Manifold Clustering

Richard Souvenir and Robert Pless
Washington University in St. Louis
Department of Computer Science and Engineering
Campus Box 1045, One Brookings Drive, St. Louis, MO 63130
{rms2, pless}@cse.wustl.edu

Abstract

Manifold learning has become a vital tool in data driven methods for interpretation of video, motion capture, and handwritten character data when they lie on a low dimensional, non-linear manifold. This work extends manifold learning to classify and parameterize unlabeled data which lie on multiple, intersecting manifolds. This approach significantly increases the domain to which manifold learning methods can be applied, allowing parameterization of example manifolds such as figure eights and intersecting paths which are quite common in natural data sets. This approach introduces several technical contributions which may be of broader interest, including node-weighted multidimensional scaling and a fast algorithm for weighted low-rank approximation for rank-one weight matrices. We show examples for intersecting manifolds of mixed topology and dimension and demonstrations on human motion capture data.

1. Introduction

Data-driven modeling is a powerful approach for non-rigid motion analysis. Manifold learning approaches have been applied to automatically parameterize image data sets including head pose, facial expressions, bird flight, MR imagery, and handwritten characters. Each of these data sets lies on low-dimensional manifolds that are not linear subspaces of the (high-dimensional) input data space. Manifold learning approaches seek to explicitly or implicitly define a low-dimensional embedding of the data points that preserves some properties (such as geodesic distance or local relationships) of the high-dimensional point set.

When the input data points are drawn from multiple (low-dimensional) manifolds, many manifold learning approaches suffer. In the case where the multiple manifolds are separated by a gap, techniques such as isometric feature mapping (Isomap) [15] may discover the dif-
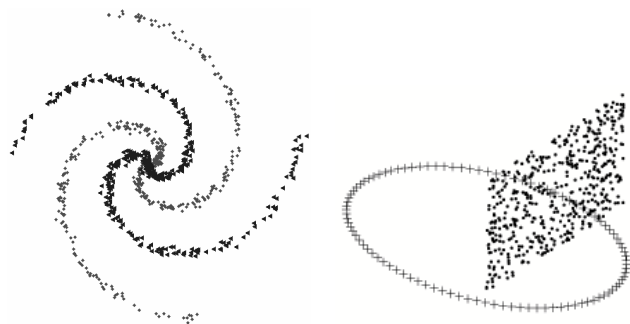

Figure 1. For high-dimensional data points which lie on intersecting low-dimensional manifolds, manifold embedding techniques benefit from first separating points into distinct classes. These toy problems illustrate relevant cases. (left) The spirals data set, which can be embedded in two dimensions with minimal error, can also be embedded as two one-dimensional manifolds. (right) The circle-plane data set includes component manifolds of different dimension. The segmentations shown here by the different symbols are automatically determined by the approach developed in this paper.

ferent manifolds as different connected components in the local neighborhood graph, and spectral clustering techniques may identify and cluster each manifold based on the optimization of certain objective functions. However, if there is significant overlap in the manifolds, prior methods fail in one of two ways: either, in the case of Isomap, from the non-existence of a low-dimensional embedding which exactly (or nearly) preserves properties of the high-dimensional manifold, or, in the case of Locally Linear Embedding (LLE) [13] or Semidefinite Embedding (SDE) [16], from the fact that additional work is necessary to interpret

the coordinates of the low-dimensional embedding.

In the next section, we review methods for non-linear dimensionality reduction and discuss the limitations of existing algorithms on datasets where the points come from multiple, intersecting manifolds. In section 3, we present our algorithm for manifold clustering and highlight two novel technical contributions: node-weighted multidimensional scaling and a fast algorithm for weighted low-rank approximation for rank-one weight matrices. In section 4 we show results of our algorithm on artificially generated data and human motion capture data of actors performing different activities. Finally, we conclude in section 5 with a brief discussion.

2. Related Work

Classical dimensionality reduction techniques for a image sets rely on Principle Component Analysis (PCA) [9] and Independent Component Analysis (ICA) [8]. These seek to represent data as linear combinations of a small number of basis vectors. However, many natural image data sets have an intrinsic dimensionality that is much less than the number of basis images required to linearly reconstruct them.

This has led to a number of methods seeking to parameterize low-dimensional, non-linear manifolds. These methods measure local distances or approximate geodesic distances between points in the original data set, and seek low-dimensional embeddings that preserve these properties. Isomap [15] extends classic multidimensional scaling (MDS) by substituting an estimate of the geodesic distance along the image manifold for the inter-image Euclidean distance as input. LLE [13] attempts to represent the image manifold locally by reconstructing each image as weighted combination of its neighbors. SDE [16] applies semidefinite programming to learn kernel matrices which can be used to create isometric embeddings. Isomap performs well for image sets sampled from convex manifolds. LLE and SDE do not fail in the case of non-convexity, but do not correctly work to parameterize a sphere (i.e., give points on a sphere two coordinates instead of 3), such as [12]. These algorithms, and others [6, 4, 2] have been used in various applications, including classification, recognition, tracking.

There has been some recent work in manifold learning for datasets whose points do not lie on a single, simple manifold. In [3], a *multi-manifold problem* is described where the data, comes from an underlying, possibly large set of low-dimensional manifolds. The authors exploit commonalities in the tangent spaces on different parts of the manifold in order to learn manifolds for under-sampled data. In [17], the *multi-class manifold* problem is presented. This considers data sets whose points come from a single, underlying low-dimensional manifold; however, this manifold is

sampled in such a way that large "gaps" are introduced and the data set is fragmented. [17] addresses a failure mode of Isomap described in [1] where, for data sets with certain properties, any neighborhood size selected either "short-circuits" the true manifold or only learns the manifold for a subset of the data points. The authors demonstrate an algorithm for learning the underlying manifold by differentiating between intra- and inter- fragment distances. Similarly, in [7], data sets whose points lie on disjoint manifolds are embedded on a single coordinate system.

In this work we consider the case where the multiple underlying manifolds are of mixed topology and dimensionality and also not necessarily fragments of a single underlying manifold. In addition, we focus on the case of intersecting manifolds.

3. Manifold Clustering

Our goal is to partition an input data set into clusters where each cluster contains data points from a single, simple low-dimensional manifold. We start by assuming that the number and dimensionality of the low-dimensional manifolds are known. Specifically:

Given a set of points $\{X_1, X_2, \cdots, X_n\}$ *derived from* k *intersecting non-linear manifolds where* $X_i \in \mathbb{R}^D$ *for some dimension D, output the set of labels* $\{c_1, c_2, \cdots, c_n\}$ *where* $c_i \in 1, 2, \ldots, k$ *is an index specifying to which output manifold a point belongs, and* $\{Y_1, Y_2, \cdots, Y_n\}$ *where* $Y \in R^d$ *(for $d < D$) is the low dimensional embedding of a point on its associated manifold.*

Without any priors on the labels of the input data, we are left to simultaneously learn this labeling and estimate the parameters of the underlying manifolds. The natural solution to this class of problems is an Expectation-Maximization (EM) type algorithm. In figure 2, we provide a sketch of the algorithm. The remainder of this section describes the problems that arise and their technical solutions.

Figure 3 shows a plot of our algorithm converging to a solution on an artificially generated "spiral" data set with $k = 3$ clusters. The plot estimates the classification error using the weighted average of the residuals to each cluster.

3.1. Algorithm Design

The algorithm begins in the same manner as Isomap (steps 1, 2, and 3), by estimating geodesic distances between points. The goal is to partition the points so that within each partition, the geodesic distances are consistent with the Euclidean distances of low dimensional embedding. This partitioning step is performed with an Expectation Maximization approach.

Classic EM algorithms have two distinct steps: the assignment of data points to model(s) of best fit (E-Step) and

Given: A set of images $\{X_1, X_2, \cdots, X_n\}$ and a desired label set size, k

1. Calculate $d(i,j) = ||X_i - X_j||_2$

2. Create a graph $G = (V, E)$ with a node for each image, and an edge between pairs of neighboring[a] images and set the edge weight to that distance.

3. Compute all pairs shortest path distances on G. Let $d(i,j)$ be the length of the shortest path between node i and node j. Define a distance matrix D such that $D(i,j) = d(i,j)^2$.

4. **Initialization:** Create a $k \times n$ matrix W where w_{ci} is the probability of X_i belonging to manifold c. W can be initialized randomly unless domain-specific priors are available.

5. **M-Step:** For each class, c, supply $\vec{w}_c$ and D and use node-weighted MDS to embed the points in the desired dimension.

6. **E-Step:** For each point, estimate the distance to the manifolds implied by the output of Step 5 and re-weight accordingly.

7. Go to Step 5 until convergence.

---

[a]Two common methods for selecting neighboring images are ϵ-sphere and k-nearest neighbors.

Figure 2. Our approach for manifold clustering. Steps 1-3 are identical to the initialization steps of the Isomap algorithm.

the estimation the parameters of those models (M-Step). In most cases, the assignments of points to models are partial, or soft, assignments. In the M-Step, these assignments serve to weight the contribution of each data point in defining each model. Since current manifold learning algorithms treat each data point equally, one challenge is to develop manifold embedding techniques for weighted point sets. This challenge is met with our algorithm using the development of *Node-Weighted Multidimensional Scaling*, which we introduce in Section 3.2.

After the manifold is constructed, the E-step seeks to update the assignments of each point by determining how well they fit each model. This cannot be done directly because MDS does not explicitly construct a manifold. Section 3.3 discusses a heuristic to approximate a measure of the distance of each point to each manifold.
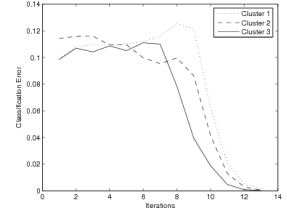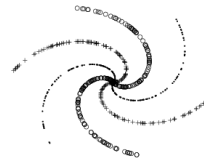

Figure 3. The plot on the right shows the decreasing classification error during the iterations of our algorithm. This estimate of classification error is the weighted average of the residuals of all the points to each cluster.

3.2. Node-Weighted MDS

Below, we outline traditional multidimensional scaling. In our approach, we require a weighted version of this procedure. We call this procedure node-weighted MDS to distinguish it from traditional weighted multidimensional scaling, commonly referred to as individual differences scaling or INDSCAL [5], which considers the problem of balancing multiple distance (or similarity matrices) when each point may have different weights with respect to different distance matrices.

MDS (Multi-Dimensional Scaling) Given $n \times n$ matrix D, such that $D(i,j)$ is the desired squared distance from point i to point j:

1. Define $\tau = -HDH/2$, (H is called the centering matrix, $H = I - \vec{e}\vec{e}^\top/n$ and $\vec{e} = [1, 1, \cdots, 1]^\top$).

2. Let $s_1, s_2, \ldots$ be the (sorted in decreasing order) eigenvalues of τ, and let $v_1, v_2, \ldots$ be the corresponding (column) eigenvectors. The matrix $Y = [\sqrt{s_1}v_1 | \sqrt{s_2}v_2 | \ldots \sqrt{s_k}v_k]$ has row vectors which are the coordinates of the best k-dimensional embedding.

The matrix $YY^\top$ is the best rank k approximation to τ (with respect to the L_2 matrix norm). The process finds the k-dimensional coordinates that minimize $\sum_{ij} \left(|Y_i - Y_j|_2^2 - D(i,j)\right)^2$ [10]. In contrast, node-weighted MDS seeks to minimize the following:

$$\sum_{ij} w_i w_j \left(|Y_i - Y_j|_2^2 - D(i,j)\right)^2.$$

The process starts by changing the initial centering matrix to be a weighted centering matrix:

$$H' = (I - \vec{w}\vec{w}^\top)/\sum_{i=1}^{n} w_i,$$

and then defining the correlation matrix $\tau = -H'DH'/2$. We then seek τ_k, a rank-k approximation to τ, that minimizes the weighted L_2 matrix norm:

$$\sum_{ij} w_i w_j \left(\tau_k(i,j) - \tau(i,j) \right)^2 .$$

Given τ_k, then finding Y such that $YY^\top = \tau_k$, (using the same eigenvector decomposition as above) gives the k-dimension coordinates of the node-weighted MDS embedding. Efficiently solving for τ_k is the subject of the next section.

Fast Low Rank Embedding. The weighted low rank embedding problem (using our variables from the last section) is formally: *Given a matrix τ and a weight matrix W of the same dimensions, find the matrix τ_k of rank k that minimizes the Frobenius norm of the weighted difference: $||W \otimes (A - M)||_F$, where the $\otimes$ operator indicates element-wise multiplication of matrix elements.* This problem has been approached with both the weight matrices constrained to be $\{0,1\}$ valued and the general case of real-valued weights. Recent work suggests an iterative solution [14] to this problem.

Our application has additional constraints on the matrices τ and W that allow a direct solution to this problem. These conditions are general enough to potentially be of interest in other approaches or application domains. In particular, our weight matrix is symmetric and of rank 1, and can be expressed as the outer product of the node weights expressed as a column vectors $W = \vec{w}\vec{w}^\top$. If we define $\tilde{W} = diag(\vec{w})$, (a matrix of all zeros with the weights w along the diagonal), then we can simplify this special case of weighted low-rank approximation as follows:

$$\begin{aligned} ||(\vec{w}\vec{w}^\top) \otimes (\tau - \tau_k)||_F &= ||\tilde{W}(\tau - \tau_k)\tilde{W}||_F \\ &= ||\tilde{W}\tau W_2 - \tilde{W}\tau_k\tilde{W}||_F \\ &= ||\tilde{W}\tau\tilde{W} - R||_F, \qquad (1) \end{aligned}$$

where R is the low rank (unweighted) approximation to $\tilde{W}\tau\tilde{W}$. R can be found with standard singular value decomposition, leaving:

$$\begin{aligned} \tilde{W}\tau_k\tilde{W} &= R \\ \tau_k &= \tilde{W}^{-1}R\tilde{W}^{-1}. \qquad (2) \end{aligned}$$

3.3. Manifold Distance Metrics

The result of node-weighted MDS and other manifold learning methods is a low dimensional embedding of the points. To implement the E-step of the algorithm in Figure 2, we need to re-weight the points based on how well they fit each of the k manifolds. This section details a method to estimate the distance of a point to a manifold
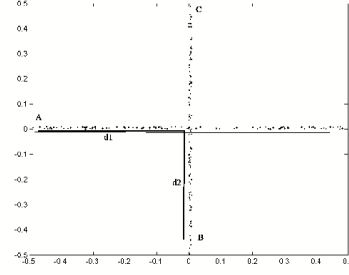
(a) Original Data Set
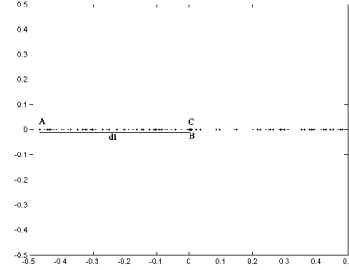
(b) Node-Weighted MDS Embedding

Figure 4. (a) A set of points representing 2 1-D manifolds in 2 dimensions. The horizontal line corresponds to the manifold implied by the highly weighted points along the x-axis. (b) The result of node-weighted MDS where the points along the x-axis (a) have a weight of 1 and the points along the y-axis have a weight of 0.

which is defined only implicitly through the weighted embedding of points. The idea is to compare the original geodesic distances between points with the distance implied by the weighted embedding. If the distances between a point p_i and the highly weighted points are similar, then p_i fits well on this manifold.

Figure 4 shows an example of 2-D points embedded using node-weighted MDS. The points along the horizontal line (including the point labeled A) have a weight of 1 and the points along the perpendicular (including points B and C) have a weight of 0. To estimate the distance of point B to this manifold, we consider the difference of the geodesic distance of point B to the other points on the manifold and the distance in the low dimensional embedding. In the case of points B and A: $d_G(A,B) = d1 + d2$, $d_E(A,B) = d1$, and the distance to the manifold $d_M(A,B) = d_G - d_E = d2$. In the weighted case, $d_M(A,B) = w(B)(d_G - d_E)$. So, for this example, the large value of $d_G(B,C) - d_E(B,C)$ would not contribute to the distance of point B to the mani-

fold since we do not consider point C to lie on this manifold. Averaging this distance over all of the points, we define the distance of point p_i to manifold M as:

$$d_M(p_i) = \frac{\sum_{j=1}^{n} w(p_j)(d_G(p_i, p_j) - d_E(p_i, p_j))}{\sum_{j=1}^{n} w(p_j)}$$

3.4. Classifying Data Points

For each iteration of our algorithm, we calculate the weight of each point belonging to each of the k clusters, using the *softmin* function. So, the weight of point p_i belonging to cluster c is:

$$w_i^c = \frac{e^{\frac{-d_{M_c}(p_i)^2}{\sigma^2}}}{\sum_{j=1}^{k} e^{\frac{-d_{M_j}(p_i)^2}{\sigma^2}}}$$

The ability of this algorithm to converge is sensitive the variance, σ. We calculate the weighted variance for each cluster:

$$\hat{\sigma}^{c^2} = \frac{\sum w_i^c}{(\sum w_i^c)^2 - \sum (w_i^{c2})} \sum w_i^c (d_{M_c})^2$$

4. Experimental Results

The extension of manifold learning to data sets that arise from multiple intersecting manifolds is an important step in applying these techniques more broadly. In this section we present results showing manifold clustering results on a several examples including manifolds of different topology and dimension, and an application to human motion capture data demonstrating the algorithms applicability to natural data sets. In each case, the dimensionality of the component clusters was provided to the algorithm (i.e., in the left most example of Table 1, the EM algorithm fit the data to 3 1-D manifolds, in the third example, the model data is fit to a circle and a 2-D plane). When the component manifold topology was circular, the low-dimensional embedding algorithm used is the node-weighted analog to the modified MDS proposed in [12].

4.1. Comparison to Intrinsic Dimensionality Estimates

Recent work in the field of intrinsic dimensionality estimation allows us to obtain a measure of the quality of our output. A method is proposed in [11] which estimates the intrinsic dimensionality of a data set by applying the principle of maximum likelihood to the distances between neighboring data points. Table 1 shows the results of our segmentation on artificial data sets and the estimated dimensionality of the subsets, as clustered by our algorithm.

4.2. Human Activity Parsing

We applied our method to the analysis of human motion capture data of various simple activities. Each data set contains examples of a single actor performing a series of simple motions. One test was applied on a data set of an actor performing a series of basketball referee signals. The actor performed each of the 3 signals (technical foul, jump ball, and carrying) 3 times in the sequence. The data set consisted of 2212 frames of $x-$, $y-$, and $z-$coordinates for 175 markers. Each frame therefore represented a point in some 525-dimensional space (175 markers * 3 coordinates). We applied our method using $k = 3$ 1-D clusters to label each frame. Ground truth was obtained by manual annotation of the original video sequence. We then compared our labeling to the ground truth. Figure 5 shows results on this data set. On this data set, our method performs with 94.8% accuracy.

5. Summary and Conclusions

In this paper, we presented a method for learning manifolds of data sets that originate from multiple, intersecting low-dimensional manifolds. The main contributions are the unsupervised learning algorithm for factoring low-rank manifolds, node-weighted multi-dimensional scaling, and a fast (rank-one weighted) low-rank approximation scheme. We applied this approach on artificially generated data sets and demonstrated the application of our method to parsing natural human motion into component behaviors.

References

[1] M. Balasubramanian, E. L. Schwartz, J. B. Tenenbaum, V. de Silva, and J. C. Langford. The isomap algorithm and topological stability (technical comment). *Science*, 295(5552):7a–, 2002.

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 585–591, Cambridge, MA, 2002. MIT Press.

[3] Y. Bengio and M. Monperrus. Non-local manifold tangent learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 129–136. MIT Press, Cambridge, MA, 2005.

[4] M. Brand. Charting a manifold. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 961–968. MIT Press, Cambridge, MA, 2003.
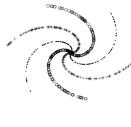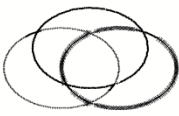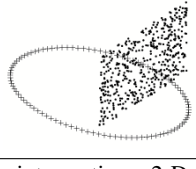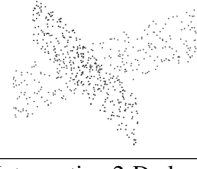
Segmentations				
Data Set	6-arm spiral	Interlocking circles	Circle intersecting a 2-D plane	Intersecting 2-D planes
Total points	800	600	500	600
Estimates	[1.05, 1.05, 1.06]	[1.23, 1.23, 1.40]	[1.14 (circle), 2.22 (plane)]	[2.14, 2.16]

Table 1. The estimated dimensionality of the output subsets after clustering using our method. 0-mean, Gaussian noise was added to the $x-$, $y-$ and (in the 3-D cases) $z-$ coordinates of the input set. The estimator used is described in [11].

(a) Example Frames

(b) 2-D Isomap Embedding

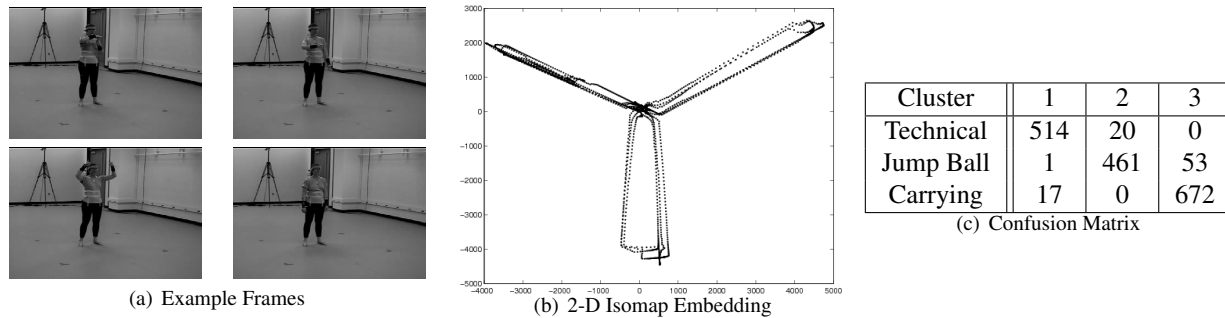| Cluster	1	2	3
Technical	514	20	0
Jump Ball	1	461	53
Carrying	17	0	672

(c) Confusion Matrix

Figure 5. Results on human motion capture data. The actor in the video performed a series of 3 basketball signals (technical foul, jump ball, and carrying) 3 times. (a) The images show examples of each signal plus the actor in the neutral position. (b) The original Isomap embedding of the 2212 frame motion capture data. (c)The confusion matrix describing the clustering results of our algorithm in terms of the number of frames classified as each signal.

[5] J. D. Carroll and J.-J. Chang. Three-way scaling and clustering. *Psychometrika*, 35:238–319, 1970.

[6] D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *PNAS*, 100(10):5591–5596, 2003.

[7] A. Hadid and M. Pietikäinen. Efficient locally linear embeddings of imperfect manifolds. In P. Perner and A. Rosenfeld, editors, *Machine Learning and Data Mining in Pattern Recognition, Third International Conference, MLDM 2003*, pages 188–201, 2003.

[8] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley and Sons, 2001.

[9] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[10] J. B. Kruskal and M. Wish. Multidimensional scaling. *Sage University Paper series on Quantitative Application in the Social Sciences*, pages 07–011, 1978.

[11] E. Levina and P. J. Bickel. Maximum likelihood estimation of intrinsic dimension. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 777–784. MIT Press, Cambridge, MA, 2005.

[12] R. Pless and I. Simon. Embedding images in non-flat spaces. In *Proc. of the International Conference on Imaging Science, Systems, and Technology*, 2002.

[13] S. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, December 2000.

[14] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, 2003.

[15] J. Tenebaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, December 2000.

[16] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-04)*, volume II, pages 988–995, Washington D.C., 2004.

[17] Y. Wu and K. L. Chan. An extended isomap algorithm for learning multi-class manifold. In *Proceedings of IEEE International Conference on Machine Learning and Cybernetics (ICMLC2004)*, Shanghai, China, 2004.