Locally Linear Metric Adaptation for Semi-Supervised Clustering

Hong Chang HONGCH@CS.UST.HK
Dit-Yan Yeung DYYEUNG@CS.UST.HK
Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

Abstract

Many supervised and unsupervised learning algorithms are very sensitive to the choice of an appropriate distance metric. While classification tasks can make use of class label information for metric learning, such information is generally unavailable in conventional clustering tasks. Some recent research sought to address a variant of the conventional clustering problem called *semi-supervised clustering*, which performs clustering in the presence of some background knowledge or supervisory information expressed as pairwise similarity or dissimilarity constraints. However, existing metric learning methods for semi-supervised clustering mostly perform global metric learning through a linear transformation. In this paper, we propose a new metric learning method which performs nonlinear transformation globally but linear transformation locally. In particular, we formulate the learning problem as an optimization problem and present two methods for solving it. Through some toy data sets, we show empirically that our *locally linear metric adaptation* (LLMA) method can handle some difficult cases that cannot be handled satisfactorily by previous methods. We also demonstrate the effectiveness of our method on some real data sets.

1. Introduction

Many machine learning and pattern recognition algorithms rely on a distance metric. Some commonly used methods are nearest neighbor classifiers, radial basis function networks and support vector machines for classification tasks and the k-means algorithm for clustering tasks. The performance of these methods often depends critically on the choice of an appropriate metric. Instead of choosing the metric manually, a promising approach is to learn the metric from data automatically. This idea can be dated back to some early work on optimizing the metric for k-nearest neighbor density estimation (Fukunaga & Hostetler, 1973). More recent research along this line continued to develop various locally adaptive metrics for nearest neighbor classifiers, e.g., (Domeniconi et al., 2002; Friedman, 1994; Hastie & Tibshirani, 1996; Lowe, 1995; Peng et al., 2002). Besides nearest neighbor classifiers, there are other methods that also perform metric learning based on nearest neighbors, e.g., radial basis function networks and variants (Poggio & Girosi, 1990).

While class label information is available for metric learning in classification tasks, such information is generally unavailable in conventional clustering tasks. To adapt the metric appropriately to improve the clustering results, some additional background knowledge or supervisory information should be made available. This learning paradigm between the supervised and unsupervised learning extremes is referred to as *semi-supervised clustering*, as contrasted to another type of semi-supervised learning tasks called *semi-supervised classification* which solves the classification problem with the aid of additional unlabeled data.

One type of supervisory information is in the form of limited labeled data.[1] Based on such information, Sinkkonen and Kaski (2002) proposed a local metric learning method to improve clustering and visualization results. Basu et al. (2002) explored using labeled data to generate initial seed clusters for the k-means clustering algorithm. Also, Zhang et al. (2003) proposed a parametric distance metric learning method for both classification and clustering tasks.

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004.

[1] Semi-supervised clustering with the aid of labeled data is essentially the same as semi-supervised classification with the aid of unlabeled data.

Another type of supervisory information is in the form of pairwise similarity or dissimilarity constraints. This type of supervisory information is weaker than the first type, in that pairwise constraints can be derived from labeled data but not vice versa. Wagstaff and Cardie (2000) and Wagstaff et al. (2001) proposed using such pairwise constraints to improve clustering results. Klein et al. (2002) introduced spatial generalizations to pairwise constraints, so that the pairwise constraints can also have influence on the neighboring data points. However, both methods do not incorporate metric learning into the clustering algorithms. Xing et al. (2003) proposed using pairwise side information in a novel way to learn a global Mahalanobis metric before performing clustering with constraints. Both Klein et al.'s and Xing et al.'s methods generally outperform Wagstaff et al.'s method in the experiments reported. Instead of using an iterative algorithm as in (Xing et al., 2003), Bar-Hillel et al. (2003) devised a more efficient, non-iterative algorithm called relevant component analysis (RCA) for learning a global Mahalanobis metric. However, their method can only incorporate similarity constraints. Shental et al. (2004) extended the work of (Bar-Hillel et al., 2003) by incorporating both pairwise similarity and dissimilarity constraints into the expectation-maximization (EM) algorithm for model-based clustering based on Gaussian mixture models. Kwok and Tsang (2003) established the relationship between metric learning and kernel matrix adaptation.

To summarize, we can categorize metric learning methods according to two different dimensions. The first dimension is concerned with whether (*supervised*) classification or (*unsupervised*) clustering is performed. Most methods were proposed for classification tasks, but some recent methods extended metric learning to clustering tasks under the semi-supervised learning paradigm. Supervisory information may be in the form of class label information or pairwise (dis)similarity information. The second dimension categorizes metric learning methods into *global* and *local* ones. Provided that sufficient data are available, local metric learning is generally preferred as it is more flexible in allowing different local metrics at different locations of the input space. In this paper, we propose a new metric learning method for semi-supervised clustering with pairwise similarity side information. While our method is local in the sense that it performs metric learning through locally linear transformation, it also achieves global consistency through interaction between adjacent local neighborhoods.

The rest of this paper is organized as follows. In Section 2, we present our metric learning method based on locally linear transformation. We also formulate the learning problem as an optimization problem. In Section 3, we present two methods for solving this optimization problem. Experimental results on both toy and real data are presented in Section 4, comparing our method with some previous methods. Finally, some concluding remarks are given in the last section.

2. Locally Linear Metric Adaptation

2.1. Basic Ideas

Let us denote a set of n data points in a d-dimensional input space by $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$. As in (Bar-Hillel et al., 2003), we only consider pairwise similarity constraints which are given in the form of a set $\mathcal{S}_0$ of similar point pairs. Intuitively, we want to transform the n data points to a new space in which the points in each similar pair will get closer to each other. To preserve the topological relationships between data points, we move not only the points involved in the similar pairs but also other points. For computational efficiency, we resort to linear transformation. One promising approach is to apply locally linear transformation so that the overall transformation of all points in $\mathcal{X}$ is linear locally but nonlinear globally, generalizing previous metric learning methods based on applying linear transformation globally (Bar-Hillel et al., 2003; Xing et al., 2003). We call this new metric learning method *locally linear metric adaptation* (LLMA). However, caution should be taken when applying linear transformation to reduce the distance between similar points, as a degenerate transformation will simply map all points to the same location so that all inter-point distances vanish (and hence become the smallest possible). Obviously this degenerate case is undesirable and should be avoided.

2.2. Metric Adaptation as an Optimization Problem

We now proceed to devise the metric learning algorithm more formally. We first generate the transitive and reflective closure $\mathcal{S}$ from $\mathcal{S}_0$. For each point pair $(\mathbf{x}_r, \mathbf{x}_s) \in \mathcal{S}$, we apply a linear transformation to the vector $(\mathbf{x}_s - \mathbf{x}_r)$ to give $\mathbf{A}_r(\mathbf{x}_s - \mathbf{x}_r) + \mathbf{c}_r$ for some $d \times d$ matrix $\mathbf{A}_r$ and d-dimensional vector $\mathbf{c}_r$. If a data point is involved in more than one point pair, we consider the transformation for each pair separately. The same linear transformation is also applied to every data point $\mathbf{x}_i$ in the neighborhood set $\mathcal{N}_r$ of $\mathbf{x}_r$. In other words, every data point $\mathbf{x}_i \in \mathcal{N}_r$ is transformed to

$$\begin{aligned} \mathbf{y}_i &= \mathbf{A}_r(\mathbf{x}_i - \mathbf{x}_r) + \mathbf{c}_r + \mathbf{x}_r \\ &= \mathbf{x}_i + (\mathbf{A}_r - \mathbf{I})\,\mathbf{x}_i + \mathbf{b}_r, \end{aligned}$$

where $\mathbf{b}_r = (\mathbf{I} - \mathbf{A}_r)\mathbf{x}_r + \mathbf{c}_r$ is the translation vector for all points $\mathbf{x}_i$'s in $\mathcal{N}_r$.

However, a data point $\mathbf{x}_i$ may belong to multiple neighborhood sets corresponding to different point pairs in $\mathcal{S}$. Thus, the new location $\mathbf{y}_i$ of $\mathbf{x}_i$ is the overall transformation effected by possibly all similar point pairs (and hence neighborhood sets):

$$\mathbf{y}_i = \mathbf{x}_i + \sum_{(\mathbf{x}_r, \mathbf{x}_s) \in \mathcal{S}} \pi_{ri}[(\mathbf{A}_r - \mathbf{I})\,\mathbf{x}_i + \mathbf{b}_r],$$

where $\pi_{ri} = 1$ if $\mathbf{x}_i \in \mathcal{N}_r$ and 0 otherwise.

Let m denote the number of point pairs in $\mathcal{S}$. Thus a total of m different transformations have to be estimated from the point pairs in $\mathcal{S}$, requiring $O(md^2)$ transformation parameters in $\{\mathbf{A}_r\}$ and $\{\mathbf{b}_r\}$. When m is small compared with the dimensionality d, we cannot estimate the $O(md^2)$ transformation parameters accurately. One way to get around this problem is to focus on a more restrictive set of linear transformations. The simplest case is to allow only translation, which can be described by md parameters. Obviously, translating all data points in a neighborhood set by the same amount leads to no change in the inter-point distances. Although some data points may fall into multiple neighborhood sets and hence this phenomenon does not hold, we want to incorporate an extra degree of freedom by changing the neighborhood sets to Gaussian neighborhood functions. More specifically, we set $\mathbf{A}_r$ to the identity matrix $\mathbf{I}$ and express the new location $\mathbf{y}_i$ of $\mathbf{x}_i$ as

$$\mathbf{y}_i = \mathbf{x}_i + \sum_{(\mathbf{x}_r, \mathbf{x}_s) \in \mathcal{S}} \pi_{ri}\mathbf{b}_r, \tag{1}$$

where π_{ri} is a Gaussian function defined as

$$\pi_{ri} = \exp\left[-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_r)^T \mathbf{\Sigma}_r^{-1}(\mathbf{x}_i - \mathbf{x}_r)\right],$$

with $\mathbf{\Sigma}_r$ being the covariance matrix. For simplicity, we use a hyperspherical Gaussian function, meaning that the covariance matrix is diagonal with all diagonal entries being ω^2. Thus π_{ri} can be rewritten as $\pi_{ri} = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_r\|^2/(2\omega^2)\right)$. Note that (1) can be expressed as

$$\mathbf{y}_i = \mathbf{x}_i + \mathbf{B}\,\boldsymbol{\pi}_i, \tag{2}$$

where $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m]$ is a $d \times m$ matrix and $\boldsymbol{\pi}_i = (\pi_{1i}, \pi_{2i}, \ldots, \pi_{mi})^T$ is an m-dimensional column vector. For data points that are far away from all points involved in $\mathcal{S}$ (and hence the centers of the neighborhoods), all π_{ri}'s are close to 0 and hence those points essentially do not move (since $\mathbf{y}_i \approx \mathbf{x}_i$).

We now formulate the optimization problem for finding the transformation parameters. The optimization criterion is defined as

$$J = d_{\mathcal{S}} + \lambda P, \tag{3}$$

where $d_{\mathcal{S}}$ is the sum of squared Euclidean distances for all similar pairs in the transformed space

$$d_{\mathcal{S}} = \sum_{(\mathbf{x}_r, \mathbf{x}_s) \in \mathcal{S}} \|\mathbf{y}_r - \mathbf{y}_s\|^2,$$

and P, a penalty term used to constrain the degree of transformation, is defined as

$$P = \sum_i \sum_j \mathcal{N}_\sigma(d_{ij})\,(q_{ij} - d_{ij})^2, \tag{4}$$

where $q_{ij} = \|\mathbf{y}_i - \mathbf{y}_j\|$ and $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$ represent the inter-point Euclidean distances in the transformed and original spaces, respectively. $\mathcal{N}_\sigma(d_{ij})$ is again in the form of a Gaussian function, as $\mathcal{N}_\sigma(d_{ij}) = \exp\left(-d_{ij}^2/\sigma^2\right)$, with parameter σ specifying the spread of the Gaussian window. The regularization parameter $\lambda > 0$ in (3) determines the relative significance of the penalty term in the objective function for the optimization problem. Note that the optimization criterion in (3) is analogous to objective functions commonly used in energy minimization models such as deformable models (Cheung et al., 2002), with the penalty term P playing the role of an internal energy term.

2.3. Iterative Metric Adaptation Procedure

The optimization problem formulated above is solved in an iterative manner, resulting in an iterative metric adaptation procedure. To increase the local specificity gradually over time to allow global nonlinearity in the transformation, the Gaussian window parameters ω and σ determining the neighborhood size and the weights in the penalty term, respectively, should decrease over time. We apply a simple method of decreasing the window parameters: $\omega^{(t)} = \beta\,\overline{q}^{(t)}/\sqrt{t}$, $\sigma^{(t)} = \gamma\,\omega^{(t)}$, for iteration $t = 1, 2, \ldots$, where $\overline{q}^{(t)}$ is the average inter-point Euclidean distance in the transformed space over all point pairs in $\mathcal{X}$ (i.e., $\overline{q}^{(t)} = \frac{2}{n(n-1)} \sum_{i<j} \|\mathbf{y}_i^{(t)} - \mathbf{y}_j^{(t)}\|$), and $\beta, \gamma > 0$ are two constant parameters.

At iteration t, given the data point locations $\{\mathbf{y}_i^{(t)}\}$ and the window parameters $\omega^{(t)}$ and $\sigma^{(t)}$, the overall optimization criterion in (3) is rewritten as the optimization criterion for iteration t:

$$J^{(t)}(\{\mathbf{b}_r\}; \{\mathbf{y}_i^{(t)}\}, \omega^{(t)}, \sigma^{(t)}) = \sum_{(\mathbf{x}_r, \mathbf{x}_s) \in \mathcal{S}} \|\mathbf{y}_r - \mathbf{y}_s\|^2 + \lambda \sum_i \sum_j \mathcal{N}_{\sigma^{(t)}}(d_{ij})(q_{ij} - d_{ij})^2. \tag{5}$$

Note that $\mathbf{y}_r$, $\mathbf{y}_s$ and q_{ij} depend on $\{\mathbf{b}_r\}$ and $\{\mathbf{y}_i^{(t)}\}$. However, for simplicity, the dependency is not explicitly shown on the right-hand side of (5). We seek to minimize $J^{(t)}$ by finding the optimal values of $\{\mathbf{b}_r\}$ as $\{\mathbf{b}_r^{(t)}\}$, which are then used to compute the location changes from $\{\mathbf{y}_i^{(t)}\}$ to $\{\mathbf{y}_i^{(t+1)}\}$.

There are two stopping criteria in our iterative algorithm. The first criterion is based on the ratio $\xi^{(t)}$ of the average inter-point distance over point pairs in $\mathcal{S}$ to that over all point pairs in $\mathcal{X}$ (i.e., $\overline{q}^{(t)}$). The procedure will stop when $\xi^{(t)}$ becomes smaller than some prespecified threshold ρ. Another stopping criterion is simply to set a maximum number of iterations T. The metric learning procedure will stop when either stopping criterion is satisfied.

We summarize our LLMA algorithm as follows:

1. $\mathbf{y}_i^{(1)} = \mathbf{x}_i$, $1 \le i \le n$; $t = 1$.
2. If $\xi^{(t)} < \rho$ or $t = T$, then exit.
3. $\omega^{(t)} = \beta\, \overline{q}^{(t)}/\sqrt{t}$; $\sigma^{(t)} = \gamma\, \omega^{(t)}$.
4. Compute $\boldsymbol{\pi}_i^{(t)} = \left(\pi_1^{(t)}(\mathbf{y}_i^{(t)}), \ldots, \pi_m^{(t)}(\mathbf{y}_i^{(t)})\right)^T$, $1 \le i \le n$, based on $\omega^{(t)}$.
5. Compute the optimal $\mathbf{b}_r^{(t)}$, $1 \le r \le m$, by minimizing $J^{(t)}$ in (5) w.r.t. $\{\mathbf{b}_r\}$.
6. Update all data points as
$$\mathbf{y}_i^{(t+1)} = \mathbf{y}_i^{(t)} + \sum_{r=1}^{m} \pi_r^{(t)}(\mathbf{y}_i^{(t)})\, \mathbf{b}_r^{(t)},\ 1 \le i \le n.$$
7. $t = t + 1$; go to Step 2.

In the algorithm, Step 5 is the key step which solves the optimization problem for each iteration based on the criterion in (5). In the next section, we present two methods for solving this optimization problem.

3. Optimization Methods

We now proceed to solve the optimization problem in Step 5 of the LLMA algorithm shown above. Two different optimization methods are discussed in the following two subsections.

3.1. Gradient Method

While the first term of $J^{(t)}$ in (5) is quadratic in $\{\mathbf{b}_r\}$, the second term is of a more complex form. So we cannot find a closed-form solution for the optimal values of $\{\mathbf{b}_r\}$ simply by solving $\nabla_{\mathbf{b}_r} J^{(t)} = \mathbf{0}$, $1 \le r \le m$. However, by using $q_{ij}^{(t)}$ to approximate $q_{ij}^{(t+1)}$, we can obtain an approximate closed-form solution

$$\mathbf{B}^{(t)} = -\mathbf{U}_1 \mathbf{U}_2^{+},$$

where

$$\begin{aligned}
\mathbf{U}_1 &= \sum_i \sum_j \left[s_{ij} + \lambda \mathcal{N}_{\sigma^{(t)}}(d_{ij}) \left(1 - d_{ij}/q_{ij}^{(t)}\right)\right] \times \\
&\qquad (\mathbf{y}_i^{(t)} - \mathbf{y}_j^{(t)})(\boldsymbol{\pi}_i^{(t)} - \boldsymbol{\pi}_j^{(t)})^T \\
\mathbf{U}_2 &= \sum_i \sum_j \left[s_{ij} + \lambda \mathcal{N}_{\sigma^{(t)}}(d_{ij}) \left(1 - d_{ij}/q_{ij}^{(t)}\right)\right] \times \\
&\qquad (\boldsymbol{\pi}_i^{(t)} - \boldsymbol{\pi}_j^{(t)})(\boldsymbol{\pi}_i^{(t)} - \boldsymbol{\pi}_j^{(t)})^T,
\end{aligned}$$

and $s_{ij} = 1$ if $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}$ and 0 otherwise. $\mathbf{U}_2^{+}$ denotes the pseudo-inverse of $\mathbf{U}_2$.

3.2. Iterative Majorization

Let us define two $d \times n$ matrices $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n]$ and $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]$ for n data points before and after transformation, respectively.[2] From (2), we have

$$\mathbf{Z} = \mathbf{Y} + \mathbf{B}\boldsymbol{\Pi} = (\mathbf{Y}\boldsymbol{\Pi}^{+} + \mathbf{B})\boldsymbol{\Pi} = \mathbf{L}\boldsymbol{\Pi},$$

where $\boldsymbol{\Pi} = [\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_n]$ is an $m \times n$ matrix. The optimization problem is then equivalent to minimization of $J(\mathbf{L})$ with respect to $\mathbf{L}$.

The optimization criterion $J(\mathbf{L})$ can be rewritten as:

$$\begin{aligned}
J(\mathbf{L}) &= \sum_{i,j} s_{ij} q_{ij}^2(\mathbf{L}) + \lambda \sum_{i,j} \mathcal{N}_\sigma(d_{ij})(q_{ij}(\mathbf{L}) - d_{ij})^2 \\
&= \sum_{i,j} (s_{ij} + \lambda \mathcal{N}_\sigma(d_{ij})) \left(q_{ij}(\mathbf{L}) - \frac{\lambda \mathcal{N}_\sigma(d_{ij})}{s_{ij} + \lambda \mathcal{N}_\sigma(d_{ij})} d_{ij} \right)^2 \\
&\quad + \lambda \sum_{i,j} \mathcal{N}_\sigma(d_{ij}) \left(1 - \frac{\lambda \mathcal{N}_\sigma(d_{ij})}{s_{ij} + \lambda \mathcal{N}_\sigma(d_{ij})} \right) d_{ij}^2.
\end{aligned}$$

We can omit the second term since it does not depend on $\mathbf{L}$. The equivalent optimization criterion is

$$\sum_i \sum_j \alpha_{ij} (q_{ij}(\mathbf{L}) - p_{ij})^2,$$

where

$$\begin{aligned}
\alpha_{ij} &= s_{ij} + \lambda \mathcal{N}_\sigma(d_{ij}) \\
p_{ij} &= \frac{\lambda \mathcal{N}_\sigma(d_{ij})}{s_{ij} + \lambda \mathcal{N}_\sigma(d_{ij})} d_{ij}.
\end{aligned}$$

Since this form is the same as that for multidimensional scaling for discriminant analysis (Webb, 1995),

[2] For notational simplicity, we use $\mathbf{Y}$ and $\mathbf{Z}$ rather than $\mathbf{Y}^{(t)}$ and $\mathbf{Y}^{(t+1)}$ here.

we can solve the optimization problem by *iterative majorization*, which can be seen as an EM-like algorithm for problems with no missing data. We define

$$\mathbf{C} = \sum_i \sum_j \alpha_{ij} (\boldsymbol{\pi}_i - \boldsymbol{\pi}_j)(\boldsymbol{\pi}_i - \boldsymbol{\pi}_j)^T$$

and

$$\mathbf{D}(\mathbf{L}) = \sum_i \sum_j e_{ij}(\mathbf{L}) \, (\boldsymbol{\pi}_i - \boldsymbol{\pi}_j)(\boldsymbol{\pi}_i - \boldsymbol{\pi}_j)^T$$

with

$$e_{ij}(\mathbf{L}) = \begin{cases} \frac{\lambda \, \mathcal{N}_\sigma(d_{ij}) \, d_{ij}}{q_{ij}(\mathbf{L})} & q_{ij}(\mathbf{L}) > 0 \\ 0 & q_{ij}(\mathbf{L}) = 0 \end{cases}$$

Then the optimization problem consists of the following steps:[3]

1. Initialize $\mathbf{L}^{(0)}$; $u = 0$.
2. $u = u + 1$; and compute
$$\mathbf{L}^{(u)} = \mathbf{L}^{(u-1)} \, (\mathbf{D}(\mathbf{L}^{(u-1)}))^T \, (\mathbf{C}^{-1})^T.$$
3. If converged, then stop; otherwise go to Step 2.

3.3. Other Methods

Recall that the penalty term P in (3) serves to constrain the degree of transformation, partly to avoid the occurrence of a degenerate transformation and partly to preserve the local topological relationships between data points. Besides defining the penalty term as in (4), there also exist other ways to achieve this goal. One possibility is to preserve the locally linear relationships between nearest neighbors, as in a nonlinear dimensionality reduction method called *locally linear embedding* (LLE) (Roweis & Saul, 2000). Due to page limit, details of this method are omitted here.

4. Experimental Results

To assess the efficacy of LLMA, we perform extensive experiments on toy data as well as real data from the UCI Machine Learning Repository.[4]

4.1. Illustrative Examples

Figure 1 demonstrates the power of our LLMA method by comparing it with the RCA method (Bar-Hillel et al., 2003) on three toy data sets.[5] RCA, as a metric learning method, changes the feature space by a global linear transformation which assigns large weights to relevant dimensions and low weights to irrelevant dimensions. The relevant dimensions are estimated based on connected components composed of similar patterns. For each data set, we randomly select 10 similar pairs to form $\mathcal{S}_0$. While RCA can perform well on the first data set, its performance is significantly worse than LLMA on the second and third data sets which are much more difficult cases. On the other hand, LLMA can give satisfactory results for all three cases. More details about these experiments will be given in Section 4.3.

4.2. Clustering Algorithms and Performance Measures for Comparative Study

In order to assess the efficacy of LLMA for semi-supervised clustering, we compare the clustering results based on k-means with and without metric learning. Besides RCA method, we also repeat the experiments using the constrained k-means algorithm (Wagstaff et al., 2001). Constrained k-means algorithm is based on default Euclidean metric subject to the constraints that patterns in a pair $(\mathbf{x}_r, \mathbf{x}_s) \in \mathcal{S}$ are always assigned to the same cluster. More specifically, the following four clustering algorithms are compared:

1. k-means without metric learning
2. Constrained k-means without metric learning
3. k-means with RCA for metric learning
4. k-means with LLMA for metric learning

The Rand index (Rand, 1971) is used to measure the clustering quality in our experiments. It reflects the agreement of the clustering result with the ground truth. Let n_s be the number of point pairs that are assigned to the same cluster (i.e., matched pairs) in both the resultant partition and the ground truth, and n_d be the number of point pairs that are assigned to different clusters (i.e., mismatched pairs) in both the resultant partition and the ground truth. The Rand index is defined as the ratio of $(n_s + n_d)$ to the total number of point pairs, i.e., $n(n-1)/2$. When there are more than two clusters, however, the standard Rand index will favor assigning data points to different clusters. We modify the Rand index as in (Xing et al., 2003) so that matched pairs and mismatched pairs are assigned weights to give them equal chance of occurrence (0.5).

[3] Note that the iteration count u here is different from t in the LLMA algorithm shown above. This optimization problem is for Step 5 of each iteration t of the algorithm.

[4] `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[5] The MATLAB code for RCA was downloaded from the web page of an author of (Bar-Hillel et al., 2003).

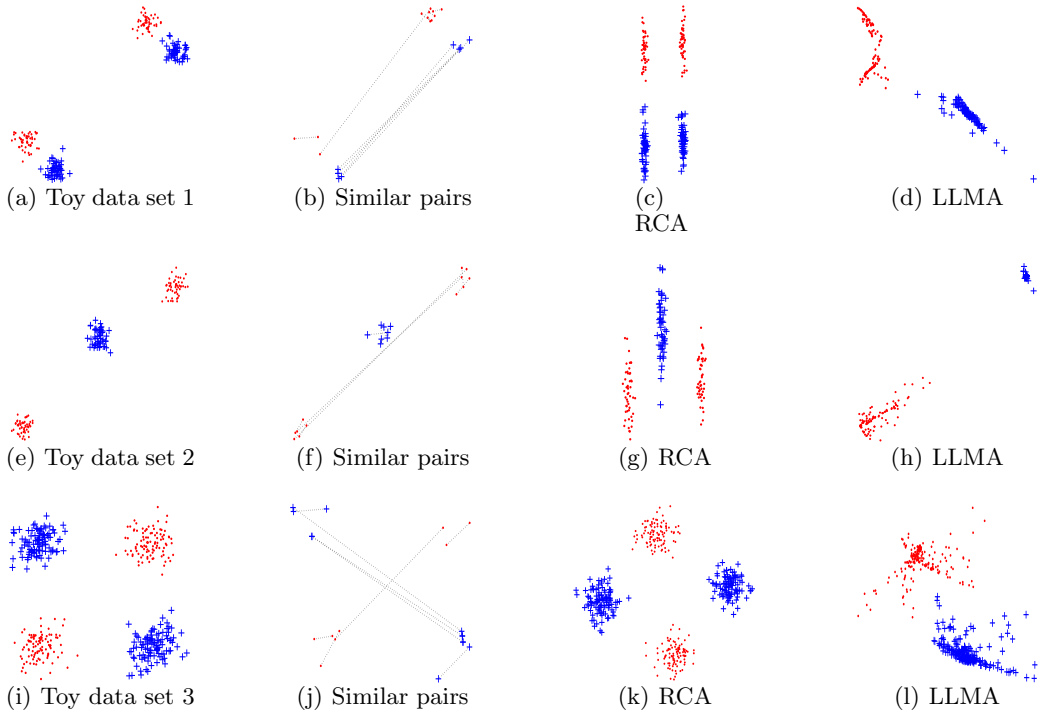


Figure 1. Comparison of LLMA with RCA on three toy data sets. Subfigures in the first column show the data sets each with two classes, while subfigures in the second column show 10 similar pairs in $\mathcal{S}_0$ for each data set. The third and fourth columns show the data sets after applying RCA and LLMA, respectively, for metric learning.

To see how different algorithms vary their performance with the background knowledge provided, we use 20 randomly generated $\mathcal{S}_0$ sets for each data set. Moreover, we compute the average Rand index over 20 random runs of (constrained) k-means for each $\mathcal{S}_0$ set. The results for all four algorithms are then shown as box-plots using MATLAB.

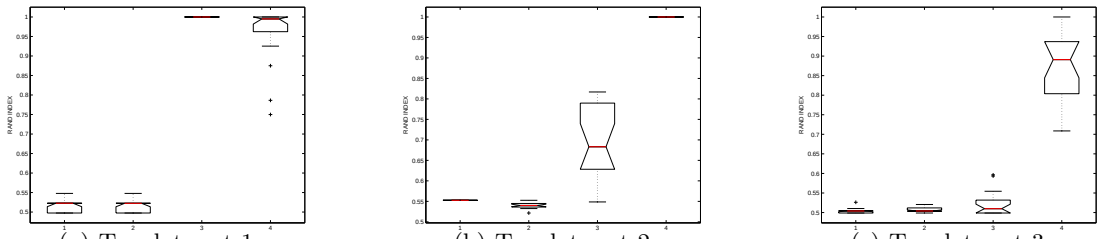
4.3. Semi-Supervised Clustering on Toy and UCI Data Sets

In the LLMA algorithm, there are a few parameters that need to be set before running the experiments. These parameters are quite easy to set based on their physical meanings. The two parameters, β and γ, for the decay functions of the Gaussian windows are set to $[0.1, 3]$ and $(0, 1)$, respectively. The regularization parameter λ adjusting the tradeoff between local transformation and geometry preservation is set to $[1, 5]$. For the stopping criteria, we set ρ to $[0.1, 0.2]$ and T to 5 (i.e., very few iterations of the LLMA algorithm are run). All data sets are normalized before applying the four algorithms. Gradient method is used to obtain the experimental results shown, which are similar to those obtained using iterative majorization.

Figure 2 shows the clustering results for the three toy data sets as illustrated in Section 4.1. Obviously, all the three data sets cannot be clustered well using the standard and constrained k-means algorithms. Even RCA can give good result only on the first data set. On the other hand, LLMA can handle all these cases and perform particularly well on the second and third data sets which cannot be handled satisfactorily by the other methods.

We further conduct experiments on nine UCI data sets. The number of data points n, the number of features d, the number of classes c, and the number of randomly selected similar pairs $|\mathcal{S}_0|$ are shown under each subfigure in Figure 3. From the clustering results, we can see that LLMA outperforms the other methods for most of these data sets. As for the iris, Boston housing and balance data sets, RCA can improve the clustering results most.

To summarize, these experimental results on both toy and real data sets demonstrate the effectiveness of our LLMA method.



(a) Toy data set 1 (b) Toy data set 2 (c) Toy data set 3

Figure 2. Clustering results for toy data sets shown as box-plots for 20 different $\mathcal{S}_0$ sets with $|\mathcal{S}_0| = 10$ (the four clustering algorithms are numbered as in Section 4.2).

5. Concluding Remarks

In this paper, we have proposed a new metric learning method called LLMA for semi-supervised clustering. Unlike previous methods which can only perform linear transformation globally, LLMA performs nonlinear transformation globally but linear transformation locally. This generalization makes it more powerful for solving some difficult clustering tasks as demonstrated through the toy data sets. To solve the optimization problem as one step in the LLMA algorithm, we have presented two methods and hinted some other possibilities, such as a spectral method like that used in LLE. We have also compared our method with some previous methods using real data sets.

Note that in LLMA, the original input space and the transformed space are explicitly related via a mapping, as $\mathbf{Y} = \mathbf{L}\mathbf{\Pi}$, where $\mathbf{\Pi}$ is a nonlinear function with respect to $\mathbf{X}$. Although it is not necessary for clustering problems, it is possible for new data points added to the input space to be mapped onto the transformed space. This possibility will be explored as we extend our LLMA method to other applications.

Currently, our method can only utilize similarity constraints. A natural question to ask is whether we can extend LLMA by incorporating dissimilarity constraints. In principle this is possible, but the optimization criterion has to be modified in order to incorporate the new constraints. One challenge to face is to maintain the form of the objective function so that the optimization problem remains tractable.

Moreover, we have only considered a restrictive form of locally linear transformation, namely, translation. A potential direction to pursue is to generalize it to more general linear transformation types. Other possible research directions include improving the current LLMA algorithm such as performing globally linear transformation first and then LLMA only when necessary.

References

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2003). Learning distance functions using equivalence relations. *Proceedings of the Twentieth International Conference on Machine Learning* (pp. 11–18). Washington, DC, USA.

Basu, S., Banerjee, A., & Mooney, R. (2002). Semi-supervised clustering by seeding. *Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 19–26). Sydney, Australia.

Cheung, K., Yeung, D., & Chin, R. (2002). On deformable models for visual pattern recognition. *Pattern Recognition*, *35*, 1507–1526.

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

Friedman, J. (1994). *Flexible metric nearest neighbor classification* (Technical Report). Department of Statistics, Stanford University, Stanford, CA, USA.

Fukunaga, K., & Hostetler, L. (1973). Optimization of k-nearest neighbor density estimates. *IEEE Transactions on Information Theory*, *19*, 320–326.

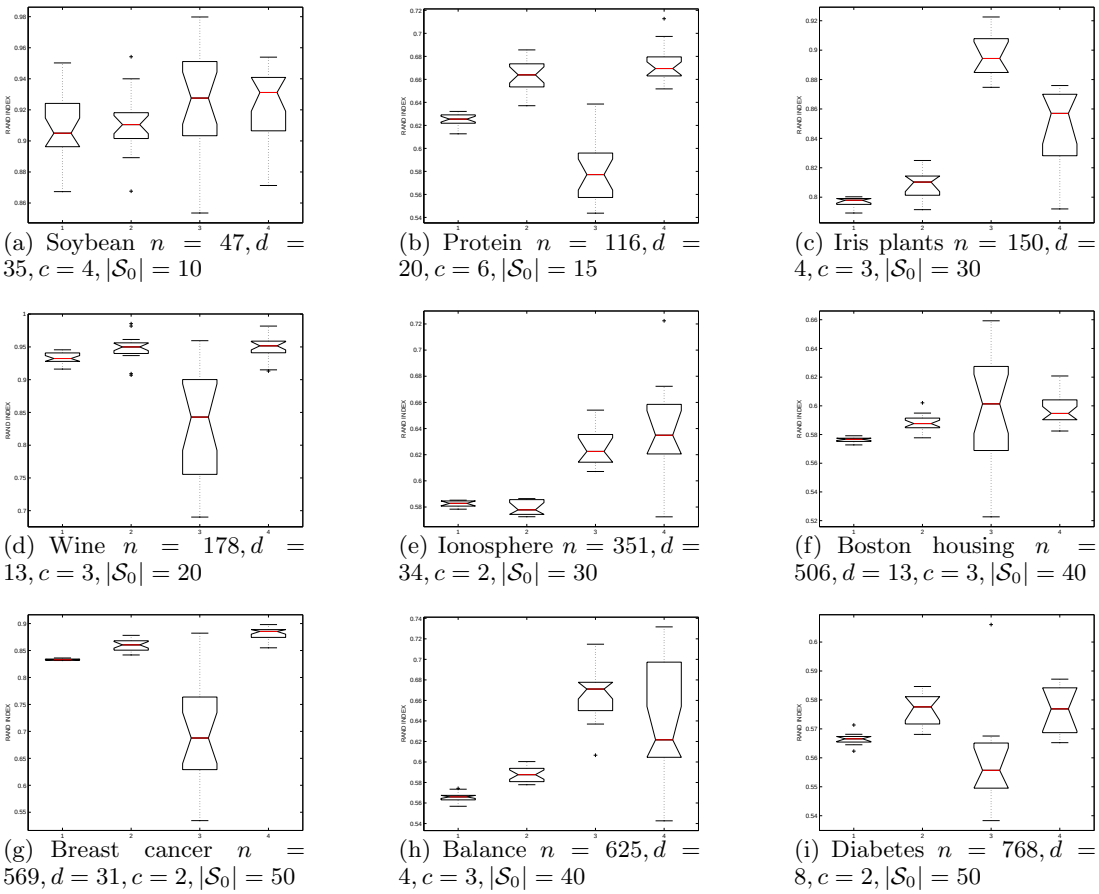
Hastie, T., & Tibshirani, R. (1996). Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*, 607–616.

Klein, D., Kamvar, S., & Manning, C. (2002). From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering. *Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 307–314). Sydney, Australia.

Kwok, J., & Tsang, I. (2003). Learning with idealized kernels. *Proceedings of the Twentieth International Conference on Machine Learning* (pp. 400–407). Washington, DC, USA.

Lowe, D. (1995). Similarity metric learning for a variable-kernel classifier. *Neural Computation*, *7*, 72–85.

Peng, J., Heisterkamp, D., & Dai, H. (2002). Adaptive kernel metric nearest neighbor classification. *Proceedings of the Sixteenth International Conference on Pattern Recognition* (pp. 33–36). Québec City, Québec, Canada.



(a) Soybean $n = 47, d = 35, c = 4, |\mathcal{S}_0| = 10$
(b) Protein $n = 116, d = 20, c = 6, |\mathcal{S}_0| = 15$
(c) Iris plants $n = 150, d = 4, c = 3, |\mathcal{S}_0| = 30$
(d) Wine $n = 178, d = 13, c = 3, |\mathcal{S}_0| = 20$
(e) Ionosphere $n = 351, d = 34, c = 2, |\mathcal{S}_0| = 30$
(f) Boston housing $n = 506, d = 13, c = 3, |\mathcal{S}_0| = 40$
(g) Breast cancer $n = 569, d = 31, c = 2, |\mathcal{S}_0| = 50$
(h) Balance $n = 625, d = 4, c = 3, |\mathcal{S}_0| = 40$
(i) Diabetes $n = 768, d = 8, c = 2, |\mathcal{S}_0| = 50$

Figure 3. Clustering results for UCI data sets shown as box-plots for 20 different $\mathcal{S}_0$ sets (the four clustering algorithms are numbered as in Section 4.2).

Poggio, T., & Girosi, F. (1990). Networks for approximation and learning. *Proceedings of the IEEE*, *78*, 1481–1497.

Rand, W. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, *66*, 846–850.

Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Shental, N., Bar-Hillel, A., Hertz, T., & Weinshall, D. (2004). Computing Gaussian mixture models with EM using equivalence constraints. In *Advances in neural information processing systems 16*. Cambridge, MA, USA: MIT Press. To appear.

Sinkkonen, J., & Kaski, S. (2002). Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, *14*, 217–239.

Wagstaff, K., & Cardie, C. (2000). Clustering with instance-level constraints. *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 1103–1110). Stanford, CA, USA.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained k-means clustering with background knowledge. *Proceedings of the Eighteenth International Conference on Machine Learning* (pp. 577–584). Williamstown, MA, USA.

Webb, A. (1995). Multidimensional scaling by iterative majorization using radial basis functions. *Pattern Recognition*, *28*, 753–759.

Xing, E., Ng, A., Jordan, M., & Russell, S. (2003). Distance metric learning, with application to clustering with side-information. In S. Becker, S. Thrun and K. Obermayer (Eds.), *Advances in neural information processing systems 15*, 505–512. Cambridge, MA, USA: MIT Press.

Zhang, Z., Kwok, J., & Yeung, D. (2003). Parametric distance metric learning with label information. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence* (pp. 1450–1452). Acapulco, Mexico.