Dimensionality Reduction: A Comparative Review

L.J.P. van der Maaten E.O. Postma H.J. van den Herik

L.J.P. van der Maaten, E.O. Postma, and H.J. van den Herik are with MICC-IKAT, Maastricht University, P.O. Box 616, NL-6200 MD Maastricht, The Netherlands. E-mail:{l.vandermaaten, postma, herik}@micc.unimaas.nl.

Abstract

In recent years, interest in nonlinear dimensionality reduction has grown. This has led to the proposal of various new nonlinear techniques that are claimed to be capable of dealing with complex low-dimensional data. These techniques have been shown to outperform traditional linear techniques on artificial tasks such as the Swiss roll task. Hitherto, the dimensionality reduction techniques have not been compared in a systematic way. In this paper, we discuss and compare ten nonlinear techniques for dimensionality reduction. We investigate the performance of the nonlinear techniques for dimensionality reduction on artificial and natural tasks, and compare their performances to those of the two principal linear techniques: (1) Principal Components Analysis, and (2) Linear Discriminant Analysis. The experiments reveal that nonlinear techniques for dimensionality reduction perform well on selected artificial tasks, but do not outperform the linear techniques on many real-world tasks. The paper concludes that the results suggest that despite their theoretical capability to find complex low-dimensional embeddings, nonlinear techniques for dimensionality reduction are not yet capable of outperforming traditional linear techniques on real-world tasks. We foresee that the performance of nonlinear techniques for dimensionality reduction will be improved by the development of new techniques that represent the global structure of manifolds by a number of separate linear models.

Index Terms

Machine learning, feature extraction or construction, data and knowledge visualization.

I. INTRODUCTION

Dimensionality reduction is the transformation of high-dimensional data into a meaningful representation of reduced dimensionality. Ideally, the reduced representation has a dimensionality that corresponds to the intrinsic dimensionality of the data. The intrinsic dimensionality of data is the minimum number of parameters needed to account for the observed properties of the data [24]. Dimensionality reduction is important in many domains, since it facilitates classification, visualization, and compression of high-dimensional data, by mitigating the curse of dimensionality and other undesired properties of high-dimensional spaces [40].

In recent years, a large number of nonlinear techniques for dimensionality reduction have been proposed [4], [20], [34], [45], [59], [63], [70], [68], [83]. These techniques have the ability to deal with complex nonlinear data and therefore constitute traditional linear techniques for dimensionality reduction, such as Principal Components Analysis (PCA) and Linear Discriminant

Analysis (LDA). In particular for real world data, nonlinear dimensionality reduction techniques may offer an advantage. Previous studies have shown that nonlinear techniques outperform their linear counterparts on artificial tasks that are highly nonlinear. For instance, the Swiss roll dataset comprises a set of points that lie on a spiral-like two-dimensional manifold within a three-dimensional space. A vast number of nonlinear techniques are perfectly able to find this embedding, whereas linear techniques fail to do so. In contrast to these successes on artificial datasets, successful applications of nonlinear dimensionality reduction techniques on natural datasets are scarce. Also, it is not clear to what extent the performances of the various dimensionality reduction techniques differ on artificial and natural tasks.

This paper presents a systematic comparative study of linear and nonlinear dimensionality reduction techniques. It performs both a theoretical and an empirical evaluation of techniques for dimensionality reduction. It investigates the linear techniques PCA [36] and LDA [23] and ten nonlinear techniques: multidimensional scaling (MDS) [15], [43], Isomap [69], [70], Kernel PCA [52], [63], diffusion maps [45], [53], multilayer autoencoders [19], [34], Locally Linear Embedding (LLE) [59], Laplacian Eigenmaps [4], Hessian LLE [20], Local Tangent Space Analysis (LTSA) [83], and Locally Linear Coordination (LLC) [68]. Our comparative review and evaluation includes all main techniques for dimensionality reduction, except self-organizing maps [42] and their probabilistic extension GTM [9], because we consider these techniques to be clustering techniques[1]. Neither do we discuss blind-source separation techniques such as ICA [5]. Because of space limitations, our comparative review does not cover a number of nonlinear techniques, most of which are variants or extensions of the ten reviewed dimensionality reduction techniques. Techniques that are not discussed include principal curves [13], Generalized Discriminant Analysis [3], kernel maps [66], Maximum Variance Unfolding [78], conformal eigenmaps [64], Locality Preserving Projections [30], Linear Local Tangent Space Alignment [81], Stochastic Proximity Embedding [1], FastMap [22], Geodesic Nullspace Analysis [11], and various methods that are based on the global alignment of linear models [10], [58], [76].

The outline of the remainder of this paper is as follows. In section II, we give a formal definition of dimensionality reduction. Section III briefly discusses two linear techniques for dimension-

[1]A theoretical discussion of the link between clustering and dimensionality reduction can be found in [6].

ality reduction. Subsequently, section IV describes and discusses ten nonlinear techniques for dimensionality reduction. Section V evaluates all techniques on theoretical characteristics. Then, in section VI, we present an empirical evaluation of techniques for dimensionality reduction on artificial and natural datasets. Section VII discusses the results of the experiments and identifies weaknesses and points of improvement of the nonlinear techniques for dimensionality reduction. Section VIII concludes on the added value of applying nonlinear dimensionality reduction techniques to real-world tasks.

II. DIMENSIONALITY REDUCTION

The problem of (nonlinear) dimensionality reduction can be defined as follows. Suppose we have a $n \times D$ matrix X consisting of n datavectors x_i with dimensionality D, and that this dataset has the intrinsic dimensionality d (where $d < D$, and often $d \ll D$). In mathematical terms, intrinsic dimensionality means that the points in dataset X are lying on or near a manifold with dimensionality d that is embedded in the D-dimensional space. A dimensionality reduction technique transforms dataset X into a new dataset Y with dimensionality d, while retaining the geometry of the data as much as possible. In general, neither the geometry of the embedded manifold, nor the intrinsic dimensionality d of the dataset X are known. Therefore, dimensionality reduction is an ill-posed problem that can only be solved by assuming certain properties of the data (such as its intrinsic dimensionality).

Techniques for dimensionality reduction can be subdivided into various groups. Figure 1 shows a taxonomy of these techniques. The main distinction between techniques for dimensionality reduction is the distinction between linear and nonlinear techniques. Linear techniques assume that the data lie on or near a linear subspace of the high-dimensional space. Nonlinear techniques for dimensionality reduction do not rely on the linearity assumption as a result of which more complex embeddings of the data in the high-dimensional space can be identified. The further subdivisions in the taxonomy are discussed in the review in the following two sections.

III. LINEAR TECHNIQUES FOR DIMENSIONALITY REDUCTION

In section II, we formally defined the problem of dimensionality reduction, and presented a taxonomy for techniques for dimensionality reduction. In the taxonomy, techniques for dimensionality reduction are subdivided into linear and nonlinear techniques. In this section, we

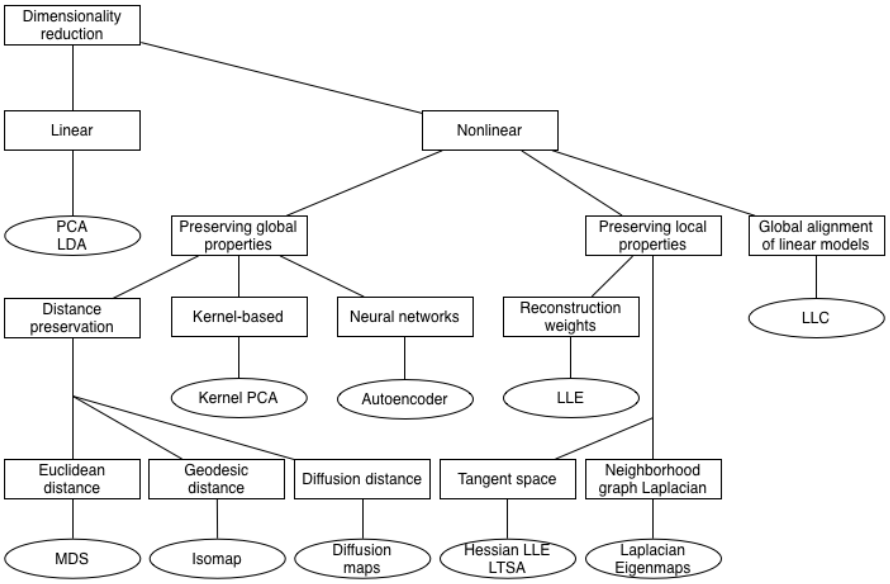


Fig. 1. Taxonomy of dimensionality reduction techniques.

discuss the two main linear techniques for dimensionality reduction: (1) PCA and (2) LDA. We discuss PCA and LDA in subsubsection III-.1 and subsubsection III-.2. Throughout the paper, we denote a high-dimensional datapoint by x_i, where x_i is the ith row of the D-dimensional data matrix X. The low-dimensional counterpart of x_i is denoted by y_i, where y_i is the ith row of the d-dimensional data matrix Y.

1) PCA: Principal Components Analysis (PCA) [36] constructs a low-dimensional representation of the data that describes as much of the variance in the data as possible. This is done by finding a linear basis of reduced dimensionality for the data, in which the amount of variance in the data is maximal.

In mathematical terms, PCA attempts to find a linear mapping M that maximizes $M^T \operatorname{cov}_{X-\bar{X}} M$, where $\operatorname{cov}_{X-\bar{X}}$ is the covariance matrix of the zero mean data X. It can be shown that this linear mapping is formed by the d principal eigenvectors (i.e., principal components) of the covariance matrix of the zero-mean data[2]. Hence, PCA solves the eigenproblem

$$\underset{X-\bar{X}}{\operatorname{cov}} M = \lambda M \tag{1}$$

[2]PCA maximizes $M^T \operatorname{cov}_{X-\bar{X}} M$ with respect to M, under the constraint that $|M| = 1$. This constraint can be enforced by introducing a Lagrange multiplier λ. Hence, an unconstrained maximization of $M^T \operatorname{cov}_{X-\bar{X}} M + \lambda(1 - M^T M)$ is performed. A stationary point of this quantity is to be found when $\operatorname{cov}_{X-\bar{X}} M = \lambda M$.

The eigenproblem is solved for the d principal eigenvalues λ. The low-dimensional data representations y_i of the datapoints x_i are computed by mapping them onto the linear basis M, i.e., $Y = (X - \bar{X})M$.

PCA has been successfully applied in a large number of domains such as face recognition [74] and coin classification [37]. The main drawback of PCA is that the size of the covariance matrix is proportional to the dimensionality of the datapoints. As a result, the computation of the eigenvectors might be infeasible for very high-dimensional data. Approximation methods, such as Simple PCA [56], deal with this problem by applying an iterative Hebbian approach in order to estimate the principal eigenvectors of the covariance matrix.

2) LDA: Linear Discriminant Analysis (LDA) [23] attempts to maximize the linear separability between datapoints belonging to different classes. In contrast to most other dimensionality reduction techniques examined in this paper, LDA is a supervised technique.

LDA finds a linear mapping M that maximizes the linear class separability in the low-dimensional representation of the data. The criteria that are used to formulate linear class separability in LDA are the within-class scatter S_w and the between-class scatter S_b, which are defined as

$$S_w = \sum_c p_c \operatorname*{cov}_{X^c - \bar{X}^c} \tag{2}$$

$$S_b = \operatorname*{cov}_{X - \bar{X}} - S_w \tag{3}$$

where p_c is the class prior of class label c, $\mathrm{cov}_{X^c - \bar{X}^c}$ is the covariance matrix of the zero mean datapoints x_i assigned to class $c \in C$ (where C is the set of possible classes), and $\mathrm{cov}_{X - \bar{X}}$ is the covariance matrix of the zero mean data X. LDA optimizes the ratio between the within-class scatter S_w and the between-class scatter S_b in the low-dimensional representation of the data, by finding a linear mapping M that maximizes the so-called Fisher criterion

$$\phi(M) = \frac{M^T S_b M}{M^T S_w M} \tag{4}$$

This maximization can be performed by computing the d principal eigenvectors of $S_w^{-1} S_b$ (under the requirement $d < |C|$). The low-dimensional data representation Y of the datapoints in X can be computed by mapping them onto the linear basis M, i.e., $Y = (X - \bar{X})M$.

LDA has been successfully applied in a large number of classification tasks. Successful applications include speech recognition [28] and document classification [73].

IV. NONLINEAR TECHNIQUES FOR DIMENSIONALITY REDUCTION

In section III, we discussed the two main linear techniques for dimensionality reduction, which are established and well understood. In contrast, most nonlinear techniques for dimensionality reduction have been proposed more recently and are therefore less well studied. In this section, we discuss ten nonlinear techniques for dimensionality reduction. Nonlinear techniques for dimensionality reduction can be subdivided into three main types[3]: (1) techniques that attempt to preserve global properties of the original data in the low-dimensional representation, (2) techniques that attempt to preserve local properties of the original data in the low-dimensional representation, and (3) techniques that perform global alignment of a number of linear models. In subsection IV-A, we discuss five global nonlinear techniques for dimensionality reduction. Subsection IV-B presents four local nonlinear techniques for dimensionality reduction. Subsection IV-C presents one method that performs global alignment of a number of linear models.

A. Global techniques

Global nonlinear techniques for dimensionality reduction are techniques that attempt to preserve global properties of the data. The subsection presents five global nonlinear techniques for dimensionality reduction: (1) MDS, (2) Isomap, (3) Kernel PCA, (4) diffusion maps, and (5) multilayer autoencoders. The techniques are discussed in subsubsection IV-A.1 to IV-A.5.

1) MDS: Multidimensional scaling (MDS) [15], [43] represents a collection of nonlinear techniques that maps the high-dimensional data representation to a low-dimensional representation while retaining the pairwise distances between the datapoints as much as possible. The quality of the mapping is expressed in the stress function, a measure of the error between the pairwise distances in the low-dimensional and high-dimensional representation of the data. Two important examples of stress functions are the raw stress function and the Sammon cost function. The raw stress function is defined by

$$\phi(Y) = \sum_{ij} (\|x_i - x_j\| - \|y_i - y_j\|)^2 \tag{5}$$

[3]The reader should note that although diffusion maps and Kernel PCA are global methods, they can behave as local methods depending on the choice of the kernel.

in which $\|x_i - x_j\|$ is the Euclidean distance between the high-dimensional datapoints x_i and x_j and $\|y_i - y_j\|$ is the Euclidean distance between the low-dimensional datapoints y_i and y_j. The Sammon cost function is given by

$$\phi(Y) = \frac{1}{\sum_{ij}\|x_i - x_j\|} \sum_{ij} \frac{(\|x_i - x_j\| - \|y_i - y_j\|)^2}{\|x_i - x_j\|} \tag{6}$$

The Sammon cost function differs from the raw stress function in that it puts more emphasis on retaining distances that were originally small. The minimization of the stress function can be performed using various methods, such as the eigendecomposition of a pairwise distance matrix, the conjugate gradient method, or a pseudo-Newton method [15].

MDS is widely used for the visualization of data, e.g., in fMRI analysis [67] and in molecular modelling [75]. The popularity of MDS has led to the proposal of variants such as SPE [1], SNE [33], and FastMap [22].

2) Isomap: Multidimensional scaling has proven to be successful in many applications, but it suffers from the fact that it is based on Euclidean distances, and does not take into account the distribution of the neighboring datapoints. If the high-dimensional data lies on or near a curved manifold, such as in the Swiss roll dataset [69], MDS might consider two datapoints as near points, whereas their distance over the manifold is much larger than the typical interpoint distance. Isomap [69] is a technique that resolves this problem by attempting to preserve pairwise geodesic (or curvilinear) distances between datapoints. Geodesic distance is the distance between two points measured over the manifold.

In Isomap [69], the geodesic distances between the datapoints x_i are computed by constructing a neighborhood graph G, in which every datapoint x_i is connected with its k nearest neighbors x_{i_j} in the dataset X. The shortest path between two points in the graph forms a good (over)estimate of the geodesic distance between these two points, and can easily be computed using Dijkstra's shortest-path algorithm. The geodesic distances between all datapoints in X are computed, thereby forming a pairwise geodesic distance matrix. The low-dimensional representations y_i of the datapoints x_i in the low-dimensional space Y are computed by applying multidimensional scaling (see subsection IV-A.1) on the resulting distance matrix.

An important weakness of the Isomap algorithm is its topological instability [2]. Isomap may construct erroneous connections in the neighborhood graph G. Such short-circuiting [47] can severely impair the performance of Isomap. Several approaches were proposed to overcome the

problem of short-circuiting, e.g., by removing datapoints with large total flows in the shortest path-algorithm [14] or by removing nearest neighbors that violate local linearity of the neighborhood graph [62]. Furthermore, Isomap may suffer from holes in the manifold, a problem that can be dealt with by tearing manifolds with holes [47]. A third weakness of Isomap is that it can fail if the manifold is nonconvex [70]. Despite these weaknesses, Isomap was successfully applied on tasks such as wood inspection [54], and visualization of biomedical data [49].

3) Kernel PCA: Kernel PCA (KPCA) is the reformulation of traditional linear PCA in a high-dimensional space that is constructed using a kernel function [63]. In recent years, the reformulation of linear techniques using the 'kernel trick' has led to the proposal of successful techniques such as kernel ridge regression and Support Vector Machines [65]. Kernel PCA computes the principal eigenvectors of the kernel matrix, rather than those of the covariance matrix. The reformulation of traditional PCA in kernel space is straightforward, since a kernel matrix is similar to the inproduct of the datapoints in the high-dimensional space that is constructed using the kernel function. The application of PCA in kernel space provides Kernel PCA the property of constructing nonlinear mappings.

Kernel PCA computes the kernel matrix K of the datapoints x_i. The entries in the kernel matrix are defined by

$$k_{ij} = \kappa(x_i, x_j) \tag{7}$$

where κ is a kernel function [65]. Subsequently, the kernel matrix K is centered using the following modification of the entries

$$k_{ij} = k_{ij} - \frac{1}{n}\sum_l k_{il} - \frac{1}{n}\sum_l k_{jl} + \frac{1}{n^2}\sum_{lm} k_{lm} \tag{8}$$

The centering operation corresponds to subtracting the mean of the features in traditional PCA. It makes sure that the features in the high-dimensional space defined by the kernel function are zero-mean. Subsequently, the principal d eigenvectors v_i of the centered kernel matrix are computed. It can be shown that the eigenvectors of the covariance matrix α_i (in the high-dimensional space constructed by κ) are scaled versions of the eigenvectors of the kernel matrix v_i

$$\alpha_i = \frac{1}{\sqrt{\lambda_i}} v_i \tag{9}$$

In order to obtain the low-dimensional data representation, the data is projected onto the eigenvectors of the covariance matrix α_i. The result of the projection (i.e., the low-dimensional data

representation Y) is given by

$$Y = \left\{ \sum_j \alpha_1 \kappa(x_j, x), \sum_j \alpha_2 \kappa(x_j, x), ..., \sum_j \alpha_d \kappa(x_j, x) \right\} \tag{10}$$

where κ is the kernel function that was also used in the computation of the kernel matrix. Since Kernel PCA is a kernel-based method, the mapping performed by Kernel PCA highly relies on the choice of the kernel function κ. Possible choices for the kernel function include the linear kernel (making Kernel PCA equal to traditional PCA), the polynomial kernel, and the Gaussian kernel [65].

Kernel PCA has been successfully applied to, e.g., speech recognition [50], and novelty detection [35]. An important drawback of Kernel PCA is that the size of the kernel matrix is square with the number of instances in the dataset. An approach to resolve this drawback is proposed in [72].

4) Diffusion maps: The diffusion maps (DM) framework [45], [53] originates from the field of dynamical systems. Diffusion maps are based on defining a Markov random walk on the graph of the data. By performing the random walk for a number of timesteps, a measure for the proximity of the datapoints is obtained. Using this measure, the so-called diffusion distance is defined. In the low-dimensional representation of the data, the pairwise diffusion distances are retained as well as possible.

In the diffusion maps framework, a graph of the data is constructed first. The weights of the edges in the graph are computed using the Gaussian kernel function, leading to a matrix W with entries

$$w_{ij} = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}} \tag{11}$$

where σ indicates the variance of the Gaussian. Subsequently, normalization of the matrix W is performed in such a way that its rows add up to 1. In this way, a matrix $P^{(1)}$ is formed with entries

$$p_{ij}^{(1)} = \frac{w_{ij}}{\sum_k w_{ik}} \tag{12}$$

Since diffusion maps originate from dynamical systems theory, the resulting matrix $P^{(1)}$ is considered a Markov matrix that defines the forward transition probability matrix of a dynamical process. Hence, the matrix $P^{(1)}$ represents the probability of a transition from one datapoint to another datapoint in a single timestep. The forward probability matrix for t timesteps $P^{(t)}$ is

given by $(P^{(1)})^t$. Using the random walk forward probabilities $p_{ij}^{(t)}$, the diffusion distance is defined by

$$D^{(t)}(x_i, x_j) = \sum_k \frac{(p_{ik}^{(t)} - p_{jk}^{(t)})^2}{\psi^{(0)}(x_k)} \tag{13}$$

In the equation, $\psi^{(0)}(x_i)$ is a term that attributes more weight to parts of the graph with high density. It is defined by $\psi^{(0)}(x_i) = \frac{m_i}{\sum_j m_j}$, where m_i is the degree of node x_i defined by $m_i = \sum_j p_{ji}$. From Equation 13, it can be observed that pairs of datapoints with a high forward transition probability have a small diffusion distance. The key idea behind the diffusion distance is that it is based on many paths through the graph. This makes the diffusion distance more robust to noise than, e.g., the geodesic distance. In the low-dimensional representation of the data Y, diffusion maps attempt to retain the diffusion distances. Using spectral theory on the random walk, it can be shown[4] that the low-dimensional representation Y that retains the diffusion distances is formed by the d nontrivial principal eigenvectors of the eigenproblem

$$P^{(t)}Y = \lambda Y \tag{14}$$

Because the graph is fully connected, the largest eigenvalue is trivial (viz., $\lambda_1 = 1$), and its eigenvector v_1 is thus discarded. The low-dimensional representation Y is given by the next d principal eigenvectors. In the low-dimensional representation, the eigenvectors are normalized by their corresponding eigenvalues. Hence, the low-dimensional data representation is given by

$$Y = \{\lambda_2 v_2, \lambda_3 v_3, ..., \lambda_{d+1} v_{d+1}\} \tag{15}$$

5) Multilayer autoencoders: Multilayer encoders are feed-forward neural networks with an odd number of hidden layers [19], [34]. The middle hidden layer has d nodes, and the input and the output layer have D nodes. An example of an autoencoder is shown schematically in Figure 2. The network is trained to minimize the mean squared error between the input and the output of the network (ideally, the input and the output are equal). Training the neural network on the datapoints x_i leads to a network in which the middle hidden layer gives a d-dimensional representation of the datapoints that preserves as much information in X as possible. The low-dimensional representations y_i can be obtained by extracting the node values in the middle hidden layer, when datapoint x_i is used as input. If linear activation functions are used in the neural

[4]See [45] for the derivation.

network, an autoencoder is very similar to PCA [44]. In order to allow the autoencoder to learn a nonlinear mapping between the high-dimensional and low-dimensional data representation, sigmoid activation functions are generally used.

Multilayer autoencoders usually have a high number of connections. Therefore, backpropagation approaches converge slowly and are likely to get stuck in local minima. In [34], this drawback is overcome by performing a pretraining using Restricted Boltzmann Machines (RBMs) [32]. RBMs are neural networks of which the units are binary and stochastic, and in which connections between hidden units are not allowed. RBMs can successfully be used to train neural networks with many hidden layers using a learning approach based on simulated annealing. Once the pretraining using RBMs is performed, finetuning of the network weights is performed using backpropagation. As an alternative approach, genetic algorithms [57] can be used to train an autoencoder.

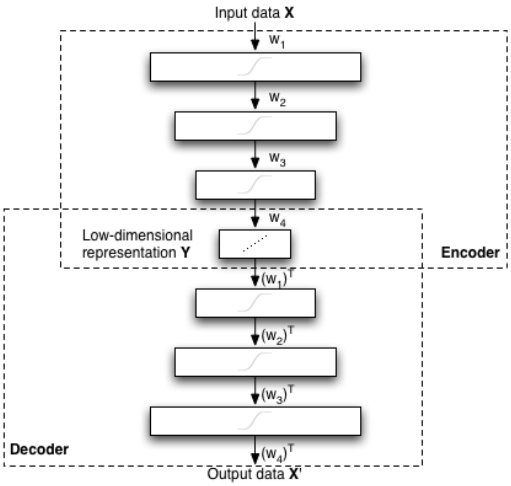


Fig. 2. Schematic structure of an autoencoder.

B. Local techniques

Subsection IV-A presented five techniques for dimensionality reduction that attempt to retain global properties of the data. In contrast, local nonlinear techniques for dimensionality reduction are based on solely preserving properties of small neighborhoods around the datapoints. This subsection presents four local nonlinear techniques for dimensionality reduction: (1) LLE, (2) Laplacian Eigenmaps, (3) Hessian LLE, and (4) LTSA in subsubsection IV-B.1 to IV-B.4. The central claim of these techniques is that by preservation of local properties of the data, the global

layout of the data manifold is retained as well. In essence, local techniques for dimensionality reduction can be viewed upon as definitions of specific local kernel functions for Kernel PCA. Therefore, these techniques can be rewritten within the Kernel PCA framework [6], [29]. In this paper, we do not elaborate further on the theoretical connection between local methods for dimensionality reduction and Kernel PCA.

1) LLE: Local Linear Embedding (LLE) [59] is a local technique for dimensionality reduction that is similar to Isomap in that it constructs a neighborhood graph representation of the datapoints. In contrast to Isomap, it attempts to preserve solely local properties of the data, making LLE less sensitive to short-circuiting than Isomap. Furthermore, the preservation of local properties allows for successful embedding of nonconvex manifolds. In LLE, the local properties of the data manifold are constructed by writing the datapoints as a linear combination of their nearest neighbors. In the low-dimensional representation of the data, LLE attempts to retain the reconstruction weights in the linear combinations as well as possible.

LLE describes the local properties of the manifold around a datapoint x_i by writing the datapoint as a linear combination W_i (the so-called reconstruction weights) of its k nearest neighbors x_{i_j}. Hence, LLE fits a hyperplane through the datapoint x_i and its nearest neighbors, thereby assuming that the manifold is locally linear. The local linearity assumption implies that the reconstruction weights W_i of the datapoints x_i are invariant to translation, rotation, and rescaling. Because of the invariance to these transformations, any linear mapping of the hyperplane to a space of lower dimensionality preserves the reconstruction weights in the space of lower dimensionality. In other words, if the low-dimensional data representation preserves the local geometry of the manifold, the reconstruction weights W_i that reconstruct datapoint x_i from its neighbors in the high-dimensional data representation also reconstruct datapoint y_i from its neighbors in the low-dimensional data representation. As a consequence, finding the d-dimensional data representation Y amounts to minimizing the cost function

$$\phi(Y) = \sum_i (y_i - \sum_{j=1}^{k} w_{ij} y_{i_j})^2 \tag{16}$$

It can be shown[5] that the coordinates of the low-dimensional representations y_i that minimize this cost function can be found by computing the eigenvectors corresponding to the smallest d nonzero eigenvalues of the inproduct of $(I - W)$. In this formula, I is the $n \times n$ identity matrix. LLE has been successfully applied for, e.g., superresolution [12] and sound source localization [21]. However, there also exist experimental studies that report weak performance of LLE. In [49], LLE was reported to fail in the visualization of even simple synthetic biomedical datasets. In [39], it is claimed that LLE performs worse than Isomap in the derivation of perceptual-motor actions. A possible explanation lies in the difficulties that LLE has when confronted with manifolds that contain holes [59].

2) Laplacian Eigenmaps: Similar to LLE, Laplacian Eigenmaps find a low-dimensional data representation by preserving local properties of the manifold [4]. In Laplacian Eigenmaps, the local properties are based on the pairwise distances between near neighbors. Laplacian Eigenmaps compute a low-dimensional representation of the data in which the distances between a datapoint and its k nearest neighbors are minimized. This is done in a weighted manner, i.e., the distance in the low-dimensional data representation between a datapoint and its first nearest neighbor contributes more to the cost function than the distance between the datapoint and its second nearest neighbor. Using spectral graph theory, the minimization of the cost function is defined as an eigenproblem.

The Laplacian Eigenmap algorithm first constructs a neighborhood graph G in which every datapoint x_i is connected to its k nearest neighbors. For all points x_i and x_j in graph G that are connected by an edge, the weight of the edge is computed using the Gaussian kernel function (see Equation 11), leading to a sparse adjacency matrix W. In the computation of the low-dimensional representations y_i, the cost function that is minimized is given by

$$\phi(Y) = \sum_{ij} (y_i - y_j)^2 w_{ij} \tag{17}$$

In the cost function, large weights w_{ij} correspond to small distances between the datapoints x_i and x_j. Hence, the difference between their low-dimensional representations y_i and y_j highly contributes to the cost function. As a consequence, nearby points in the high-dimensional space

[5] $\phi(Y) = (Y - WY)^2 = Y^T (I - W)^T (I - W) Y$ is the function that has to be minimized. Hence, the eigenvectors of $(I - W)^T (I - W)$ corresponding to the smallest nonzero eigenvalues form the solution that minimizes $\phi(Y)$.

are brought closer together in the low-dimensional representation.

The computation of the degree matrix M and the graph Laplacian L of the graph W allows for formulating the minimization problem as an eigenproblem. The degree matrix M of W is a diagonal matrix, whose entries are the row sums of W (i.e., $m_{ii} = \sum_j w_{ij}$). The graph Laplacian L is computed by $L = M - W$. It can be shown that the following holds[6]

$$\phi(Y) = \sum_{ij} (y_i - y_j)^2 w_{ij} = 2Y^T LY \tag{18}$$

Hence, minimizing $\phi(Y)$ is proportional to minimizing $Y^T LY$. The low-dimensional data representation Y can thus be found by solving the generalized eigenvector problem

$$Lv = \lambda Mv \tag{19}$$

for the d smallest nonzero eigenvalues. The d eigenvectors v_i corresponding to the smallest nonzero eigenvalues form the low-dimensional data representation Y.

Laplacian Eigenmaps have been successfully applied to, e.g., clustering [80] and face recognition [31].

3) Hessian LLE: Hessian LLE (HLLE) [20] is a variant of LLE that minimizes the 'curviness' of the high-dimensional manifold when embedding it into a low-dimensional space, under the constraint that the low-dimensional data representation is locally isometric. This is done by the eigenanalysis of a matrix $\mathcal{H}$ that describes the curviness of the manifold around the datapoints. The curviness of the manifold is measured by means of the local Hessian at every datapoint. The local Hessian is represented in the local tangent space at the datapoint, in order to obtain a representation of the local Hessian that is invariant to differences in the positions of the datapoints. It can be shown[7] that the coordinates of the low-dimensional representation can be found by performing an eigenanalysis of $\mathcal{H}$.

Hessian LLE starts with identifying the k nearest neighbors for each datapoint x_i using Euclidean distance. In the neighborhood, local linearity of the manifold is assumed. Hence, a basis for the local tangent space at point x_i can be found by applying PCA on its k nearest neighbors x_{i_j}. In other words, for every datapoint x_i, a basis for the local tangent space at point x_i is determined

[6]Note that $\phi(Y) = \sum_{ij}(y_i - y_j)^2 w_{ij} = \sum_{ij}(y_i^2 + y_j^2 - 2y_i y_j)w_{ij} = \sum_i y_i^2 m_{ii} + \sum_j y_j^2 m_{jj} - 2\sum_{ij} y_i y_j w_{ij} = 2Y^T MY - 2Y^T WY = 2Y^T LY$

[7]The derivation is too extensive for this paper, but can be found in [20].

by computing the d principal eigenvectors $M = \{m_1, m_2, ..., m_d\}$ of the covariance matrix $\text{cov}_{x_{i_j} - \overline{x}_{i_j}}$. Note that the above requires that $k \geq d$. Subsequently, an estimator for the Hessian of the manifold at point x_i in local tangent space coordinates is computed. In order to do this, a matrix Z_i is formed that contains (in the columns) all cross products of M up to the dth order (including a column with ones). The matrix Z_i is orthonormalized by applying Gram-Schmidt orthonormalization on the matrix Z_i. The estimation of the tangent Hessian H_i is now given by the transpose of the last $\frac{d(d+1)}{2}$ columns of the matrix Z_i. Using the Hessian estimators in local tangent coordinates, a matrix $\mathcal{H}$ is constructed with entries

$$\mathcal{H}_{lm} = \sum_i \sum_j ((H_i)_{jl} \times (H_i)_{jm}) \tag{20}$$

The matrix $\mathcal{H}$ represents information on the curviness of the high-dimensional data manifold. An eigenanalysis of $\mathcal{H}$ is performed in order to find the low-dimensional data representation that minimizes the curviness of the manifold. The eigenvectors corresponding to the d smallest nonzero eigenvalues of $\mathcal{H}$ are selected and form the matrix Y, which contains the low-dimensional representation of the data.

4) LTSA: Similar to Hessian LLE, Local Tangent Space Analysis (LTSA) is a technique that describes local properties of the high-dimensional data using the local tangent space of each datapoint [83]. LTSA is based on the observation that, if local linearity of the manifold is assumed, there exists a linear mapping from a high-dimensional datapoint to its local tangent space, and that there exists a linear mapping from the corresponding low-dimensional datapoint to the same local tangent space [83]. LTSA attempts to align these linear mappings in such a way, that they construct the local tangent space of the manifold from the low-dimensional representation. In other words, LTSA simultaneously searches for the coordinates of the low-dimensional data representations, and for the linear mappings of the low-dimensional datapoints to the local tangent space of the high-dimensional data.

Similar to Hessian LLE, LTSA starts with computing bases for the local tangent spaces at the datapoints x_i. This is done by applying PCA on the k datapoints x_{i_j} that are neighbors of datapoint x_i. This results in a mapping M_i from the neighborhood of x_i to the local tangent space Θ_i. A property of the local tangent space Θ_i is that there exists a linear mapping L_i from the local tangent space coordinates θ_{i_j} to the low-dimensional representations y_{i_j}. Using this

property of the local tangent space, LTSA performs the following minimization

$$\min_{Y_i, L_i} \sum_i \parallel Y_i J_k - L_i \Theta_i \parallel^2 \tag{21}$$

where J_k is the centering matrix of size k [65]. It can be shown[8] that the solution of the minimization is formed by the eigenvectors of an alignment matrix B that correspond to the d smallest nonzero eigenvalues of B. The entries of the alignment matrix B are obtained by iterative summation (for all matrices V_i and starting from $b_{ij} = 0$ for $\forall ij$)

$$B_{N_i N_i} = B_{N_i N_i} + J_k(I - V_i V_i^T)J_k \tag{22}$$

where N_i is a selection matrix that contains the indices of the nearest neighbors of datapoint x_i. Subsequently, the low-dimensional representation Y is obtained by computation of the eigenvectors corresponding to the d smallest nonzero eigenvectors of the symmetric matrix $\frac{1}{2}(B + B^T)$. In [71], a successful application of LTSA to microarray data is reported.

C. Global alignment of linear models

In the previous subsections, we discussed techniques that compute a low-dimensional data representation by preserving global or local properties of the data. Techniques that perform global alignment of linear models compute a number of linear models and construct a low-dimensional data representation by aligning these linear models. In subsubsection IV-C.1 we present one such technique, viz., LLC.

1) LLC: Locally Linear Coordination (LLC) [68] computes a mixture of factor analyzers and subsequently performs a global alignment of the mixture of linear models. This process consists of two steps: (1) computing a mixture of factor analyzers on the data by means of an Expectation Maximization (EM) algorithm and (2) aligning the linear models in order to obtain the low-dimensional data representation using a variant of LLE. A similar technique called manifold charting was proposed in [11].

LLC first constructs a mixture of m factor analyzers using the EM-algorithm [25]. The mixture of factor analyzers outputs m local data representations z_{ij} and corresponding responsibilities r_{ij} (where $j \in \{1, ..., m\}$) for every datapoint x_i. The responsibility r_{ij} describes to what extent

[8]The proof is too extensive for this paper, but can be found in [83].

datapoint x_i corresponds to the linear model j, and satisfies $\sum_j r_{ij} = 1$. Using the linear models and the corresponding responsibilities, responsibility-weighted data representations $u_{ij} = r_{ij} z_{ij}$ are computed. The responsibility-weighted data representations u_{ij} are stored in a $n \times mD$ block matrix U. The alignment of the linear models is performed based on U and on a matrix M that is given by $M = (I - W)^T (I - W)$. Herein, the matrix W contains the reconstruction weights computed by LLE (see subsubsection IV-B.1), and I denotes the $n \times n$ identity matrix. LLC aligns the linear models by solving the generalized eigenproblem

$$Av = \lambda Bv \tag{23}$$

for the d smallest nonzero eigenvalues[9]. In the equation, A is the inproduct of $M^T U$ and B is the inproduct of U. The d eigenvectors v_i form a matrix L, that can be shown to define a linear mapping from the responsibility-weighted data representation U to the underlying low-dimensional data representation Y. The low-dimensional data representation is thus obtained by computing $Y = UL$.

V. Characterization of the techniques

In the previous two sections, we provided an overview of techniques for dimensionality reduction. This section evaluates the techniques by four theoretical characterizations. First, we evaluate four general properties and underlying assumptions of the techniques (subsection V-A). Second, we evaluate the computational complexities of the techniques, as well as their memory complexities (subsection V-B). Third, the out-of-sample extension of the techniques is discussed (subsection V-C). Fourth, we evaluate the exploitation of class label information by techniques for dimensionality reduction (subsection V-D). In all evaluations, we do not take MDS into account, because MDS does not represent one, but a collection of metric and nonmetric techniques.

A. General properties

Table I lists four general properties of the techniques that are of relevance to: (1) whether the technique assumes the data to be sampled dense, (2) whether proximities can be used as inputs instead of the data itself, (3) whether the optimization problem is convex, and (4) the free

[9]The derivation of this eigenproblem can be found in [68].

Technique	*Dense*	*Proximity*	*Convexity*	*Parameters*
PCA/LDA	no	no	yes	none
Isomap	yes	yes	yes	k
Kernel PCA	no	no	yes	$\kappa(\cdot,\cdot)$
Diffusion maps	no	yes	yes	σ, t
Autoencoders	no	no	no	net size
LLE	yes	no	yes	k
Laplacian Eigenmaps	yes	yes	yes	k, σ
Hessian LLE	yes	no	yes	k
LTSA	yes	no	yes	k
LLC	yes	no	no	m, k

TABLE I

PROPERTIES OF TECHNIQUES FOR DIMENSIONALITY REDUCTION.

parameters that have to be optimized. We briefly discuss the four general properties below.

First, the general properties described in Table I reveal that all techniques for dimensionality reduction that employ a neighborhood graph require that the data are sampled dense, because of the local linearity assumption these techniques are submitted to. Dense sampling indicates that the local linearity assumption is valid (to a certain extent) with respect to the curvature and dimensionality of the manifold. This property is a disadvantage, because in realistic applications the amount of data is often scarce. As a result, nonlinear techniques that employ neighborhood graphs in the dimensionality reduction are likely to have low performance on data that is poorly sampled.

Second, Table I reveals that some nonlinear techniques allow for using proximities as input. The property of allowing proximities as inputs extends the variety of tasks in which the technique is applicable to, e.g., tasks where the data is discrete but a proper distance metric exists. In this respect, some nonlinear techniques have an advantage over linear techniques for dimensionality reduction.

Third, Table I reveals that most techniques for dimensionality reduction optimize a convex cost function, which is advantageous, because it allows for finding the global optimum of the cost function. Because of their nonconvex cost functions, LLC and autoencoders may suffer from getting stuck in local optima.

Fourth, Table I shows that the nonlinear techniques for dimensionality reduction all have free parameters that have to be optimized. By free parameters, we mean parameters that directly influence the cost function that is optimized. The reader should note that iterative techniques

for dimensionality reduction have additional free parameters such as the learning rate and the maximum number of iterations. Not surprisingly, Table I shows that linear techniques for dimensionality reduction have an advantage over nonlinear techniques with respect to the number of free parameters, because the selection of proper parameters is a problem in itself.
Taken together, Table I shows that nonlinear techniques for dimensionality reduction have one or more disadvantageous general properties compared to linear techniques. In contrast, some nonlinear techniques have the advantage of allowing proximities as input.

B. Computational complexity

The computational complexity of a dimensionality reduction technique is of importance to its applicability. If the memory or computational resources necessary grow too large, application becomes infeasible. The computational complexity of a dimensionality reduction technique is determined by the number of datapoints n, the original dimensionality D, the target dimensionality d, and by parameters such as the number of nearest neighbors k (for techniques based on neighborhood graphs) and the number of iterations i (for iterative techniques). In Table II, we provide an overview of the computational and memory complexities of the main parts of the techniques. In the table, p denotes the ratio of nonzero elements in a sparse matrix, m indicates the number of models in a mixture of factor analyzers, and w is the number of weights in a neural network. Below, the complexities in Table II are clarified.
The computation of the covariance matrix required in both PCA and LDA has a computational complexity of $O(nD)$. The eigenanalysis of the $D \times D$ covariance matrix required is performed using a power method in $O(D^3)$. Because PCA and LDA store a $D \times D$ covariance matrix, their memory complexity is $O(D^2)$.
Isomap, diffusion maps, and Kernel PCA perform an eigenanalysis of an $n \times n$ matrix using a power method in $O(n^3)$. Isomap performs n additional nearest neighbor searches, which has computational complexity $O(Dn \log n)$ [41], and it performs Dijkstra's algorithm on the neighborhood graph in $O(nk + n \log n)$ (using a Fibonacci heap implementation). Diffusion maps, and Kernel PCA perform an additional kernel computation in $O(Dn^2)$ (for the Gaussian kernel). Because Isomap, diffusion maps, and Kernel PCA store a full $n \times n$ kernel matrix, the memory complexity of these techniques is $O(n^2)$.
In contrast to the spectral techniques discussed above, autoencoders are iterative. Training an

autoencoder using backpropagation is performed in $O(inw)$. The training of autoencoders may converge very slowly, especially in cases where the input and target dimensionality are very high (since this yields a high number of weights in the network). The memory complexity of an autoencoder is $O(w)$.

Similar to, e.g., Kernel PCA, local techniques for dimensionality reduction perform an eigenanalysis of an $n \times n$ matrix. However, for local techniques the $n \times n$ matrix is sparse. The sparsity of the matrices is beneficial, because it lowers the computational complexity of the eigenanalysis. Eigenanalysis of a sparse matrix (using Arnoldi or Jacobi-Davidsson methods) has computational complexity $O(pn^2)$, where p is the ratio of nonzero to zero elements in the sparse matrix. For LLE and Laplacian Eigenmaps, the value of p is typically lower than for Hessian LLE and LTSA, making the former methods computationally less intensive. In addition to the eigenanalysis of the sparse weight matrix, LLE, Laplacian Eigenmaps, Hessian LLE, and LTSA construct a nearest neighbor graph in $O(Dn \log n)$. Furthermore, LLE solves n systems of linear equations of size $k \times k$ in $O(nk^3)$. Laplacian Eigenmaps perform a sparse Gaussian kernel computation in $O(pnD)$. Hessian LLE and LTSA compute the eigendecomposition of n matrices with size $k \times k$ in $O(nk^3)$. In addition, Hessian LLE performs n Gram-Schmidt orthogonalizations of a $k \times D$ matrix, which has computational complexity $O(nkD^2)$.

LLC is a technique that incorporates both an iterative and an spectral part. First, LLC performs the EM-algorithm for i iterations in order to construct a mixture of m factor analyzers. Hence, the computational complexity of this part is bounded by $O(imD^3)$. Subsequently, LLC performs the identification of k nearest neighbors in $O(n \log n)$, and it solves a generalized eigenproblem of size $k \times k$ in $O(pk^2)$. Because the EM algorithm stores data representations for m factor analyzers, the memory complexity of LLC is $O(mnd)$.

From the evaluation of the computational and memory complexity of techniques for dimensionality reduction above, we observe nonlinear techniques to have computational disadvantages compared to linear techniques (if we assume that $D < n$). Furthermore, a number of nonlinear techniques suffer from a memory complexity that is square with the number of datapoints n. Attempts to reduce the computational and/or memory complexities of nonlinear techniques have been proposed for, e.g., Isomap [18] and Kernel PCA [72].

Technique	*Computational*	*Memory*
PCA/LDA	$O(nD) + O(D^3)$	$O(D^2)$
Isomap	$O(Dn \log n) + O(nk + n \log n) + O(n^3)$	$O(n^2)$
Kernel PCA	$O(Dn^2) + O(Dn^3)$	$O(n^2)$
Diffusion maps	$O(Dn^2) + O(n^3)$	$O(n^2)$
Autoencoders	$O(inw)$	$O(w)$
LLE	$O(Dn \log n) + O(nk^3) + O(pn^2)$	$O(pn^2)$
Laplacian Eigenmaps	$O(Dn \log n) + O(pnD) + O(pn^2)$	$O(pn^2)$
Hessian LLE	$O(Dn \log n) + O(nk^3) + O(nkD^2) + O(pn^2)$	$O(pn^2)$
LTSA	$O(Dn \log n) + O(nk^3) + O(pn^2)$	$O(pn^2)$
LLC	$O(imD^3) + O(Dn \log n) + O(pk^2)$	$O(mnd)$

TABLE II

COMPUTATIONAL AND MEMORY COMPLEXITIES.

C. Out-of-sample extension

For the linear techniques PCA and LDA, the out-of-sample extension (i.e., how to embed a new datapoint in the low-dimensional space) is straightforward. In PCA and LDA, the out-of-sample extension is computed by multiplying the new datapoint with the linear mapping matrix M. A similar approach is also viable for Kernel PCA. For autoencoders, out-of-sample extension can readily be performed, since the trained network defines the transformation from the high-dimensional to the low-dimensional data representation.

For the other nonlinear dimensionality reduction techniques the out-of-sample extension is not straightforward. A method for the out-of-sample extensions of Isomap, LLE, and Laplacian Eigenmaps has been presented [8] in which the techniques are redefined in the Kernel PCA framework, thereby allowing for out-of-sample extension. Similar methods for the out-of-sample extension of Isomap are proposed in [14], [18]. An estimation method for out-of-sample extension that can be applied to all nonlinear dimensionality reduction techniques is proposed in [48]. The method finds the nearest neighbor of the new datapoint in the high-dimensional representation, and computes the linear mapping from the nearest neighbor to its corresponding low-dimensional representation. The low-dimensional representation of the new datapoint is found by applying the same linear mapping on this datapoint.

From the evaluation above, we observe that linear and nonlinear techniques for dimensionality reduction are similar in that they allow for out-of-sample extension. However, for a number of nonlinear techniques, out-of-sample extension can only be performed using estimation methods, which leads to estimation errors in the out-of-sample extension.

D. Supervision

In many real-world applications, supervised learning settings are required. The extent to which dimensionality reduction techniques support supervised learning is of importance for such applications. Techniques for dimensionality reduction can exploit the information in the class labels by means of the Fisher criterion (as is done in LDA). In Generalized Discriminant Analysis [3], the Fisher criterion is maximized in a high-dimensional space constructed by a kernel function (similar to Kernel PCA), thereby allowing for nonlinear mappings. An attempt to incorporate the Fisher criterion in a variant of MDS is proposed in [84]. However, the method presented in [84] suffers from the difficulty of minimizing a complex nonconvex cost function, leading to inferior results. In [60], a supervised variant of autoencoders is claimed to have a strong performance. Attempts to exploit class labels in local nonlinear techniques have been proposed for LLE [17] and LTSA [48]. In both techniques, the similarity matrix Δ is replaced by a similarity matrix Δ' that contains information on both the local properties of the data and on the class labels of the datapoints. A drawback of this approach is that it does not allow for proper out-of-sample extension.

From the evaluation above, we observe that linear and nonlinear are similar in that they allow for exploiting the class labels of the data. The Fisher criterion can be maximized in nonlinear techniques, and supervised autoencoders are claimed to perform strongly on certain tasks. However, attempts to incorporate supervision into local nonlinear techniques are generally inapplicable due to the lack of a proper out-of-sample extension.

VI. EXPERIMENTS

In the previous section, we evaluated theoretical properties of the linear and nonlinear techniques for dimensionality reduction. We showed that linear techniques for dimensionality reduction have advantages over nonlinear techniques with respect to one or more of the following five properties: (1) underlying assumptions, (2) number of free parameters, (3) computational complexity, (4) out-of-sample extension, and (5) supervision. In this section, a systematic empirical comparison of the performance of linear and nonlinear techniques for dimensionality reduction is performed. We perform the evaluation by measuring generalization errors in classification tasks on two types of datasets: (1) artificial datasets and (2) natural datasets.

The setup of our experiments is described in subsection VI-A. In subsection VI-B, the results of

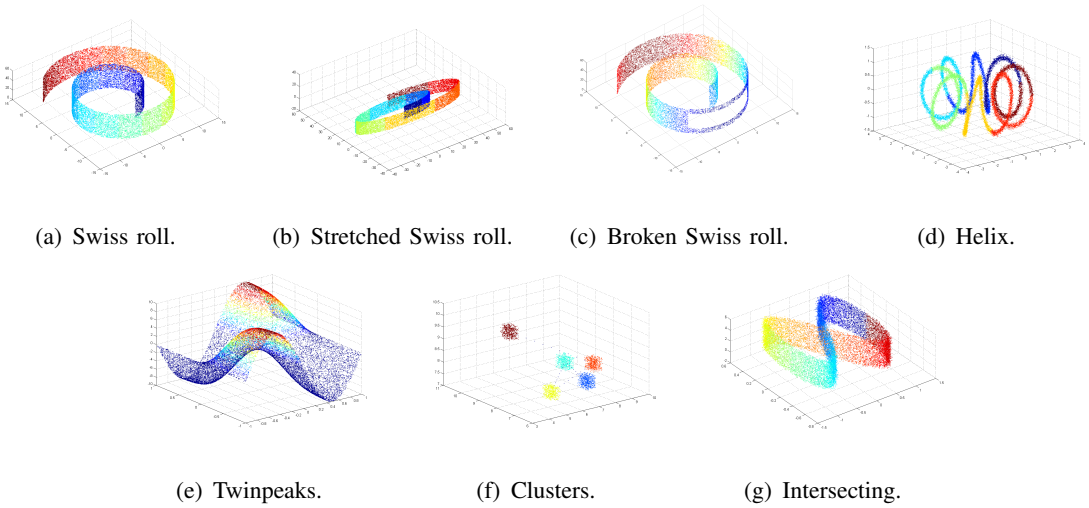
our experiments on seven artificial datasets are presented. Subsection VI-C presents the results of the experiments on seven natural datasets.

A. Experimental setup

In our experiments, we apply the techniques for dimensionality reduction on the high-dimensional representation of the data, and evaluate the generalization performance of various classifiers on the obtained low-dimensional representation of the data. Our motivation for this classification-based evaluation of the techniques instead of one based on measuring reconstruction errors is that for most natural tasks, the true low-dimensional data representation is unknown.

We performed experiments on seven artificial datasets and seven natural datasets. The artificial datasets on which we performed experiments are: (1) the Swiss roll dataset, (2) the stretched Swiss roll dataset, (3) the broken Swiss roll dataset, (4) the helix dataset, (5) the twin peaks dataset, (6) the clusters dataset, and (7) the intersecting dataset. Figure 3 shows plots of the seven artificial datasets. Datapoints belonging to the same class are assigned the same colors. All artificial datasets (except for the clusters dataset) consist of 20,000 samples. The clusters dataset consists of 3,000 samples. The results of the experiments on the artificial datasets allow for investigating how the techniques perform on datasets with changing curvature of the manifold, holes in the manifold, intersections in the manifold, and how the techniques deal with the presence of a number of small manifolds.

For our experiments on natural datasets, we selected seven datasets that represent tasks from a variety of domains: (1) the MNIST dataset, (2) the COIL20 dataset, (3) the ADA dataset, (4) the GINA dataset, (5) the HIVA dataset, (6) the NOVA dataset, and (7) the SYLVA dataset. The MNIST dataset is a dataset of 60,000 handwritten digits of which we randomly selected 20,000 digits for our experiments. Since the images in the MNIST dataset have size 28×28, they can be considered as points in a 784-dimensional space. The COIL20 dataset contains images of 20 different objects, depicted from 72 viewpoints. The size of the images is 32×32 pixels, leading to a 1,024-dimensional space. The ADA, GINA, HIVA, NOVA, and SYLVA datasets are binary classification datasets, which are used in a recent NIPS benchmark competition. The ADA dataset originates from the marketing domain, and consists of 4,147 datapoints in a 48-dimensional space. The GINA dataset is a handwriting recognition dataset that has 3,153 datapoints described by 970 features. The HIVA dataset is a chemical dataset with 3,845 datapoints in 1,617 dimensions.



(a) Swiss roll. (b) Stretched Swiss roll. (c) Broken Swiss roll. (d) Helix.

(e) Twinpeaks. (f) Clusters. (g) Intersecting.

Fig. 3. Artificial datasets.

The NOVA dataset is a text classification dataset with 1,754 samples in a 16,969-dimensional space. The SYLVA dataset originates from the ecology domain and consists of 13,086 datapoints with 216 dimensions. The results of the experiments on the natural datasets provide insight into how techniques for dimensionality reduction perform on real-world datasets.

On the artificial datasets, we performed experiments using the linear discriminant classifier. The choice for the linear discriminant classifier is motivated by the knowledge that all class boundaries in the artificial datasets are linear (in the high-dimensional data representation). On the natural datasets, we performed experiments with five classifiers: (1) the 1-nearest neighbor classifier, (2) the linear discriminant classifier, (3) the quadratic discriminant classifier, (4) the naive Bayes classifier, and (5) the Least Squares Support Vector Machine (LS-SVM). By employing five different classifiers, we ensure that the selected classifier does not influence the results. The results of all experiments were obtained using 10-fold cross-validation. The parameter settings we used in the experiments were determined by hold-out testing in preliminary experiments.

In the experiments, we do not consider out-of-sample extensions of the techniques. We performed experiments without out-of-sample extension, because the quality of the dimensionality reduction techniques themselves is our main interest. Furthermore, out-of-sample extension can readily be performed by adding the new samples to the existing data, performing the dimensionality reduction, and classifying the test samples in the low-dimensional space.

		Linear		Nonlinear								
Dataset (d)	*None*	*PCA*	*LDA*	*Isomap*	*KPCA*	*DM*	*Autoenc.*	*LLE*	*LEM*	*HLLE*	*LTSA*	*LLC*
Swiss roll (2D)	1.57%	55.64%	1.54%	1.86%	85.79%	61.94%	63.35%	10.98%	10.20%	1.17%	**1.13%**	3.42%
Stretched SR (2D)	1.53%	61.74%	**1.52%**	6.31%	82.31%	71.11%	20.22%	8.42%	12.36%	6.44%	6.42%	61.16%
Broken SR (2D)	1.62%	53.40%	1.61%	2.34%	84.01%	54.19%	56.44%	11.75%	11.84%	**1.40%**	1.41%	62.80%
Helix (2D)	4.65%	5.18%	6.24%	**3.05%**	24.87%	5.18%	54.32%	21.65%	3.42%	84.09%	3.88%	16.18%
Twinpeaks (2D)	20.21%	20.21%	20.43%	46.03%	33.02%	**7.57%**	21.63%	22.58%	46.03%	20.15%	19.92%	34.52%
Clusters (2D)	0.87%	0.89%	0.93%	0.87%	0.93%	0.89%	19.46%	0.87%	1.22%	1.00%	0.96%	**0.00%**
Intersect (2D)	10.91%	53.54%	**10.91%**	50.54%	90.43%	53.37%	95.08%	21.35%	55.68%	64.91%	66.02%	73.42%
Average	5.91%	35.80%	**6.17%**	15.86%	57.34%	36.32%	47.21%	13.94%	20.11%	25.59%	14.25%	35.93%

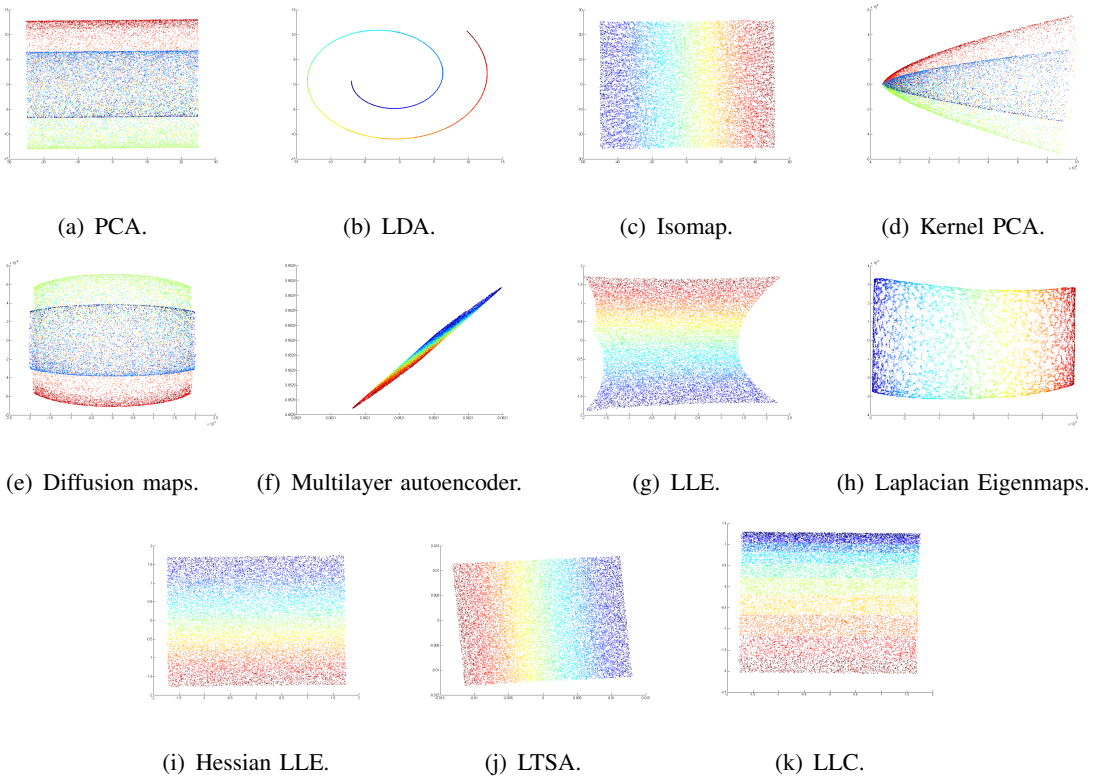
TABLE III

GENERALIZATION ERRORS ON ARTIFICIAL DATA (USING LINEAR DISCRIMINANT CLASSIFIER).

B. Artificial datasets

In Table III, we present the generalization errors of linear discriminant classifiers trained on artificial datasets that were processed using the techniques for dimensionality reduction. The left column indicates the name of the dataset and the dimensionality to which we reduced the 3-dimensional datasets. The best performing technique for each dataset is shown in boldface.

From the results in Table III, we can make five observations. First, we observe there is no single method that outperforms all other methods on the datasets. In the results of the experiments, LDA, Isomap, diffusion maps, Hessian LLE, LTSA, and LLC are all superior on selected datasets. Second, we observe that most nonlinear techniques that employ neighborhood graphs (viz., Isomap, LLE, Laplacian Eigenmaps, Hessian LLE, and LTSA) outperform PCA on the artificial datasets. On the other hand, nonlinear techniques that do not employ neighborhood graphs (viz., diffusion maps, Kernel PCA, and autoencoders) perform poorly on datasets such as the Swiss roll dataset. LLC performs comparable to PCA on the artificial datasets. Third, we observe that most techniques have severe problems when faced with an intersecting dataset. Except for LDA, all techniques mix up the classes in the two-dimensional representation of the intersecting dataset. Fourth, comparison of the results on the stretched Swiss roll dataset with the results on the normal Swiss roll dataset leads to the observation that LLC suffers severely from the changing curvature in the stretched Swiss roll dataset. Isomap, Hessian LLE, and LTSA also suffer somewhat from the changing curvature of the manifold. The results do not show that the presence of a hole in the data manifold has a negative influence on the performance of the



(a) PCA. (b) LDA. (c) Isomap. (d) Kernel PCA.

(e) Diffusion maps. (f) Multilayer autoencoder. (g) LLE. (h) Laplacian Eigenmaps.

(i) Hessian LLE. (j) LTSA. (k) LLC.

Fig. 4. Two-dimensional representations of the Swiss roll dataset.

techniques. Fifth, the results in Table III show that four techniques for dimensionality reduction achieve good performances on all non-intersecting artificial datasets: (1) LDA, (2) LLE, (3) Laplacian Eigenmaps, and (4) LTSA.

Figure 4 presents the two-dimensional representations of the Swiss roll dataset constructed by the dimensionality reduction techniques[10]. From the depicted representations, we derive three observations. First, we observe that techniques that do not construct a neighborhood graph of the data (viz., PCA, LDA, diffusion maps, Kernel PCA, and autoencoders) are not capable of successfully learning the 2-dimensional structure of the Swiss roll manifold. Second, we observe that the low-dimensional representations constructed by LLE and Laplacian Eigenmaps contain global radial distortions, sometimes referred to as folding [11]. These radial distortions cause non-linearities in class boundaries on the manifold that were originally linear. Third, the plots reveal that Laplacian Eigenmaps are capable of learning the global structure of the manifold, but introduce local distortions in the low-dimensional representation of the manifold. The local

[10]Plots of two-dimensional representations of other datasets are shown on `http://www.cs.unimaas.nl/l.vandermaaten`

		Linear		Nonlinear								
Dataset (d)	*None*	*PCA*	*LDA*	*Isomap*	*KPCA*	*DM*	*Autoenc.*	*LLE*	*LEM*	*HLLE*	*LTSA*	*LLC*
MNIST (2D)	69.68%	61.26%	69.43%	59.46%	63.12%	86.92%	67.74%	52.08%	61.11%	77.76%	82.10%	**49.85%**
MNIST (20D)	69.68%	16.30%	79.06%	**11.95%**	16.58%	86.92%	50.82%	13.29%	15.15%	90.00%	74.88%	17.33%
COIL20 (2D)	73.97%	44.22%	**21.79%**	41.50%	48.70%	93.21%	66.14%	57.90%	77.42%	77.40%	71.00%	48.85%
COIL20 (20D)	73.97%	13.27%	**0.33%**	14.63%	14.64%	91.89%	46.54%	20.96%	17.15%	17.86%	22.93%	15.44%
ADA (11D)	27.22%	20.99%	**17.35%**	29.09%	22.32%	32.74%	20.49%	25.18%	23.32%	24.81%	24.19%	34.51%
GINA (195D)	42.93%	42.84%	**6.84%**	44.22%	43.32%	35.61%	49.03%	25.84%	31.39%	49.16%	47.14%	17.91%
HIVA (324D)	11.56%	4.91%	**1.58%**	23.00%	12.02%	3.79%	9.69%	4.70%	3.76%	4.18%	5.42%	4.05%
NOVA (395D)	27.29%	21.25%	**0.01%**	28.45%	22.95%	24.47%	fail	38.00%	27.55%	31.61%	29.61%	34.15%
SYLVA (44D)	21.44%	3.18%	**1.59%**	4.62%	5.77%	6.28%	10.02%	15.30%	9.90%	7.24%	6.15%	11.68%
Average	46.42%	25.36%	**22.00%**	28.55%	27.71%	53.35%	41.16%	28.14%	29.64%	42.23%	40.38%	25.97%

TABLE IV

AVERAGE GENERALIZATION ERRORS ON NATURAL DATASETS (OVER FIVE CLASSIFIERS).

		Linear		Nonlinear								
Dataset (d)	*None*	*PCA*	*LDA*	*Isomap*	*KPCA*	*DM*	*Autoenc.*	*LLE*	*LEM*	*HLLE*	*LTSA*	*LLC*
MNIST (2D)	3.93%	54.92%	52.87%	52.08%	56.95%	84.50%	60.37%	36.84%	44.88%	56.31%	70.96%	**38.92%**
MNIST (20D)	3.93%	4.38%	38.68%	4.99%	**4.45%**	81.06%	41.05%	10.14%	6.40%	89.12%	55.06%	8.66%
COIL20 (2D)	0.00%	29.93%	**0.14%**	23.04%	32.85%	88.33%	49.09%	29.03%	63.40%	65.14%	55.90%	23.06%
COIL20 (20D)	0.00%	**0.00%**	**0.00%**	3.76%	0.42%	84.65%	5.49%	10.49%	10.00%	12.08%	5.90%	0.14%
ADA (11D)	16.81%	17.02%	**15.92%**	24.81%	19.33%	23.46%	17.31%	24.76%	21.41%	24.81%	22.88%	24.13%
GINA (195D)	18.01%	17.54%	**5.99%**	24.48%	19.95%	12.59%	48.59%	13.83%	16.11%	49.16%	40.03%	14.88%
HIVA (324D)	3.51%	3.51%	**1.25%**	3.52%	3.43%	3.51%	4.34%	3.51%	3.41%	3.51%	3.51%	3.49%
NOVA (395D)	24.97%	14.54%	**0.00%**	28.44%	14.03%	10.78%	fail	28.79%	24.46%	28.45%	27.42%	28.39%
SYLVA (44D)	1.46%	1.50%	**1.15%**	3.23%	4.23%	6.15%	5.90%	5.43%	5.84%	6.15%	6.15%	5.73%
Average	8.07%	15.93%	**12.89%**	18.71%	17.29%	43.89%	31.35%	18.09%	21.77%	37.19%	31.98%	16.38%

TABLE V

BEST GENERALIZATION ERRORS ON NATURAL DATASETS (OF FIVE CLASSIFIERS).

distortions are a consequence of the cost function that is minimized in Laplacian Eigenmaps, which aims at minimizing the distance between near datapoints in the low-dimensional representation of the data.

C. Natural datasets

Table IV presents the average generalization errors we measured on the seven real-world datasets. The generalization errors are averaged over five classifiers: (1) the 1-nearest neighbor classifier, (2) the linear discriminant classifier, (3) the quadratic discriminant classifier, (4) the naive Bayes classifier, and (5) the Least Squares-Support Vector Machine. Table V reports the best

generalization error of the five classifiers. In the tables, the left column indicates the name of the dataset and the target dimensionality to which we attempted to transform the high-dimensional data. Note that for Hessian LLE and LTSA, the dimensionality of the actual low-dimensional representation is not higher than 12, since we set $k = 12$. For LDA, the actual dimensionality of the low-dimensional data representation does not exceed the number of classes minus one. The best performing technique for a dataset is shown in boldface.

From the results in Table IV and Table V, we make three observations. First, we observe that the results reveal that the strong performance of nonlinear dimensionality reduction techniques does not generalize from artificial datasets to natural datasets. On average, the linear techniques LDA en PCA outperform nonlinear techniques for dimensionality reduction. LLC is the only nonlinear technique that performs comparable to PCA. Second, the results reveal that Kernel PCA has a relatively good performance on natural datasets, despite its poor performance on artificial datasets. This observation even holds for datasets in which manifolds are intuitively clearly identifiable, such as the COIL20 dataset (because it contains images of rotated objects). Local nonlinear techniques have an inferior performance on the COIL20 dataset, when compared to Kernel PCA. Third, we observe that the use of autoencoders is infeasible for datasets with a high dimensionality D (such as the NOVA dataset). In our experiments, autoencoders failed to reduce the dimensionality of the NOVA dataset due to a lack of memory (on a server with 32GB RAM).

VII. DISCUSSION

In the previous section, we observed that nonlinear techniques do not outperform linear techniques for dimensionality reduction on natural datasets, despite their ability to learn the structure of complex nonlinear manifolds. In this section, we discuss various weaknesses of nonlinear techniques that explain our experimental results. Our results reveal that nonlinear techniques for dimensionality reduction that do not employ neighborhood graphs (e.g., Kernel PCA, diffusion maps, and autoencoders) do not outperform linear techniques on both artificial and natural datasets. Most likely, this is due to the following four weaknesses of these techniques. First, kernel-based methods such as Kernel PCA require the selection of a proper kernel function, which is a nontrivial problem. In general, model selection in kernel methods is performed using some form of hold-out testing [26], leading to high computational costs. Alternative approaches

to model selection for kernel methods are based on, e.g., maximizing the between-class margin or the data variance using semidefinite programming [46], [79]. Second, depending on the selection of parameters, global techniques for dimensionality reduction may suffer from similar weaknesses as local techniques (e.g., when a Gaussian kernel with a small value of σ is employed). Third, techniques with nonconvex cost functions such as autoencoders suffer from slow convergence and from getting stuck in local minima. Fourth, in [79], it is claimed that the use of the Gaussian kernel in dimensionality reduction generally leads to a poor performance. This claim is supported by our experimental results with diffusion maps.

Nonlinear techniques for dimensionality reduction based on neighborhood graphs do not suffer from the weaknesses discussed above, but were shown to perform poorly on natural datasets. Most likely, the poor performance on natural datasets is due to the following four weaknesses. First, there exist theoretical arguments that techniques based on neighborhood graphs suffer from the curse of dimensionality [7] at the dimensionality of the embedded manifold (i.e., the intrinsic dimension of the data). For high intrinsic dimensionalities, the number of datapoints that is necessary to allow a neighborhood graph to correctly characterize the data grows exponentially. Obviously, for artificial datasets with low intrinsic dimensionality, this weakness does not apply. However, in most real-world tasks, the intrinsic dimensionality of the data is much higher. Second, local properties of a manifold do not necessarily follow the global structure of the manifold (as noted in e.g., [10], [58]) in the presence of noise around the manifold. In other words, local techniques suffer from overfitting on the manifold. This drawback can be overcome by the construction of a number of linear models, and subsequently aligning these linear models (as in LLC). In such an approach, the global geometry of the manifold in the low-dimensional data representation is not distorted by overfitting on the manifold. A drawback of the approach in LLC is that the construction of a mixture of factor analyzers is performed using an EM algorithm. EM algorithms suffer from local minima and are very sensitive to outliers [16]. Furthermore, global alignment of linear models requires careful optimization of the number of linear models that is used (in addition to the optimization of the parameters of the linear models). Third, local methods for dimensionality reduction suffer from folding [11]. Folding is caused by a value of k that is too high with respect to the sampling density of (parts of) the manifold. Folding causes the local linearity assumption to be violated, leading to radial or other distortions. In real-world datasets, folding is likely to occur because the sampling density of the data is not equal for

the entire manifold (e.g., because the prior is not uniformly distributed over the manifold). An approach that might resolve this problem is adaptive neighborhood selection. Techniques for adaptive neighbor selection are presented in, e.g., [51], [61], [77]. Fourth, a problem of local techniques for dimensionality reduction is their sensitivity to outliers [13]. In local techniques for dimensionality reduction, outliers are connected to their k nearest neighbors, even when they are very distant. As a consequence, outliers degrade the performance of local techniques for dimensionality reduction. A possible approach to resolve this problem is the usage of an ϵ-neighborhood. In an ϵ-neighborhood, datapoints are connected to all datapoints that lie within a sphere with radius ϵ. Another approach to overcome the problem of outliers is preprocessing the data in order to remove outliers from the data [55], [82].

The empirical results in the paper agree with those of studies reported in the literature. On selected datasets, nonlinear techniques for dimensionality reduction outperform linear techniques [54], [71], but on many other natural datasets, nonlinear techniques for dimensionality reduction are reported to perform poorly [27], [38], [39], [49]. Future work should aim on improving the results of nonlinear techniques. Most importantly, the susceptibility to the curse of dimensionality and the sensitivity to noise of techniques for dimensionality reduction based on neighborhood graphs have to be overcome. These problems might be addressed by the design of new techniques that share information about the global geometry of the data manifold between a number of models [7]. The relatively good performance of LLC in our experiments with natural datasets support this claim. Future work should address problems with local minima in the construction of mixtures of linear models in LLC.

VIII. Conclusions

The paper presented a review and comparative study of techniques for dimensionality reduction. From the results obtained, we may conclude that nonlinear techniques for dimensionality reduction are not yet capable of outperforming linear techniques for dimensionality reduction, despite their increased modelling power. In the future, we foresee the development of nonlinear techniques that represent the geometry of the data manifold in a number of linear models. Techniques based on global alignment of linear models such as LLC form a good step towards this aim, as the results of our experiments on natural datasets suggest.

REFERENCES

[1] D.K. Agrafiotis. Stochastic proximity embedding. *Journal of Computational Chemistry*, 24(10):1215–1221, 2003.

[2] M. Balasubramanian and E.L. Schwartz. The Isomap algorithm and topological stability. *Science*, 295(5552):7, 2002.

[3] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

[4] M. Belkin and P. Niyogi. Laplacian Eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, volume 14, pages 585–591, Cambridge, MA, USA, 2002. The MIT Press.

[5] A.J. Bell and T.J. Sejnowski. An information maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[6] Y. Bengio, O. Delalleau, N. Le Roux, J.-F. Paiement P. Vincent, and M. Ouimet. Learning eigenfunctions links spectral embedding and Kernel PCA. *Neural Computation*, 16(10):2197–2219, 2004.

[7] Y. Bengio and M. Monperrus. Non-local manifold tangent learning. In *Advances in Neural Information Processing Systems*, volume 17, pages 129–136, Cambridge, MA, USA, 2004. The MIT Press.

[8] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems*, volume 16, Cambridge, MA, USA, 2004. The MIT Press.

[9] C. Bishop, M. Svensen, and C. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[10] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems*, volume 15, pages 985–992, Cambridge, MA, USA, 2002. The MIT Press.

[11] M. Brand. From subspaces to submanifolds. In *Proc. of the* 15^{th}*British Machine Vision Conference*, London, UK, 2004.

[12] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-resolution through neighbor embedding. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 275–282, 2004.

[13] K.-Y. Chang and J. Ghosh. Principal curves for nonlinear feature extraction and classification. In *Applications of Artificial Neural Networks in Image Processing III*, pages 120–129, Bellingham, WA, USA, 1998. SPIE.

[14] H. Choi and S. Choi. Robust kernel Isomap. *Pattern Recognition*, 40(3):853–862, 2007.

[15] T. Cox and M. Cox. *Multidimensional scaling*. Chapman & Hall, London, UK, 1994.

[16] D. de Ridder and V. Franc. Robust manifold learning. Technical Report CTU-CMP-2003-08, Department of Cybernetics, Czech Technical University, Prague, Czech Republic, April 2003.

[17] D. de Ridder, O. Kouropteva, O. Okun, M. Pietikäinen, and R.P.W. Duin. Supervised locally linear embedding. In *Lecture Notes in Computer Science*, volume 2714, pages 333–341, Berlin, Germany, 2003. Springer Verlag.

[18] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems*, volume 15, pages 721–728, Cambridge, MA, USA, 2003. The MIT Press.

[19] D. DeMers and G. Cottrell. Non-linear dimensionality reduction. In *Advances in Neural Information Processing Systems*, volume 5, pages 580–587, San Mateo, CA, USA, 1993. Morgan Kaufmann.

[20] D.L. Donoho and C. Grimes. Hessian eigenmaps: New locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 102(21):7426–7431, 2005.

[21] R. Duraiswami and V.C. Raykar. The manifolds of spatial hearing. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 285–288, 2005.

[22] C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proc. of the 1995 ACM International Conference on Management of Data*, pages 163–174, 1995.

[23] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[24] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, San Diego, CA, USA, 1990.

[25] Z. Ghahramani and G.E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, Department of Computer Science, University of Toronto, 1996.

[26] G. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215–224, 1979.

[27] A.B.A. Graf and F.A. Wichmann. Gender classification of human faces. In *Biologically Motivated Computer Vision 2002, LNCS 2525*, pages 491–501, 2002.

[28] R. Haeb-Umbach and H. Ney. Linear discriminant analysis for improved large vocabulary continuous speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, 1992*, volume 1, pages 13–16, 1992.

[29] J. Ham, D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. Technical Report TR-110, Max Planck Institute for Biological Cybernetics, Germany, 2003.

[30] X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems*, volume 16, page 37, Cambridge, MA, USA, 2004. The MIT Press.

[31] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[32] G.E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[33] G.E. Hinton and S.T. Roweis. Stochastic Neighbor Embedding. In *Advances in Neural Information Processing Systems*, volume 15, pages 833–840, Cambridge, MA, USA, 2002. The MIT Press.

[34] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[35] H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007.

[36] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[37] R. Huber, H. Ramoser, K. Mayer, H. Penz, and M. Rubik. Classification of coins using an eigenspace approach. *Pattern Recognition Letters*, 26(1):61–75, 2005.

[38] N.P. Hughes and L. Tarassenko. Novel signal shape descriptors through wavelet transforms and dimensionality reduction. In *Wavelet Applications in Signal and Image Processing X*, pages 763–773, 2003.

[39] O.C. Jenkins and M.J. Mataric. Deriving action and behavior primitives from human motion data. In *International Conference on Intelligent Robots and Systems*, volume 3, pages 2551–2556, 2002.

[40] L.O. Jimenez and D.A. Landgrebe. Supervised classification in high-dimensional space: geometrical,statistical, and asymptotical properties of multivariate data. *IEEE Transactions on Systems, Man and Cybernetics*, 28(1):39–54, 1997.

[41] D.R. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. In *Proceedings of the* 44^{th} *Annual ACM Symposium on Theory of Computing*, pages 741–750, New York, NY, USA, 2002. ACM Press.

[42] T. Kohonen. *Self-organization and associative memory*. Springer-Verlag, Berlin, Germany, 1988.

[43] J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–27, 1964.

[44] S.Y. Kung, K.I. Diamantaras, and J.S. Taur. Adaptive Principal component EXtraction (APEX) and applications. *IEEE Transactions on Signal Processing*, 42(5):1202–1217, 1994.

[45] S. Lafon and A.B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1393–1403, 2006.

[46] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, and L.E. Ghaouiand M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[47] J.A. Lee and M. Verleysen. Nonlinear dimensionality reduction of data manifolds with essential loops. *Neurocomputing*, 67:29–53, 2005.

[48] H. Li, L. Teng, W. Chen, and I.-F. Shen. Supervised learning on local tangent space. In *Lecture Notes on Computer Science*, volume 3496, pages 546–551, Berlin, Germany, 2005. Springer Verlag.

[49] I.S. Lim, P.H. Ciechomski, S. Sarni, and D. Thalmann. Planar arrangement of high-dimensional biomedical data sets by Isomap coordinates. In *Proc. of the* 16^{th} *IEEE Symposium on Computer-Based Medical Systems*, pages 50–55, 2003.

[50] A. Lima, H. Zen, Y. Nankaku, C. Miyajima, K. Tokuda, and T. Kitamura. On the use of Kernel PCA for feature extraction in speech recognition. *IEICE Transactions on Information Systems*, E87-D(12):2802–2811, 2004.

[51] N. Mekuz and J.K. Tsotsos. Parameterless Isomap with adaptive neighborhood selection. In *Proceedings of the* 28^{th} *DAGM Symposium*, pages 364–373, Berlin, Germany, 2006. Springer.

[52] S. Mika, B. Schölkopf, A.J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de-noising in feature spaces. In *Advances in Neural Information Processing Systems*, volume 11, Cambridge, MA, USA, 1999. The MIT Press.

[53] B. Nadler, S. Lafon, R.R. Coifman, and I.G. Kevrekidis. Diffusion maps, spectral clustering and the reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21:113–127, 2006.

[54] M. Niskanen and O. Silvén. Comparison of dimensionality reduction methods for wood surface inspection. In *Proceedings of the* 6^{th} *International Conference on Quality Control by Artificial Vision*, pages 178–188, Gatlinburg, TN, USA, 2003.

[55] J.-H. Park, Z. Zhang, H. Zha, and R. Kasturi. Local smoothing for manifold learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 452–459, 2004.

[56] M. Partridge and R. Calvo. Fast dimensionality reduction and Simple PCA. *Intelligent Data Analysis*, 2(3):292–298, 1997.

[57] M.L. Raymer, W.F. Punch, E.D. Goodman, L.A. Kuhn, and A.K. Jain. Dimensionality reduction using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4:164–171, 2000.

[58] S.T. Roweis, L. Saul, and G. Hinton. Global coordination of local linear models. In *Advances in Neural Information Processing Systems*, volume 14, pages 889–896, Cambridge, MA, USA, 2001. The MIT Press.

[59] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by Locally Linear Embedding. *Science*, 290(5500):2323–2326, 2000.

[60] R. Salakhutdinov and G. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *Proceedings of AISTATS*07*, 2007.

[61] O. Samko, A.D. Marshall, and P.L. Rosin. Selection of the optimal parameter value for the Isomap algorithm. *Pattern Recognition Letters*, 27(9):968–979, 2006.

[62] A. Saxena, A. Gupta, and A. Mukerjee. Non-linear dimensionality reduction by locally linear isomaps. *Lecture Notes in Computer Science*, 3316:1038–1043, 2004.

[63] B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[64] F. Sha and L.K. Saul. Analysis and extension of spectral methods for nonlinear dimensionality reduction. In *Proceedings of the* 22^{nd} *International Conference on Machine Learning*, pages 785–792, 2005.

[65] J. Shawe-Taylor and N. Christianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[66] J.A.K. Suykens. Data visualization and dimensionality reduction using kernel maps with a reference point. Technical Report 07-22, ESAT-SISTA, K.U. Leuven, 2007.

[67] G.A. Tagaris, W. Richter, S.G. Kim, G. Pellizzer, P. Andersen, K. Ugurbil, and A.P. Georgopoulos. fmri of mental rotation and memory scanning: a multidimensional scaling analysis of brain activation patterns. *Brain Res.*, 26(2-3):106–12, 1998.

[68] Y.W. Teh and S.T. Roweis. Automatic alignment of hidden representations. In *Advances in Neural Information Processing Systems*, volume 15, pages 841–848, Cambridge, MA, USA, 2002. The MIT Press.

[69] J.B. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in Neural Information Processing Systems*, volume 10, pages 682–688, Cambridge, MA, USA, 1998. The MIT Press.

[70] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[71] L. Teng, H. Li, X. Fu, W. Chen, and I.-F. Shen. Dimension reduction of microarray data based on local tangent space alignment. In *Proceedings of the* 4^{th} *IEEE International Conference on Cognitive Informatics*, pages 154–159, 2005.

[72] M.E. Tipping. Sparse kernel principal component analysis. In *Advances in Neural Information Processing Systems*, volume 13, pages 633–639, Cambridge, MA, USA, 2000. The MIT Press.

[73] K. Torkkola. Linear discriminant analysis in document classification. In *IEEE TextDM 2001*, pages 800–806, 2001.

[74] M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. In *Proceedings of the Computer Vision and Pattern Recognition 1991*, pages 586–591, 1991.

[75] M.S. Venkatarajan and W. Braun. New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physicalchemical properties. *Journal of Molecular Modeling*, 7(12):445–453, 2004.

[76] J. Verbeek. Learning nonlinear image manifolds by global alignment of local linear models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1236–1250, 2006.

[77] J. Wang, Z. Zhang, and H. Zha. Adaptive manifold learning. In *Advances in Neural Information Processing Systems*, volume 17, pages 1473–1480, Cambridge, MA, USA, 2005. The MIT Press.

[78] K.Q. Weinberger, B.D. Packer, and L.K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *Proceedings of the* 10^{th} *International Workshop on AI and Statistics*, 2005.

[79] K.Q. Weinberger, F. Sha, and L.K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the* 21^{st} *International Confernence on Machine Learning*, 2004.

[80] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pages 975–982, Los Alamitos, CA, USA, 1999. IEEE Computer Society Press.

[81] T. Zhang, J. Yang, D. Zhao, and X. Ge. Linear local tangent space alignment and application to face recognition. *Neurocomputing*, 70:1547–1533, 2007.

[82] Z. Zhang and H. Zha. Local linear smoothing for nonlinear manifold learning. Technical Report CSE-03-003, Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA, 2003.

[83] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26(1):313–338, 2004.

[84] A. Zien and J.Q. Candela. Large margin non-linear embedding. In *Proceedings of the* 22^{nd} *International Conference on Machine Learning*, pages 1060–1067, 2005.