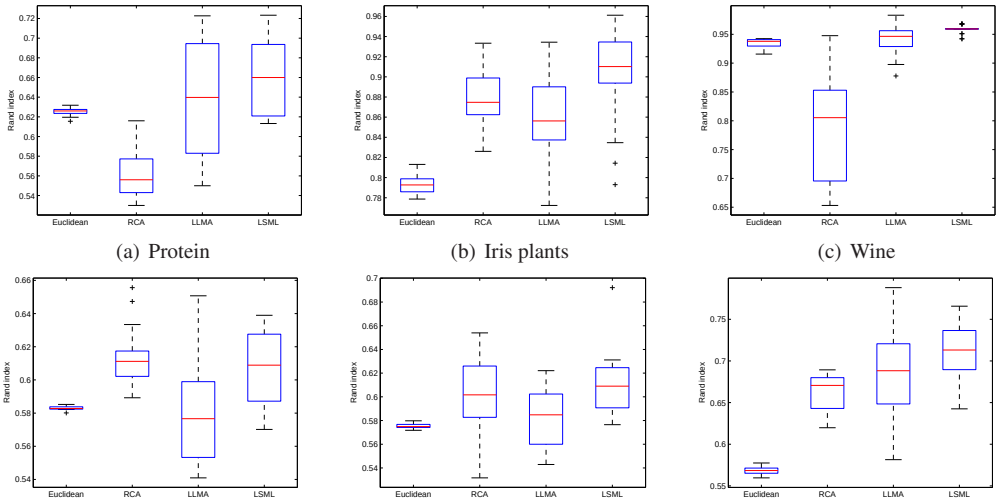
3.3. Efficiency

Besides the promising performance of LSML for different semi-supervised learning tasks, our method also has the additional advantage of being fairly efficient. As discussed in Section 2.3, each local affine transformation can be obtained directly from Equations (4) and (5). In the experiments we have performed, LSML is much faster than LMNN [13], which solves an SDP problem based on gradient descent, and LLMA [2], which optimizes a non-convex objective function using an iterative algorithm.

4. Experiments on Image Retrieval

For a specific feature representation, a good similarity measure can improve the performance of image retrieval tasks. Recently, learning distance metrics for image retrieval has aroused great interest in the research community. In particular, RCA and LLMA have been applied to boost

[2]The MATLAB code for RCA and LLMA was obtained from the authors of [1] and [2], respectively.



(a) Protein (b) Iris plants (c) Wine

(d) Ionosphere (e) Boston housing (f) Balance

Figure 3. Semi-supervised clustering results on six UCI data sets based on different distance metrics.

the image retrieval performance of CBIR tasks. Besides, a nonmetric distance learning algorithm called DistBoost [7], which makes use of pairwise contraints and performs boosting, has also demonstrated very good image retrieval results in CBIR tasks.

In this section, we will apply our LSML algorithm to CBIR and compare it with RCA, LLMA and DistBoost, which are representative semi-supervised metric learning methods that are also based on pairwise constraints as supervisory information.

4.1. Image Databases and Feature Representation

Our image retrieval experiments are based on two image databases. One database is a subset of the Corel Photo Gallery, which contains 1010 images belonging to 10 different classes. The 10 classes include bear (122), butterfly (109), cactus (58), dog (101), eagle (116), elephant (105), horse (110), penguin (76), rose (98), and tiger (115). Another database contains 547 images belonging to six classes that we downloaded from the Internet. The image classes are manually defined based on high-level semantics.

We first represent the images in the HSV color space, and then compute the *color coherence vector* (CCV) [9] as the feature vector for each image. Specifically, we quantize each image to $8\times8\times8$ color bins, and then represent the image as a 1024-dimensional CCV $(\alpha_1, \beta_1, \ldots, \alpha_{512}, \beta_{512})^T$, with α_i and β_i representing the numbers of coherent and non-coherent pixels, respectively, in the ith color bin. The CCV representation gives finer distinctions than the use of color histograms. Thus it usually gives better image retrieval results. For computational efficiency, we first apply PCA to retain the 60 dominating principal components before applying metric learning as described in the previous section.

4.2. Experimental Settings

The similarity constraints used in LLMA and LSML are obtained from the relevance feedback of the CBIR system, with each relevant image and the query image forming a similar image pair. It is straightforward to construct *chunklets* for RCA from the similarity constraints.

Besides RCA and LLMA, we also compare the image retrieval performance of our method with the baseline method of using Euclidean distance without distance learning. In summary, the following five methods are included in our comparative study: (1) Euclidean distance without metric learning; (2) Global metric learning with RCA; (3) Nonmetric distance boosting with DistBoost; (4) Local metric learning with LLMA; (5) Local metric learning with LSML.

We measure the retrieval performance based on *cumulative neighbor purity* curves. Cumulative neighbor purity measures the percentage of correctly retrieved images in the k nearest neighbors of the query image, averaged over all queries, with k up to some value K ($K = 20$ or 40 in our experiments).

For each retrieval task, we compute the average performance statistics over 5 randomly generated sets of similar image pairs. For both databases, the number of similar image pairs is set to 150, which is about 0.3% and 0.6%, respectively, of the total number of possible image pairs in the databases.

4.3. Experimental Results

Figure 4 shows the retrieval results on the first image database based on cumulative neighbor purity curves. We

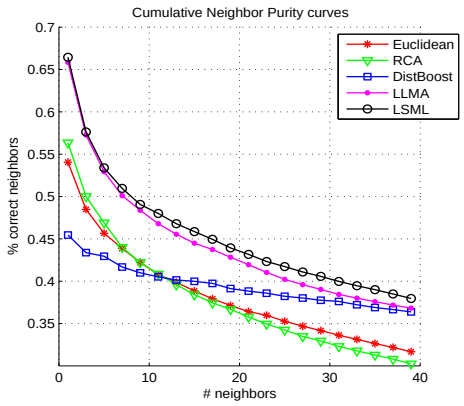


Figure 4. Retrieval results on the first image database.

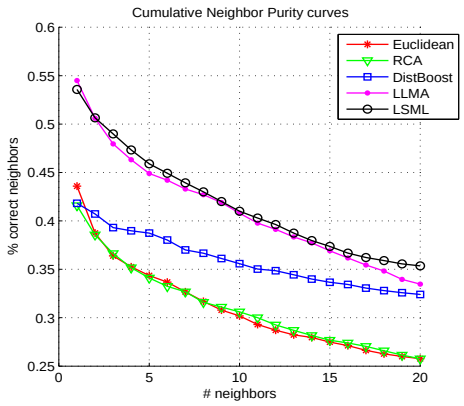


Figure 5. Retrieval results on the second image database.

can see that local metric learning with LLMA and LSML methods significantly improves the retrieval performance, while LSML leads to slightly better result than LLMA. DistBoost obtains high recall with more images retrieved, but low precision on the top ranking images, which most online image retrieval engines return. The retrieval results on the second image database are shown in Figure 5. Again, LSML outperforms other metric learning methods.

5. Concluding Remarks

We have proposed a novel semi-supervised metric learning method based on regularized moving least squares. This metric learning method possesses the simultaneous advantages of being flexible, efficient, topology-preserving, and generically applicable to different semi-supervised learning tasks. In particular, this method can be used to boost image retrieval performance.

A promising approach to the nonlinear extension of linear methods is through the so-called kernel trick. In our future work, we will investigate a kernel version of LSML so that the locally linear transformations are modeled as an implicit mapping corresponding to a kernel function.

References

[1] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 11–18, 2003. 1, 5

[2] H. Chang and D. Yeung. Locally linear metric adaptation for semi-supervised clustering. In *Proceedings of the Twenty-First International Conference on Machine Learning*, pages 153–160, 2004. 1, 2, 5

[3] C. Domeniconi, J. Peng, and D. Gunopulos. Locally adaptive metric nearest-neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285, 2002. 1

[4] J. Friedman. Flexible metric nearest neighbor classification. Technical report, Department of Statistics, Stanford University, Stanford, CA, USA, November 1994. 1

[5] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood component analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520. 2005. 1

[6] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):607–616, 1996. 1

[7] T. Hertz, A. Bar-Hillel, and D. Weinshall. Learning distance functions for image retrieval. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 570–577, 2004. 6

[8] D. Levin. The approximation power of moving least squares. *Mathematics of Computation*, 67(224):1517–1531, 1998. 1, 2

[9] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proceedings of the Fourth ACM International Conference on Multimedia*, 1996. 6

[10] W. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971. 5

[11] S. Schaefer, T. McPhail, and J. Warren. Image deformation using moving least squares. In *Proceedings of SIGGRAPH 2006*, 2006. 2

[12] R. Short and K. Fukunaga. The optimal distance measure for nearest neighbor classification. *IEEE Transactions on Information Theory*, 27(5):622–627, 1981. 1

[13] K. Weinberger, J. Blitzer, and L. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems 18*. 2006. 1, 3, 5

[14] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*. 2003. 1, 5

[15] Z. Zhang, J. Kwok, and D. Yeung. Parametric distance metric learning with label information. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 1450–1452, 2003. 1

[16] X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, 2005. CMU-LTI-05-192. 2