Indian Statistical Institute
Kolkata

M.Tech. (Computer Science) Dissertation

Novel Approaches to Neighborhood Selection for Nonlinear Dimensionality Reduction

A dissertation submitted in partial fulfillment of the requirements
for the award of Master of Technology
in
Computer Science

Author:
Shankhadeep Roy
Roll No: CS1302

Supervisor:
Dr. Swagatam Das
ECSU

Contents

Abbreviations 5

1 Introduction **6**

1.1 Motivation 6

1.2 Thesis Outline 7

1.3 Dimensionality Reduction 9

1.4 Manifold Learning 11

1.5 Nonlinear dimensionality reduction techniques 14

1.5.1 Multi-dimensional Scaling 14

1.5.2 Sammon's Mapping 15

1.5.3 Locally Linear Embedding 16

1.5.4 Local Tangent Space Alignment 16

1.5.5 Isomap 18

2 Locally Linear Embedding **21**

2.1 Locally Linear Embedding 21

2.2 LLE using other distance measures 26

2.3 LLE using other neighborhood selection 27

2.4 LLE using a reasonable reconstruction weights 28

3 Proposed works **30**

3.1 Problem Definition 30

3.2 Past Work 31
3.3 True neighbors using similarity 32
3.4 True neighbors using normal distribution 36
3.5 Timing Complexity analysis 38

4 Experimental Evaluation **42**
4.1 Datasets 42
4.2 k-NN 43
4.3 Three Topology Preservation Measures 44
4.3.1 Spearmans rho 45
4.3.2 Konigs Measure (KM) 46
4.3.3 Mean Relative Rank Errors (MRRE) 47
4.4 Experiments 48

5 Conclusion **57**
5.1 Future Work 58

List of Figures

1.1 PCA performed on two dimensional Iris data. 10
1.2 Nonlinear data: Sphere manifold 10
1.3 Twin Peaks 12
1.4 Nonlinear dimensionality reduction: Punctured Sphere 14
1.5 Swiss Hole ISOMAP embedding 20

2.1 LLE algorithm. Source: Nonlinear dimensionality reduction by locally linear embedding. Roweis, Saul (2000) 22
2.2 LLE embedding for S-Curve 26
2.3 How the WLLE changes the neighborhood of the points 27

3.1 Swiss Roll true neighborhood through similarity 33
3.2 Figure to show how similarity of neighborhood is performed 35
3.3 Gausian manifold embedding 39
3.4 Flowchart of the nonlinear dimensionality reduction process 41

4.1 Swiss roll embedding 52
4.2 S-curve embedding 53
4.3 Swiss Hole embedding: (a) LLE embedding, (b) Proposed algo 2+LLE embedding of the same data, (c) Proposed algo 1+LLE embedding of the same data 54
4.4 3D cluster embedding 54
4.6 MNIST dataset embedding 56

List of Tables

3.1 Timing complexities of nonlinear dimensionality reduction techniques 40

4.1 Description of Datasets 43
4.2 Topological measure results for embeddings of Swiss Roll dataset . . 48
4.3 Topological measure results for embeddings of S-Curve dataset . . . 49
4.4 Topological measure results for embeddings of Mobius Strip dataset 49
4.5 Topological measure results for embeddings of Twin Peaks dataset . 50
4.6 Misclassification percentages for classifying the lower dimensional embeddings of the separate datasets 50
4.7 Misclassification percentages for classifying face and character dataset 51

Abbreviations

PCA ..... Principal Component Analysis

MDS ..... Multi-Dimensional Scaling

LTSA ..... Local Tangent Space Alignment

LLE ..... Locally Linear Embedding

MVU ..... Maximum Variance Unfolding

KPCA ..... Kernel Principal Component Analysis

UCI ..... University of California, Irvine

MNIST ..... Mixed National Institute of Standards and Technology

k-NN ..... k-Nearest Neighbour

WLLE ..... Weighted Locally Linear Embedding

RLLE ..... Robust Locally Linear Embedding

RT ..... Relative Transformation

KRT ..... Kernel Relative Transformation

NSE ..... Neighbor Smoothing Embedding

KM ..... Konig's Measure

MRRE ..... Mean Relative Rank Error

Chapter 1

Introduction

This thesis is on improving the nonlinear dimensionality reduction techniques using a better approach for neighborhood selection and thereby improving classification and clustering processes.

1.1 Motivation

Most real datasets have very high dimensions: hence handling them is important as high dimensional data computation is costly. But due to high correlation of the data, it is safe to assume, feature extraction techniques are needed to reduce the dimensions into relevant information. In this thesis, the work is focused on how nonlinear datasets are such that they have high dimension feature space but can be embedded into a lower dimensional data. Embedding consists of *planarizing* a high dimension nonlinear data into linear data.

1.2 Thesis Outline

The remaining chapters are organized as follows. In Chapter 1, dimensionality reduction is discussed in detail along with other nonlinear dimensionality reduction techniques. The various other nonlinear techniques identify how our proposed algorithm is different in approach and performs better.

In Chapter 2, Locally linear embedding is discussed in detail along with a short survey of all modifications done on it and all previous work done related to neighbourhood selection for nonlinear dimensionality reduction.

In Chapter 3, the proposed algorithms are discussed along with their time complexities and how they are an improvement over the former.

In Chapter 4, the experiments are discussed. These testing are done on the results obtained after classifying the low dimensional embedding. The embeddings are done using Locally Linear Embedding while the classification is done by applying k-Nearest Neighbor classification technique.

In this report, initially we see how the LLE works and performs on manifolds as well as real datasets. LLE having no internal model tries a generalized nonlinear dimensionality reduction approach: this method is not intended as depending upon the geometry and spatial organization of the points if the point model is made adaptive, the algorithm will run faster and with better result as it would then always tend to return *true neighbours* not just any neighbours through Euclidean distance measure. This is the motivation for our proposed algorithms, where *true neighbours* are found using the neighbourhood similarity of the points. If two houses are *true neighbours* for each other in a locality, then the two house share a large of neighbours among each other, while the converse is also true. The other proposed algorithm is based on the fact that the neighbours are distributed on the higher dimensional space in Gaussian distribution, so the Euclidean distances of the neighbours from a point follows a Gaussian distribution. This being the fact

then by central limit theorem, only the points from the central mean of threshold distance are *true neighbors* once the distances are fit to a normal distribution. This approach works sufficiently well and better than the original LLE algorithm as the datasets tend to follow a Gaussian distribution to the fact that the Entropy of the distances from the all other points fits a normal distribution.

These proposed algorithms find a better way of finding the neighbourhood of each point, beyond that this neighbourhood is fed to the LLE algorithm which in turn does the nonlinear dimensionality reduction. Topological preservation techniques are the measures of finding if a new topological embedding is better than the original topological embedding, or one embedding is better than the other. Since a lot of algorithms do nonlinear dimensionality reduction techniques, a measure or a ranking system is needed to see which algorithm outperforms the other. The topological preservation techniques are used to measure the embeddings of manifold datasets of Swiss Roll, S-Curve, Mobius strip and few other manifolds. Through this a comparison can be easily made as to which algorithm ranked best. Since the nonlinear dimensionality reduction techniques are mostly used in real datasets and even more so on face and character recognition, linear classification tools are used to classify the results of the various embedding results.

1.3 Dimensionality Reduction

Apart from few simple datasets, most datasets have a huge number of dimensions. These datasets are computationally expensive to work with. But it is seen, that in many cases, the features are highly correlated to one another i.e. one feature is highly dependent on another feature. So these features are redundant and dont provide any new insight/affect the properties of the datasets. Though high dimensional data is hugely important, getting rid of such features and reducing the dimensions becomes a more important job. Lower the dimension of feature space, better the algorithm works to understand the properties of the data. Manifold data is one where data is Euclidean in a close neighborhood but globally is not. Lets consider the Earth, Earth being a manifold. All the cities on earth are datapoints for the manifold. Distance between two cities: Kolkata and Howrah is distance from Kolkata to Howrah through the surface of Earth, more like a geodesic distance. But, Kolkata and Howrah being so close the geodesic distance and the Euclidean distance between the two are the same. Now consider Kolkata and New York, the distance between these two cities through the surface(geodesic distance) and the Euclidean distance(through the center of earth) are quite different. So Earth is a manifold, considering how it shows Euclidean properties in close neighborhood while in large global space it doesnt. Discovering the underlying manifold from a dataset becomes very important to know how to reduce the dimensions of the feature space.

Over the years a bunch of dimensionality reduction techniques have been devised and improved upon. Principal Component analysis or PCA is probably the most important of them all.

In this figure 1.1, the PCA algorithm identifies the principal components for the data. Thus the new dimensions of the datasets becomes the new eigenvectors. The eigenvector along the stretch of the data is the first principal component. To

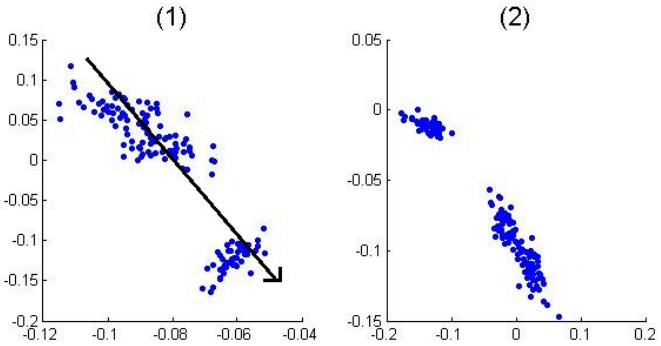


Figure 1.1: PCA performed on two dimensional Iris data.

note, PCA can identify only the linear dimensional reduction. So for nonlinear data like manifolds, the the principal components would be long the equator and through the poles. Take Figure 1.2 for example, this is a nonlinear data, and the PCA doesn't work. Other linear dimensional reduction techniques are Singular Value decomposition, Factor analysis, Linear discriminant analysis.

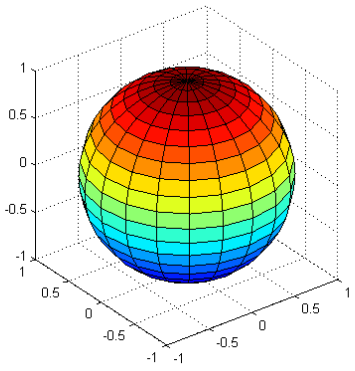


Figure 1.2: Nonlinear data: Sphere manifold

The nonlinear dimensionality reduction techniques can handle such cases. They are discussed In details later. They can be broadly categorized as two types: either they perform mapping from high to low dimensional data or they use high dimensional data to form datapoints using linear combination of their neighbors. Once

that is done, the dimensions reduced, keeping constrain the linear combination, hence the intrinsic manifold properties of the data remains intact.

Real datasets have high dimensions and highly correlated features. In an ideal case, they fall into a manifold and thereby the manifold can be properly embedded without any error. But real dataset dont possess the ideal case, they do tend to fall on a manifold but with a degree of noise. So the embedding them causes few loss of intrinsic properties. This is the curse of dimensionality reduction. If a dataset is dimensionality reduced, it losses intrinsic details. But the dataset becomes much less costly for computation. A very important scenario is when the nearest neighbor doesnt remain the remain the nearest neighbor after embedding, or when nearest neighbor and the furthest neighbor becomes so close that it cannot be detected as measurement noise, then the nonlinear dimensionality reduction techniques fail to work. Nonlinear dimensionality reduction techniques work on the basic assumption that there is a lower dimensional manifold for these data.

1.4 Manifold Learning

Manifold data is one where data is Euclidean in a close neighbourhood but globally is not. Lets consider the Earth, Earth being a manifold. All the cities on earth are datapoints for the manifold. Distance between two cities: Kolkata and Howrah is distance from Kolkata to Howrah through the surface of Earth, more like a geodesic distance. But, Kolkata and Howrah being so close the geodesic distance and the Euclidean distance between the two are the same. Now consider Kolkata and New York, the distance between these two cities through the surface(geodesic distance) and the Euclidean distance(through the center of earth) are quite different. So Earth is a manifold, considering how it shows Euclidean properties in close neighbourhood while in large global space it doesnt. Discovering the under-

lying manifold from a dataset becomes very important to know how to reduce the dimensions of the feature space.

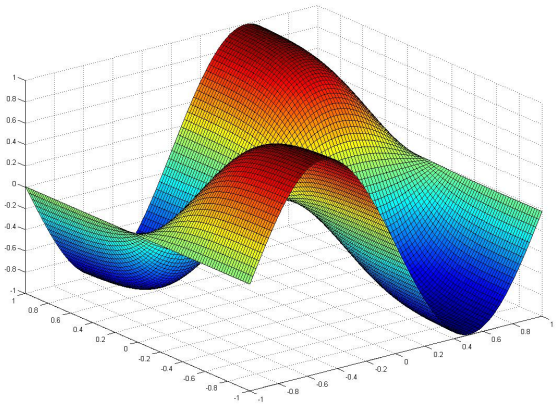


Figure 1.3: Twin Peaks

In the figure 1.3, it is Twin peaks data surface generated through MATLAB commands. Twin Peaks is a nonlinear dataset and the manifold is the surface of the Twin peaks that is drawn. The concept of manifold is very important as it refers to topology and manifold learning being topological preservation techniques.

Manifold learning is an approach to non-linear dimensionality reduction. It refers to understanding the intrinsic topology of the graph and then determining the optimal dimensionality reduction based on the topology. It is in general an unsupervised algorithm as it doesnt consider labels to determine the optimal dimensional reduction.

Most manifold learning techniques are based on the following three steps:

1. Identify local neighbors.
2. Make each point a linear combination of its neighbors.
3. Solve the Eigen decomposition problem of minimizing the global cost function.

The first step and the most important step for the manifold learning is effective choice of neighbors. The thesis focuses on two novel neighbourhood selection techniques. Since most manifold learning techniques uses local neighbourhood information, thus the proposed algorithms can be used with any manifold learning algorithm. The locality identification is the most important step as it identifies the intrinsic properties of the dataset and thereby how the manifold is aligned.

All manifold learning techniques need a parameter of what is the dimension of the reduced dataset. So, if the original dataset is of 10 dimensions which contain a manifold of 6 dimensions, then the parameter value should be 6. Since this is a value to be inputted by the user, it relies on the fact that the user has an understanding of the manifold, which is rarely true. If the same dataset is reduced to 5 dimension, then the data loses major information, it loses the sense of locality and hence the manifold learning becomes a waste.

Let X be the dataset which needs to dimensionality reduced. X is a matrix of $n \times m$ where n represents the number of datapoints and m represents the dimension of the feature space. The dataset is dimensionality reduced to Y, $n \times d$ where $d << m$. That is the data is transformed from a m-dimensional feature space to a d-dimensional feature space.

Dimensionality reduction techniques are of two type mainly:

1. Nonlinear dimensionality reduction techniques

2. Linear dimensionality reduction techniques

This thesis identifies how to better identify neighborhood for manifold. Nonlinear dimensionality reduction techniques is thereby discussed in details.

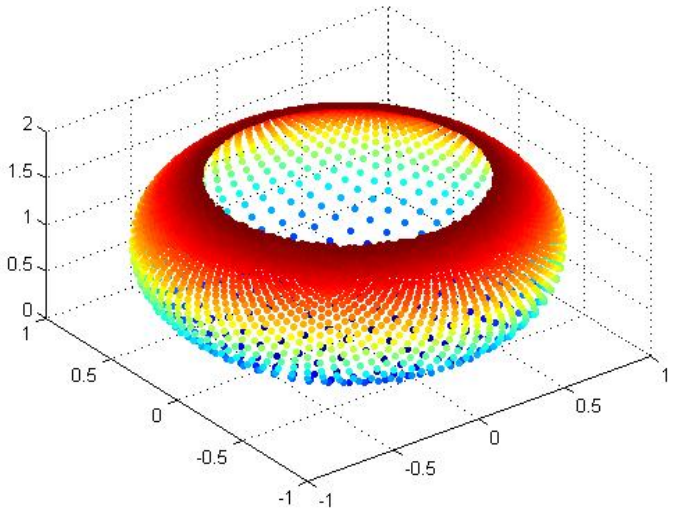


Figure 1.4: Nonlinear dimensionality reduction: Punctured Sphere

1.5 Nonlinear dimensionality reduction techniques

Figure 1.4 shows a nonlinear dataset. These cannot be embedded using linear dimensionality reduction techniques. In this section, the most common nonlinear techniques for dimensionality reduction are explained in details along with their timing complexities.

Some of the non-linear dimensionality reduction techniques are discussed below.

1.5.1 Multi-dimensional Scaling

Multi-dimensional scaling or MDS is an algorithm which causes a simple mapping from high dimensional data to low dimensional data: as it retains the intrinsic properties of the data, it is a nonlinear dimensionality reduction technique. But unlike other algorithms which we will discuss later, it doesnt consider the optimization cost function required for the locality preservation.

The error is calculated by the raw stress function given below:

$$\phi(Y) = \sum_{ij} (\|(x_i - x_j)\| - \|(y_i - y_j)\|)^2 \tag{1.1}$$

Where x_i, x_j are the corresponding high dimensional datapoints and $\|x_i - x_j\|$ is the normal Euclidean distance between these two points. Similarly, y_i, y_j are the corresponding low dimensional datapoints and $\|y_i - y_j\|$is the Euclidean distance between them.

1.5.2 Sammon's Mapping

Sammons mapping, named after John W. Sammon who proposed the algorithm in 1969, is the most primary and simple approach algorithm that maps a high-dimensional space to a space of lower dimensional space while maintaining the intrinsic property of the data, hence by embedding the data. Unlike traditional PCA approach, it involves a cost function optimization problem and not solving a system a linear equations. To note, being a nonlinear dimensionality reduction technique, Sammon's mapping performs better than the traditional PCA.

The cost function for Sammons mapping is given below.

$$\phi(Y) = \frac{1}{\sum_{ij} \|x_i - x_j\|} \sum_{i \neq j} \frac{(\|x_i - x_j\| - \|y_i - y_j\|)^2}{\|x_i - x_j\|} \tag{1.2}$$

Clearly the cost function doesn't take in consideration of the locality condition for the nonlinear data, but considers the entire data, thus the algorithm is not suitable for manifold embedding.

Coming back to the optimization problem in hand, the optimization problem can be easily solved by "*Eigen decomposition of a pairwise dissimilarity matrix*" or many other means possible. Due to the simplicity of Eigen decomposition problem, it is most commonly used in dimensionality reduction. Only drawback

being the algorithm doesn't always converges, so in such cases the algorithm is iterated multiple times and then stopped.

1.5.3 Locally Linear Embedding

Locally Linear Embedding has the basic assumption of the manifold/topological subspace being very well sampled i.e., datapoints are much uniformly spread and vice versa all the corresponding true neighbors lie in a Euclidean neighborhood (i.e. satisfies the manifold property). It can be inferred from the assumption that in an ideal case, the datapoints are possibly some weighted linear combination of its true neighbors. To note, this is an approximation the better the neighborhood set, better is the approximation. From this comes the intrinsic idea for LLE that similar weighted linear combination becomes invariable under simple linear transformations such as translation, rotation, and scaling: Therefore, the intrinsic property of the data should remain undeterred even after the manifold is transformed (while unfolding) into a lower dimensional data. The lower dimensional unfolding and dimensionality reduction of the datapoints is given by solving two constrained least squares optimization problems.

Locally Linear Embedding (LLE) is discussed in details in the later sections.

1.5.4 Local Tangent Space Alignment

Local Tangent Space Alignment (LTSA) identifies locality of the data by considering local tangent space of each datapoint. For each point a tangent space is created and all point are projected on the tangent space and then decided which points are true neighbors of the datapoint. The motivation of the algorithm is that true neighbors share the same tangent space so projected points in turn becomes neighbors. When the data is embedded, the tangent spaces becomes same and uniform thus solidifying the idea. LTSA preserves the intrinsic properties of the

dataset. It considers locality to reduce the dimensions of the dataset, hence by it is a nonlinear dimensionality technique.

The algorithm of LTSA is discussed below.

Algorithm 1: Local Tangent Space Alignment

Result: Embedded dataset

initialization;

1. Selecting neighbourhood for each point;
2. Extracting local coordinates;
3. Aligning local coordinates;

1. **Selecting neighbourhood for each point**

   For each of the datapoints x_i, the k nearest neighbours x_{ij}. The nearest neighbors for each datapoints x_i is identified based on a desired distance metric which in general is Euclidean norm.

2. **Extracting local coordinates**

   Since the locality the datapoints hold the linear property, PCA is performed on each of the datapoints x_i and its k neighbours x_{ik}. PCA identifies the tangent space and projects the neighbors x_{ik} on them. The optimization error function becomes:

$$\sum_{j=1}^{k_i} \left\| x_{ij} - x_i - Q_i \theta_j^{(i)} \right\|^2 \tag{1.3}$$

   where x_{ij} is the coordinate of the j-th neighbour of datapoint x_i
   Q_i is the tangent span of datapoint x_i
   $\theta_j^{(i)}$ is the local coordinate of the datapoints x_i in tangent space
   $\|x_i - y_j\|$ represents the Euclidean norm.

3. **Aligning local coordinates**

   These local tangent space coordinates are used to find the lower dimensional embedding.The optimization error function becomes:

$$\sum_{i=1}^{N} \left\| T_{ij} - c_i - L_i \theta_j^{(i)} \right\|^2 \tag{1.4}$$

   where T is the set of all global co-ordinates and ϕ is the semi-definite matrix represented as

$$\phi = \sum_{i=1}^{N} \frac{1}{k_i} S_i \phi_i S_i^T \tag{1.5}$$

   It can be solved by Eigen decomposition problem.

1.5.5 Isomap

Isometric feature mapping, or Isomap is a combination of the Floyd Warshall algorithm with classical MDS. This method was proposed by Joshua B. Tenenbaum, Vin de Silva, John C. Langford in the year 2000. MDS takes a matrix of pair-wise distances between all points, and computes a position for each point. But MDS only considers the Euclidean distance between the datapoints. As a result, the global geometry of the datapoints may not be captured properly. Isomap uses geodesic distances thereby confirming the global shape of the dataset. Geodesic distance is the distance between two points measured over the surface of the manifold. Finally given all the geodesic distance matrix, it uses Floyd Warshall algorithm to find the neighborhood graph.Isomap then uses classic MDS to compute the reduced-dimensional positions of all the points.

Algorithm 2: ISOMAP

Result: Embedded dataset

initialization;

1. Neighbourhood graph construction;
2. Shortest path computation;
3. Low-dimensional Embedding;

The Isomap, algorithm has three major steps which are discussed below.

1. **Neighborhood graph construction**

   Most nonlinear dimensionality reduction technique uses locality condition and thereby uses any distance metric. In this step, geodesic distance is used to compute distances d_{ij}^x between pairs of points i and j in the high-dimensional space. Geodesic distances compute distances over the manifold. Once the geodesic distances are compute it gives a rough measure of which are neighbors and which aren't. Computing geodesic distances also transforms the dataset into a neighborhood graph, where shortest distance paths are connected by edges representing the weight d_{ij}^x between neighbouring points i and j.

2. **Shortest path computation**

   In this step, the shortest path d_{ij}^g between each pair of point is calculated in the transformed new graph. This can be achieved by using the Floyd-Warshall algorithm of all pair shortest distance in graphs.

3. **Low-dimensional Embedding**

   Isomap is a variant of the classical MDS method. Once the global neighborhood is identified using the geodesic distances, the cost function of the classical MDS can be applied to reduce the dimension of the dataset.

Let Y_i be the new embedding of the point corresponding to X_i in the embedding Y. The cost function:

$$\phi(Y) = \|\tau(D_G) - \tau(D_Y)\|_{L^2} \tag{1.6}$$

where D_Y is the Euclidean distance matrix in the embedded coordinate system and $d_{ij}^y = \|Y_i - Y_j\|$ is the Euclidean norm for Y_i and Y_j.. The τ operator converts distances to inner products. This is later solved by Eigen decomposition problem.

As is clear from figure 1.5, Swiss Hole manifold, embedding cannot be achieved well by ISOMAP, as it cannot make up the gap the point to identify nearest neighbors as it is more susceptible to noise and outliers as the geodesic distance is used.

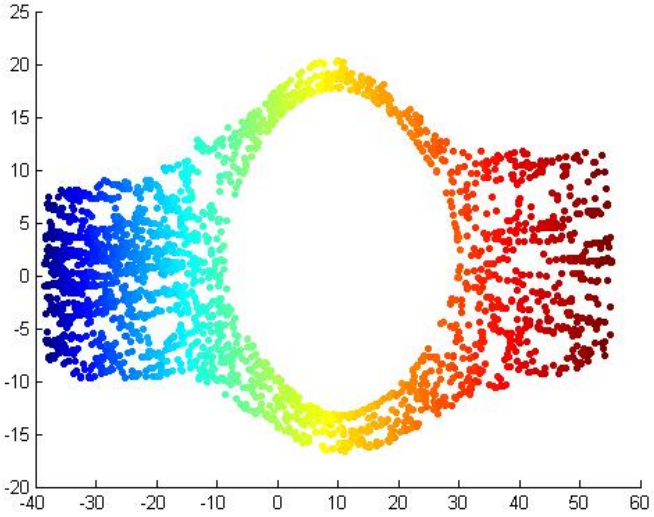


Figure 1.5: Swiss Hole ISOMAP embedding

Chapter 2

Locally Linear Embedding

2.1 Locally Linear Embedding

Locally Linear Embedding has the basic assumption of the manifold/topological subspace being very well sampled i.e., datapoints are much uniformly spread and vice versa all the corresponding *true neighbors* lie in a Euclidean neighborhood (i.e. satisfies the manifold property). It can be inferred from the assumption that in an ideal case, the datapoints are possibly some weighted linear combination of its *true neighbors*. To note, this is an approximation the better the neighborhood set, better is the approximation. From this comes the intrinsic idea for LLE that similar weighted linear combination becomes invariable under simple linear transformations such as translation, rotation, and scaling: Therefore, the intrinsic property of the data should remain undeterred even after the manifold is transformed (while unfolding) into a lower dimensional data. The lower dimensional unfolding and dimensionality reduction of the datapoints is given by solving two constrained least squares optimization problems.

Given a set of datapoints, X where each datapoint, X_i, is represented by each high dimensional features, which in turn serves as the input to the Locally Linear

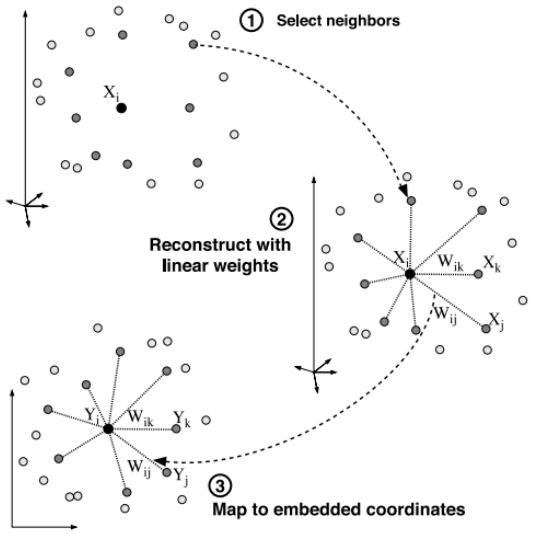


Figure 2.1: LLE algorithm. Source: Nonlinear dimensionality reduction by locally linear embedding. Roweis, Saul (2000)

algorithm consisting of m n-dimensional datapoints/vectors $X_i = (x_{i1}, ..., x_{in}), i = \overline{1, m}, (X_i \epsilon R^n)$,. Thereby X becomes a matrix of size $n \times m$. The target of the LLE algorithm is to reduce the n dimensional features into d dimensions. Thereby the output of the LLE algorithm becomes m d-dimensional datapoints/vectors $Y_i = (y_{i1}, ..., y_{in}), i = \overline{1, m}, (Y_i \epsilon R^d)$, and Y becomes a matrix of size $d \times m$. The LLE algorithm has three simple stages. The first stage consists of identifying all the k nearest neighbors of each datapoint, X_i. Initially all pairwise distances are measured using a distance measure, possibly Euclidean distances, or considering a radius of neighborhood and take all points in the radius as neighbors: based on these distances, the k nearest neighbors are chosen.In the second stage, the weights w_{ij} are computed which can approximate each datapoint, X_i, from a linear combination of the neighborhood, $X_{i1}, ..., X_{ik}$, minimizing the following cost function:

$$E(W) = \sum_{i=1}^{m} \left\| X_i - \sum_{j=1}^{k} w_{ij} X_{ij} \right\|^2, \tag{2.1}$$

given that $\sum_k^{j=1} w_{ij} = 1$. In general, LLE algorithm uses the Euclidean distance to compute the distance, thereby $X_{ij} = (x_{i1}^j, ..., x_{in}^j), i = \overline{1,m}, j = \overline{1,k}, and \, \|X_i - Y_i\|$ is the Euclidean norm. This is constrained least squares optimization problem that can be easily solved by a system of linear equations.

Consider a particular data point X_i with its k nearest neighbors X_{ij} and reconstruction weights $X_{ij}, j = \overline{1,k}$, which sums up to one. Then the reconstruction error can be written as:

$$E^{(i)}(W) = \left\| X_i - \sum_{j=1}^{k} w_{ij} X_{ij} \right\|^2 = \left\| \sum_{j=1}^{k} w_{ij}(X_i - X_{ij}) \right\|^2$$

$$E^{(i)}(W) = \sum_{j,l=1}^{k} w_{ij} w_d c_{jl}^i = \sum_{j=1}^{k} w_{ij} \sum_{l=1}^{k} w_d c_{jl}^i \tag{2.2}$$

Here $C^i = c_{jl}^i, j, l = \overline{1,k}$ is the $k \times k$ local Gram matrix with the elements defined by the following equation:

$$c_{jl}^i = (X_i - X_{ij}) \cdot (X_i - X_{il}) \tag{2.3}$$

where X_{ij} and X_{il} are the neighbors of X_i.

Like previously mentioned, LLE may use any metric distances not just Euclidean to form the distances between points, e.g. instead of traditional distance measure a kernel distance from a kernel feature space can also compute the same distances. This way the neighborhood detection is not unique, and the LLE algo is more generalized.

The following system of linear equations solves the cost optimization problem:

$$\sum_{l=1}^{k} c_{jl}^i w_{il} = 1 \tag{2.4}$$

and then adjusted to hold the constraint $\sum_k^{j=1} w_{ij} = 1$:

$$w_{ij} \leftarrow w_{ij} / \sum_{l=1}^{k} w_{il} \tag{2.5}$$

When the number of neighbors is greater than the original data dimensionality ($k > n$), the weight matrix becomes "*nearly singular*". To solve that problem reconstruction weight equation is modified:

$$c^i_{jl} \leftarrow c^i_{jl} + \delta_{jl} Tr(C^i)t \tag{2.6}$$

where $Tr(C^i)$ is the trace of $C^i, \delta_{jl} = 1$ if $j = l$, and 0, otherwise and t is the control parameter set: $(t > 0, t \ll 1)$.

In the third stage, each datapoint X_i is mapped to a low-dimensional point Y_i, based on the weights w_{ij} so it can finally preserve the same high dimensional neighborhood geometry. So, the new lower dimensional vectors Y_i are determined by minimizing:

$$\Phi(Y) = \sum_{i=1}^{m} \left\| Y_i - \sum_{j=1}^{k} w_{ij} Y_{ij} \right\|^2 \tag{2.7}$$

subject to the following constraints:

$$\frac{1}{m} \sum_{i=1}^{m} Y_i Y_i^T = I, \text{and } \sum_{i=1}^{m} Y_i = 0 \tag{2.8}$$

In the above equation 2.8, I is the identity matrix. Since all the eigenvectors are sorted, the d-dimensional coordinates are computed $(d < n)$ is by taking the bottom $d+1$ eigenvector of $M = (I-W)^T(I-W)$ from the LLE equation, where $W = w_{1j}, ..., w_{mj}, j = \overline{1,k}$.

The LLE algorithm works very well, and thereby has many advantages:

- There are only two parameters to be set.
- Only equation to optimize is minimizing the error.
- Being a topological preservation technique, manifold can be properly embedded.

LLE performs very well on most datasets.The main disadvantages of LLE are as follows:

1. LLE cannot handle non-uniform sample densities efficiently. As various regions differs in sample densities, the weights also drifts accordingly. This cannot be prevented

2. LLE do not have any internal model, so it cannot embed new datapoints.

3. LLE is sensitive to its control parameter i.e. number of neighbours (k) which is a fixed value.

4. LLE is very sensitive to noise. Even a small noise would cause failure in deriving low dimensional coordinates.

5. Eigen decompsosition problem may not always be solvable like when the number of sample is less than the number of neighbors.

6. LLE cannot ensure that two different data points in the high-dimensional space are embedded at different points in a lower-dimensional space.

The original LLE algorithm remains very effective while embedding any manifold structure. But it overlooks the geometry of the manifold and thereby various errors are introduced due to wrong distance considered in the least k distances. So in light of these problems shown by LLE, we wanted to address the issue that LLE has a constant value of k. So the number of nearest neighbors considered should be varied with the geometry of the data. Again, simply using Euclidean distances to compute which points are *true neighbors* are wrong as geodesic distances are more accurate representations of the information.

Our approach holds in heart the basic LLE algorithm but incorporates the geometry of the manifold and thereby making the k more adaptive. This causes the algorithm to take better embedding with lesser errors and standard deviations and also increases the stability of the manifold embedding. Here, stability of the manifold learning is meant by correct embedding of the manifold with greater

variation of the manifold or value of k. As already seen, the value of k is constant and assumed before the start of the LLE. But our approach removes this manual part of the algorithm and supervises k according to how the other *true neighbors* are doing.

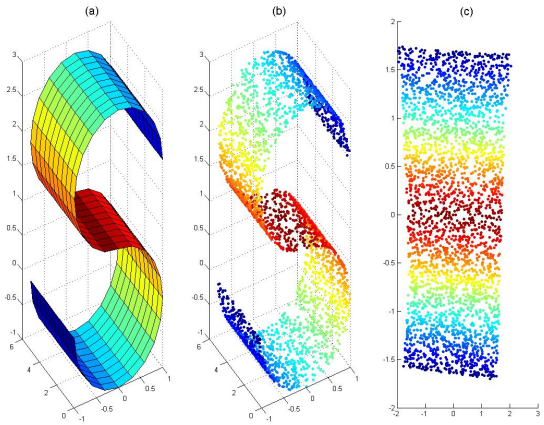


Figure 2.2: LLE embedding for S-Curve

2.2 LLE using other distance measures

Any other distance measures works the same way as these metrics: like the cam weighted distance introduced by Zhou and Chen 2006 (the weighted LLE algorithm, WLLE) which computes the distance by dividing the manifold into patches, and subsequently introducing greater weights to low density areas and high density weights to denser areas.

For the manifold datasets, the WLLE doesn't works well, as forming patches causes erroneous selection of *true neighbors*: while using this technique for feature extraction is very helpful as it shows the important features to be extracted of the lot, as the points are allocated to patches, so definite features can be identified. The Relative Transformation (RT) and Kernel Relative Transformation (KRT are

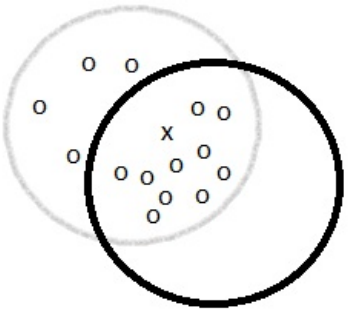


Figure 2.3: How the WLLE changes the neighborhood of the points

proposed by Guihua et al. (2008). It is the kernel approach where a kernel distance is used as metric and the neighborhood is calculated using nearest neighbor in the kernel subspace. Consequently the datapoints very less susceptible to errors caused due to noise and sparsity of data. The kernel method generates then a new neighborhood of the datapoints, which in turn is used by the LLE algorithm.

2.3 LLE using other neighborhood selection

k-Nearest neighbor is used to find the k most near neighbors based on the distance measure. But when the data violates the basic assumption of the LLE algorithm that the data is well-sampled, a new neighborhood selection must be chosen as these common methods don't fruit proper rue neighbors to be selected.

A basic approach involves a PCA for reprocessing the data to remove the erroneous data and then applying LLE on the analyses data. This technique was proposed by Park et al. (2004). Robust Locally Linear Embedding (RLLE) proposed by Chang and Yeung (2006) utilizes the same idea for applying a robust PCA to pre-process the data, giving a probability of how likely a dataset would

fall on the actual manifold, that is how likely a datapoint is not an oultier or noise. Hence the approach is more susceptible to noise and outliers.

It is known that the number of neighbors to be selected, k is constant and user defined, Kouropteva et al. (2002) proposed an algorithm so that the number of neighbors to be selected for the dataset is chosen automatically based on the various topological measures to show which k value optimizes topological measures. A number of experiments are performed on a set of k values to find which suits the dataset best. The topological measures are Spearman's Rho, Konig's measure, Mean Relative Rank Error. The topological measures are well discussed later in Chapter 4.

Once the neighborhood is selected, the linear combination of weights are determined. The neighbor smoothing embedding (NSE) approach nullifies the noise as much as possible by using a local linear surface estimator (Qiu 2004). This estimator smooths the locally linear patches of neighbors. Finally the new reconstruction weight is calculated using the smoothed distances. NSE algorithm performs as well as the RLLE as it can handle noise very well.

2.4 LLE using a reasonable reconstruction weights

The previous section discussed about how the distance metric can be varied, how the neighborhood can be chosen. This section identifies the linear combination of the neighbors. The second stage of the LLE algorithm is that each point is approximated as a linear combination of its neighbors. For non-uniform data, the reconstruction weight becomes very unstable as the neighbors are far away located. So, if the density of the datapoints are considered, the LLE can be made more stable and efficient.

Hou et al (2009) proposed an algorithm to employ linear local transformations

in the sway the reconstruction weights for less denser areas.

$\widehat{G_i} = [z_{i1} - z_{ii}, ..., z_{ik} - z_{ii}] = P_i^T G_i$. Then:

$$Q_i = G_i^T G_i = V_i \Sigma_i^T U_i^T U_i \Sigma_i V_i^T = V_i \Sigma_i^T \Sigma_i V_i^T \tag{2.9}$$

The equation arrives from the singular decomposition of the matrix G_i.

$$U_i = (U_{i1}, U_{i2}); \Sigma_i = \begin{pmatrix} \Sigma_{i1} & 0 \\ 0 & \Sigma_{i2} \end{pmatrix}; V_i = (V_{i1}, V_{i2}) \tag{2.10}$$

where $U_{i1} = (u_{i1}, ...u_{id})$ and $V_{i1} = (v_{i1}, ...v_{id})$ are the first d columns of Ui and V_i, and $U_{i2} = (u_{i(d+1)}, ...u_{iD})$ and $V_{i2} = (v_{i(d+1)}, ...v_{ik})$ are the last $D - d$ and $k - d$ columns of U_i and V_i, respectively, and Σ_{i1} and Σ_{i2} are the dimension of $d \times d$ and $(D - d) \times (k - d)$, respectively.

$$\widehat{Q_i} = \widehat{G_i}^T \widehat{G_i} = V_i \Sigma_i^T \begin{bmatrix} U_{i1}^T \\ U_{i2}^T \end{bmatrix} U_{i1} U_{i1}^T [U_{i1}, U_{i2}] \Sigma_i V_i^T \tag{2.11}$$

$$\widehat{Q_i} = V_i \Sigma_i^T \begin{bmatrix} I_{d \times d} & 0_{d \times (D-d)} \\ 0_{(D-d) \times d} & I_{(D-d) \times (D-d)} \end{bmatrix} \Sigma_i V_i^T = V_{i1} \Sigma_{i1}^T \Sigma_{i1} V_{i1}^T \tag{2.12}$$

Thereby the LLE cost optimization problem becomes:

$$\min \ w_i^T V_{i1} \Sigma_{i1}^2 V_{i1}^T w_i \tag{2.13}$$

$$\text{s.t. } w_i^T \Gamma = 1$$

If $w \, \epsilon \, span(v_{i(d+1)}, ...v_{ik}))$, then $\left\| \Sigma_{i1} V_{i1}^T w_i \right\| = 0$. So the problem can be restated as follows: look for $w_i^T \Gamma = 1$. In other words, we are looking for:

$$\underset{(w \ \epsilon \ span(v_{i(d+1)}, ...v_{ik}), w_i^T \Gamma = 1}{\arg \min} \left\| w_i^2 \right\| \tag{2.14}$$

The solution to the problem is

$$w_i = \frac{V_{i2}^T V_{i2} \Gamma}{\Gamma^T V_{i2}^T V_{i2} \Gamma} \tag{2.15}$$

Chapter 3

Proposed works

3.1 Problem Definition

Most dimensionality reduction techniques under-perform in reducing the dimensions for the real datasets. To improve a nonlinear dimensionality reduction technique, it needs to perform well in neighborhood selection on which points are *true neighbors* of the point.

Definition : *True neighbors*: True neighbors of a point, X_i are the points in the dataset, $X_j \, \epsilon \, X$ that are neighbors of the point in an ideal embedding of the dataset.

Neighborhood selection is a very important part of the dimensionality reduction. So better an algorithm identifies the *true neighbors*, the better would be the embedding. Since neighborhood selection is necessary for all nonlinear dimensionality reduction technique, the proposed algorithm works as a black-box which can be used by other nonlinear dimensionality reduction techniques. The algorithm is discussed with reference to LLE, as it is one of the simpler algorithms and has much less control parameters.

3.2 Past Work

LLE in itself faces a lot of problems as due to lack of parameters, the algorithm cannot be modified for different nonlinear surfaces. On easy way to identify the geometry of the point around a neighborhood, is to compute the Gaussian and mean curvatures of the point.

The value of k is determined dynamically for each point X_i of the manifold. For each point X_i, the corresponding Gaussian curvature ($K_G(i)$) and Mean curvature ($K_H(i)$) are computed. Now $K_G(i)$ and $K_H(i)$ values of each points X_i are considered. If both $K_G(i)$ and $K_H(i)$ of point X_i are within 10% range of $K_G(i)$ and $K_H(i)$ of point X_j , then points X_i and X_j are said to be neighbors and count K_i is increased by 1. This method is applied for all points to determine the dynamic k value of each point. There is a limit to the maximum value of k, (K_{max}). If for some X_i, K_i exceeds K_{max}, then nearest k neighbors are chosen based on Euclidean distance.

This algorithm performs well but this modification works only on LLE. The next two approaches are two novel approaches for neighborhood selection for nonlinear dimensionality reduction techniques. So the approaches not only finds neighbors which can be used for the LLE algorithm but also other nonlinear dimensionality reduction techniques. For simplicity, only the LLE implementation of the neighborhood selection is shown.

Algorithm 3: Neighbors selection using Gaussian and mean curvatures

Result: Newneighborhood

initialization;

Gaussian curvature $K_G(i)$ and Mean curvature $K_H(i)$ for each data point Xi are computed;

while *Two points* x_i *and* x_j *are taken for neighborhood comparison* **do**

  if $0.9 * K_G(i) \leq K_G(j) \leq 1.1 * K_G(i) and 0.9 * K_H(i) \leq K_H(j) 1.1 * K_H(i)$ **then**

    newneighborhood(x_j) should contain x_i;

    newneighborhood(x_i) should contain x_j;

  end

end

Since it takes into consideration the geometry of the manifold, thus it can adapt the value of the k, rather form the d-dimensional vector is greater precision and lesser error as k isn't fixed manually.

3.3 True neighbors using similarity

This algorithm is a direct consequence of what neighbors are *true neighbors* of a point x_i. The Euclidean distances are not the correct reflection of *true neighbors*. If two points "share" more neighbors, then these two points can be called *true neighbors*: as can be seen in a household locality, that if two houses share a lot of neighbors in between them, then those two houses are indeed neighbors to themselves. To prove they will always be neighbors of each other, lets assume an ideal case where in a neighborhood of uniformly distributed, two houses X and Y share m neighbors among themselves. To note, this is not the case for real datasets, real datasets are more sparse, so the neighborhood selection is harder and the following threshold measure might not be applicable.

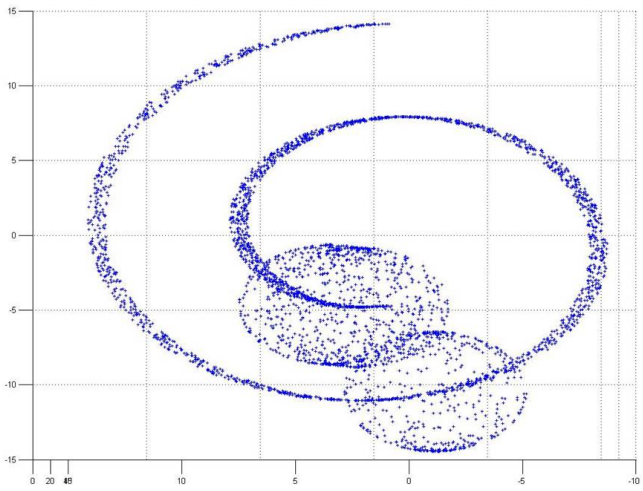


Figure 3.1: Swiss Roll true neighborhood through similarity

As the image shows these two houses, can share at most a portion of the common area.

$$\text{Percentage of neighborhood shared} = \frac{2\left(\frac{\pi R^2}{3} - \frac{\sqrt{3}R^2}{4}\right)}{\pi R^2} = .391 \approx 39\%$$

So if the households share more than these many neighbors, the houses are neighbors to each other. thus if $m \geq 39\%$, then the houses are neighbors to each other. Since for real datasets, it might not be applicable, that's why an experimental value is chosen $\geq 39\%$.

But this is based on the assumption that the dataset is uniformly distributed, but the neighborhood selection process of original LLE causes outliers only in case of non-uniform data. So the threshold is experimentally calculated.

Consider any two datapoints of the dataset, X, say x_i and x_j. The motivation of the algorithm comes the following ideas :

1. If x_i and x_j are sufficiently "*similar*" or are *true neighbors* of each other, then the probability that their neighborhood are highly similar in the higher

dimensional space approaches unity, i.e. more they are similar the more likely it is that x_i and x_j are *true neighbors*.

2. Similarity of points is judged based on the assumption that points in the same neighborhood have the same properties i.e. distance measure actually identifies how varied one datapoint is from another: thereby two datapoints are highly similar/correlated if their neighborhood are highly similar.

Figure 3.2 shows how the similarity of the neighbors are performed on the sorted neighbors list. A similar idea was proposed by Jarvis and Patrick(1973) who used the similarity of the neighbors to cluster the data. The idea has an added advantage that it has "*own built-in automatic scaling*". Therefore, extra parameters aren't required to dilate the radius of the neighborhood search where the number of neighbors are sparse: converse is also true. Though the idea of clustering is not open to similarity to find neighborhood, the following approach identifies the neighborhood. All pairs of points are tested if they are *true neighbors* of each other, through the following approach:

$$|Neighbor(X_i) \cap Neighbor(X_j)| \geq threshold$$

where $k > threshold > .39k$, as threshold cannot exceed the number of neighbors originally considered in Euclidean distance. In figure 3.1, suppose a hybrid Swiss roll, the circle represents radius a globular sphere where the neighbors points are located. Still the points on the Swiss Roll around which the two spheres are located have a lot of shared neighbors, but they aren't *true neighbors* as can be seen clearly from the figure.

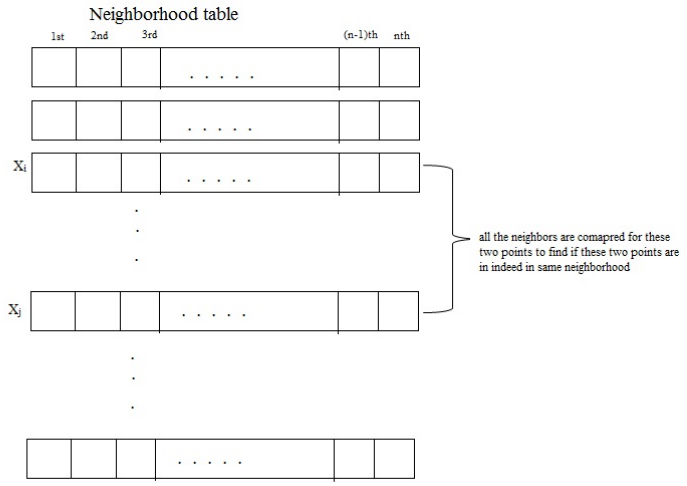


Figure 3.2: Figure to show how similarity of neighborhood is performed

Algorithm 4: True Neighbors using neighborhood similarity

Result: Newneighborhood

initialization;

while *Two points* x_i *and* x_j *are taken for neighborhood comparison* **do**

  Y=$neighbor(x_i) \cap neighbor(x_j)$;

  if $|Y| \geq threshold$ **then**

    newneighborhood(x_j) should contain x_i;

    newneighborhood(x_i) should contain x_j;

  end

end

The initialization step consists of calculating the Neighbor matrix. The neighbor matrix is the ranked matrix of the points with respect to distance from itself. Now for general purpose, the distance measure used is the Euclidean distance. $X = \{x_i : x_i \epsilon \mathbf{R}^m\}$ being the set of all points that participated in the nonlinear dimensionality reduction, where m is the number of features or the dimension of the feature space.

This algorithm finally doesn't take in the ordering of which points are better neighbors to which are not so *true neighbors* of a point. The algorithm doesn't suffers as the LLE algorithm itself doesn't matter if the points are randomly ordered, as it forms a weighted matrix irrespective of the order to make a linear combination of the neighbors. Since linear combination doesn't require ordering, henceby this too doesn't require ordering.

Complexity of this algorithm is of order $\mathrm{O}(k\log(k) * \binom{n}{2})$.All neighborhood pairs need to be considered i.e. each x_i and x_j needs to be compared to see if they are *true neighbors* of each other. Thus the total number of comparison turns out as $k\log(k) * \binom{n}{2}$. Finally, for each pair of points, the k-neighbors of the two points are intersected to find the "similar" neighbors. This is a huge time-overhead on top of the normal LLE algorithm which has a time complexity of $O[D\log(k)N\log(N)]+O[DNk^3]+O[dN^2]$. But, this time can reduced significantly through a small improvement to the the *true neighbor* selection process. Instead of taking each of the $\binom{n}{2}$ pairs to check if they are *true neighbors*, only the k-Euclidean distance neighbors are checked to find if they are true neighbors. Since each of the k neighbors are only checked for the N points, this little modification in the algorithm makes the complexity $\mathrm{O}(nk^2\log(k))$. This is not a huge time overhaul on the original LLE algorithm, and the results are better with more precision.

3.4 True neighbors using normal distribution

Neighborhood pairwise distances indicate that the point are in close neighborhood or non-uniformly placed. Higher a specific distance from x_i and x_j varies the less likely that the points are *true neighbors*. So all the pairwise distances are used to form a distribution so that higher probability can be assured to lower distances while lesser probability to higher distances. Generally this follows a Bell-curve.

Thus there exists a normal distribution to explain the neighborhood distances of a datapoint.

The next proposed algorithm works on the idea that the neighborhood of a point in the Euclidean space follows a normal distribution, that is the neighborhood distances, or the distances of all points to each other around the neighborhood, follows a Gaussian distribution.

Algorithm 5: Neighborhood selection based on probability of fitting the distribution

Result: Newneighborhood

initialization of the neighborhood using a distance measure;

for *Each point* x_i *in the dataset* **do**

Let M be the set of all the k neighbors of x_i;

$D_i = dist(x_j, x_k) : x_j, x_k \epsilon neighborhood(x_i)$;

Fit a normal distribution on D_i;

Find the membership of the radial distances from the point x_i ;

Top k' probability values indicate the most likely neighbors;

end

The original neighborhood distances are put to the normal distribution, thus with a mean, μ and standard deviation, σ they fit the distribution. The neighborhood distances being an always positive distances, the distances forms a Half-normal distribution by itself. Finally, the radial distances are then put to fit on the normal distribution with the same μ, and σ. Fitting the radial distances on the distribution will give a probability of how likely the points are likely to fall close to the central. Thus with higher probability it is likely to fall away from the center while with lower probability it is likely to fall towards the center. The distances that fall closest to the neighbors are taken as the neighbors of the original point x_i. Since following the 3-sigma rule, $k' = 0.68k$, as then the new neighborhood would fall on the first quadrant of the normal distribution. Figure 3.3 shows how

the guassian dataset can be embedded using the above algorithm.

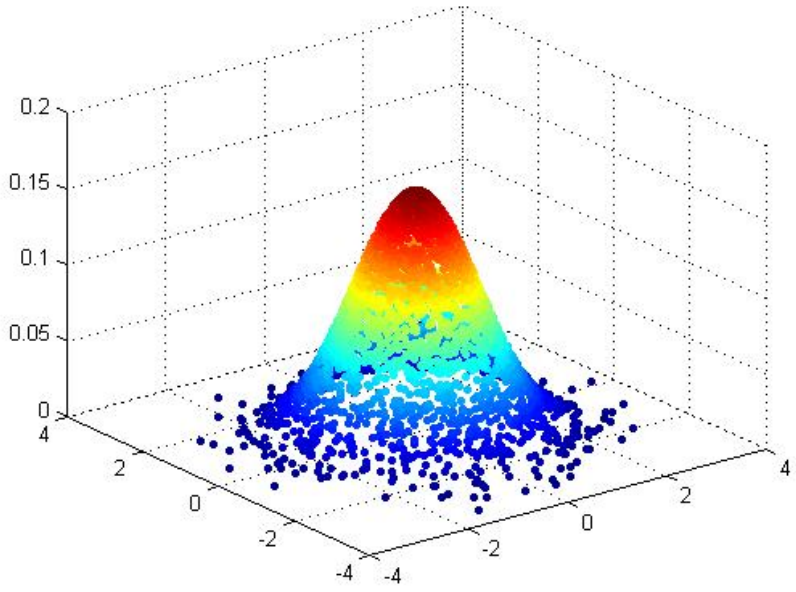
Time complexity of the above proposed algorithm is $O(n\binom{k}{2})$. As for each of the point in the dataset all the $\binom{k}{2}$ combinations are taken as the neighborhood distances so that they can be fit on the normal distribution. Now once the normal distribution is fitted, only the k neighborhood distances are compared to the normal distribution to find their membership. Hence the total time complexity is governed by the initial $O(n\binom{k}{2})$. This neighborhood selection is then the new neighbor for the points fit to the LLE algorithm. Thus the complexity of the total run-time is complexity of this algorithm along with the complexity of the LLE algorithm, thus making the run time of the overall algorithm not affect much on the proposed algorithm as the complexity of the LLE algo is greater.

3.5 Timing Complexity analysis

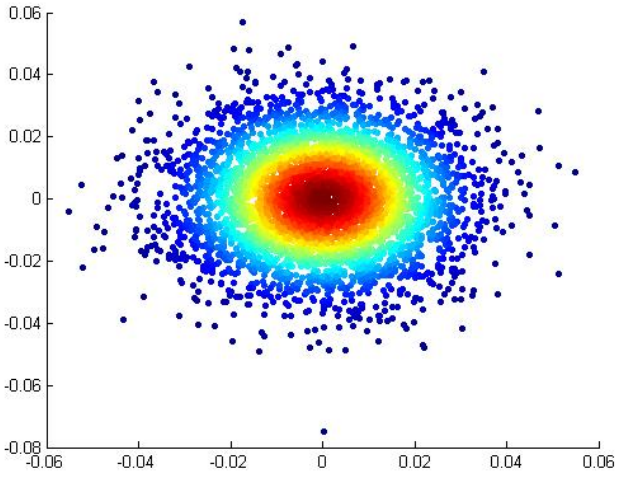
The table 3.1 shows all the timing complexities of the nonlinear dimensionality reduction techniques which can be used with the proposed neighborhood selection process to yield better neighborhood and hence better embedding of the datasets. In this table, mentioned, D is the input higher dimension of the data, k is the number of neighboring points considered, d is the output dimensions, and N is the number of the samples/datapoints.

For Locally Linear Embedding, the initial $O[D\log(k)N\log(N)]$ is for the neighborhood selection process. The initial searching of k-Neighbors based on any distance measure and then sorting the data of dimensions $N \times D$. The proposed algorithms work after this step, so the total cost of K-nearest neighborhood selection turns out to be $O[D\log(k)N\log(N)] + O[Nk^2\log(k)]$.

The final step of any nonlinear dimensionality reduction is a partial Eigen decomposition problem. This step converts the D dimensional input data into d



(a)



(b)

Figure 3.3: (a) Gaussian manifold (b) 2 dimensional Embedding of the dataset done using proposed algorithm 1

Table 3.1: Timing complexities of nonlinear dimensionality reduction techniques

	Complexity
Isomap	$O[D\log(k)N\log(N)] + O[N^2(k+\log(k))] + O[dN^2]$
LLE	$O[D\log(k)N\log(N)] + O[DNk^3] + O[dN^2]$
Modified LLE	$O[D\log(k)N\log(N)] + O[DNk^3] + O[N(k-D)k^2] + O[dN^2]$
Hessian LLE	$O[D\log(k)N\log(N)] + O[DNk^3] + O[Nd^6] + O[dN^2]$
LTSA	$O[D\log(k)N\log(N)] + O[DNk^3] + O[dk^2] + O[dN^2]$
Laplacian Eigenmaps	$O[D\log(k)N\log(N)] + O[DNk^3] + O[dN^2]$

dimensions. The complexity of this step is $O[dN^2]$. Since this step already has a $O[dN^2]$ order, thus $dN^2 >> Nk^2\log(k)$ and thereby $dN >> k^2\log(k)$ condition needs to satisfy to not affect the complexity of the algorithm. Again the second step of most nonlinear dimensionality reduction technique uses a reconstruction matrix of the neighbors which is of the complexity $O[DNk^3]$. This step is clearly the rate determining step in most algorithms. And this step takes far longer time than $O[Nk^2\log(k)]$. Hence the addition of the algorithm to find *true neighbors* doesn't hamper the timing of the nonlinear dimensionality reduction techniques.

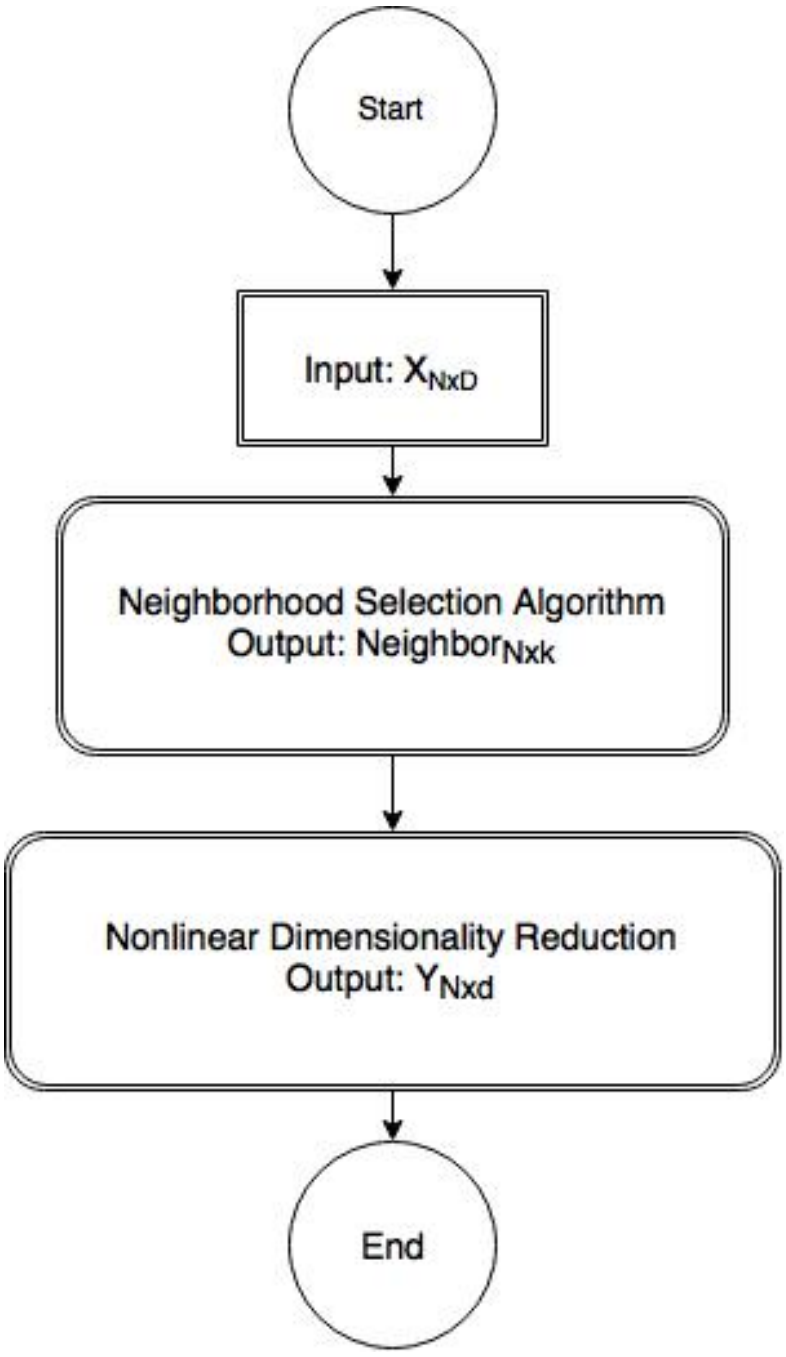


Figure 3.4: Flowchart of the nonlinear dimensionality reduction process

Chapter 4

Experimental Evaluation

In this section, we discuss about the setup of the experiments. The datasets used for the experiments are clearly described below along with the tools used in the experimentation and testing. For the synthetic datasets the topological preservation techniques are used which identify how well an embedding preserves the inherent topology. For the real real datasets, k-NN classifier is used. The real datasets are explained below.

4.1 Datasets

For the experiments, only 11 datasets are used, 9 of them are selected from UCI machine learning repository rest two are MNIST and YALE datasets, they are obtained from Saul Roweis's website.

Here, #sample, #attribute and #class represents the number of number of data samples or examples, number of attributes or features and number of classes for each dataset respectively. Each row represents a dataset.

Table 4.1: Description of Datasets

Datasets	#sample	#attribute	#class
Breast cancer	457	10	2
Diabetes	513	8	2
Haber	205	3	2
Heart	181	13	2
Ion	235	34	2
Iris	150	4	3
Liver	231	5	2
MNIST (number)	2105	784	10
Thyroid	143	5	2
Wine	120	13	2
Yale (face)	165	1024	15

4.2 k-NN

k-NN or k nearest neighbors is a classification technique. For a set of labeled data, k-NN divides them into train set and test set. Train set consists of the subset of the data which is fed to the k-NN algorithm along with the corresponding labels. The k-NN algorithm is then fed the test set without the labels. Each of the datapoint in the test set is then calculated to find which label it belongs to. It is a classification technique as the labels are known and a misclassification rate can be computed based on how many datapoints are labeled different from their original labels. The only parameter for the algorithm is k, hence the algorithm is simple to implement and performs well too.

Algorithm 6: k-NN algorithm

Result: k nearest neighbors

1. All pairwise distances need to be calculated based on a distance metric,like Euclidean distances;

2. The distances are then sorted in increasing order and the top $(k+1)$ distances are chosen;

3. All the points corresponding to respective distances are mapped;

The bottom k points result in the k nearest neighbors;

In the experiments, 7-NN is chosen with a 5-fold cross validation which is iterated 5 times.

4.3 Three Topology Preservation Measures

Given the various nonlinear dimensionality reduction techniques proposed, and the algorithms already there are compared to each other on manifolds.These techniques needs to be compared to see mathematically which outperforms the other, by some embedding measure. Since by nonlinear dimensionality reduction techniques we refer primarily to manifold embedding, any measure or ranking measure should be based on manifold functionality, thus ranking an algorithm better or worse than another should be based on how the ranking measure ranks their respective embeddings of the manifolds. A manifold is is a topological space that resembles Euclidean space near each point. The best way of such ranking measure and also the most useful way to find such a ranking is through topological preservation techniques.The topological preservation techniques can somehow capture the essence of how the topology is maintained when a higher dimensionality manifold is reduced to a lower dimension. Most common topological preservation techniques which are used to measure manifold embeddings, are: Spearman's Rho, Konig's measure(or KM), and Mean Relative Rank Error(or MRRE).

4.3.1 Spearmans rho

Spearman's rho also sometimes referred to as Spearman's rank correlation, ρ or $r_{spearman}$, is a metric to find correlation between two variables. Manifold embedding is supposed to preserve the intrinsic data topology, so it can be used to compare an embedded point and the original point in the higher dimensional space. These two points represent the same data hence the more similar they are the better should be the embedding. The correlation between each X_i and Y_i are recorded to find the total topology preservation between the lower order projection and the higher order original data.

For each point all the neighbors are ranked in order differently for the original data and the embedded data. These ordered ranks are henceforth used to find the Spearman's rank correlation.

$$\rho_{Sp} = 1 - \frac{6\sum_{i=1}^{T}(r_X(i) - r_Y(i))^2}{T^3 - T} \tag{4.1}$$

where $T = m(m-1)/2), r_X(i)$, are the ranks of the neighbors calculated for the original data based on increasing radial distances, and $r_Y(i)$ are the ranks of the neighbors calculated for the embedded data. Henceforth it is clear that

$$-1 \leq \rho_{Sp} \leq 1$$

The best value of the Spearman rho's index being 1 meaning the embedded manifold perfectly represents the original manifold, while -1 being the worst index. The sign corresponds to the direction of correlation. +1 signifies as the ranks for the original data increases the corresponding ranks for the embedded dataset also increases: while -1 signifies as the ranks for the original data increases the corresponding ranks for the embedded dataset also decreases.

The clear advantage of the algorithm is that it has no input parameters and hence simple to use. As is clear from the equation, Spearman's rho is computed for

entire pairwise ranks. But manifold is a subspace which follows Euclidean space properties in its locality but not globally so considering all pairwise distance is not practical as this is turn is a topological preservation technique. So even if ideally the data manages to successfully embed the data, still few local geodesic distanced ranks would differ from the original data ranks. Thereby the Spearman's rho measure would result poor value for the embedding.

4.3.2 Konigs Measure (KM)

To take advantage of the locality idea for the manifold, a new scoring system was proposed by Konig's (2000) to identify how well the locality of the topology is preserved. This is a improvement over Spearman's rho as not the entire dataset is considered but the neighborhood of the data from the original dataset and the embedded dataset are considered. The algorithm works based on two parameters numbers of the nearest neighbors: k_1 and $k_2(k_1 \gg k_2)$. The Euclidean distances estimate a neighborhood here. The score ratings are as follows for the ith and the jth neighbor:

$$KM_{ij} = \begin{cases} 3, \text{if } \overline{r}_X(i,j) = \overline{r}_Y(i,j), \\ 2, \text{if } \overline{r}_X(i,j) = \overline{r}_Y(i,l), l = \overline{1,k_1}, l \neq j \\ 1, \text{if } \overline{r}_X(i,j) = \overline{r}_Y(i,l), l = \overline{k_1+1,k_2}, k_1 < k_2 \\ 0, otherwise \end{cases} \tag{4.2}$$

where $\overline{r}_X(i,j)$ a rank of the jth neighbor X_{ij} of the points X_i where the rank means the order number of X_{ij} in the analyzed data set $X = (x_1, x_2, ..., x_n)$, $\overline{r}_Y(i,j)$ a rank of the jth neighbor Y_{ij} of the points Y_i where the rank means the order number of Y_{ij} in the analyzed data set $Y = (y_1, y_2, ..., y_n)$,

The general measure KM is calculated as follows:

$$KM = \frac{1}{3k_1 \times m} \sum_{i=1}^{m} \sum_{j=1}^{k_1} KM_{ij} \tag{4.3}$$

The range of KM is between 0 and 1, where 0 indicates a poor neighborhood preservation, and 1 indicates a perfect one.

4.3.3 Mean Relative Rank Errors (MRRE)

As an improvement to Spearman's rho, Lee and Verleysen (2007) proposed a topological measure that uses local neighborhood ranks. The ranks $\overline{r}_X(i,j)$ are calculated differently:

- From the original dataset of high dimensions, X, for each of the point X_i as a reference, all the Euclidean norms are computed $\|X_i - X_t\|$, for $1 \leq t \leq m,\ t \neq i$
- the ranks are then obtained by sorting the distances.$\overline{r}_X(i,j)$ be the rank of X_j then X_j is the $\overline{r}_X(i,j)$ index in the sorted distance matrix for X_i. Note that if

$$j = \underset{1 \leq t \leq m, t \neq i}{arg\ min} \|X_i - X_t\| \text{ then } \overline{r}_X(i,j) = 1 \tag{4.4}$$

Since MRRE doesn't consider global ranks, but only neighbors, so $MRRE(X \to Y)$ is different from $MRRE(Y \to X)$.

1. $MRRE(X \to Y) = \frac{1}{C} \sum_{j=1}^{m} \sum_{j \epsilon N_k(X_i))} \frac{\overline{r}_X(i,j) - \overline{r}_Y(i,j)}{\overline{r}_X(i,j)}$
2. $MRRE(Y \to X) = \frac{1}{C} \sum_{j=1}^{m} \sum_{j \epsilon N_k(Y_i))} \frac{\overline{r}_X(i,j) - \overline{r}_Y(i,j)}{\overline{r}_Y(i,j)}$

where $N_k(X_i)$ denotes the set of order numbers of K neighbors of X_i. The normalization factor is given by

$$C = m \sum_{l=1}^{K} \frac{|2l - m - 1|}{l} \tag{4.5}$$

It ranges between 0 and 1. Both measures $MRRE(X \rightarrow Y)$ and $MRRE(Y \rightarrow X)$ vanish if the nearest neighbors of each data point are the same, then the ranks are same and the result causes 0. So the ideal value for $MRRE$ result is 0.

4.4 Experiments

Initially the experiments were carried on the manifold data with the above Topological measures to validate the perfectness of the algorithm over the original LLE algorithm. This way we can judge on perfect manifold data with Gaussian noise to find how the algorithms perform. For the test setup, the three algorithm codes are run over all the manifolds, mainly Swiss roll, S Curve, Mobius Strip, Twin Peaks, Swiss Hole, Torroidal Helix, Occluded disks, 3D clusters with parameter for the topological measure, k=20. To note that experimental results show that k=20 gives almost best results in the topological measures for these manifolds. For the embeddings, the value of nearest neighbors considered are k=25.

Table 4.2: Topological measure results for embeddings of Swiss Roll dataset

	KM	MRRE	Spearman's Rho
Original LLE	0.6367	0.0018	**0.1661**
Proposed Algo 1	0.6065	0.0020	**0.1661**
Proposed Algo 2	**0.6435**	**0.0017**	**0.1661**

As is clear from the topological measures from table 4.2, 4.3, 4.4,4.5, our proposed algorithms does better than the original algorithm both in the MRRE and the KM. By MRRE, the lower is the better, hence in our case our algorithms give lower value in all of the standard manifolds: for KM, the higher the index is, it's better, as is the case for our algorithms. The important thing to note is that, the Spearman's rho didn't get changed for the algorithms: the reason is firstly

Table 4.3: Topological measure results for embeddings of S-Curve dataset

	KM	MRRE	Spearman's Rho
Original LLE	**0.7474**	**0.63784e-04**	**0.1209**
Proposed Algo 1	0.7441	0.69013e-04	**0.1209**
Proposed Algo 2	0.7304	0.69789e-04	**0.1209**

Table 4.4: Topological measure results for embeddings of Mobius Strip dataset

	KM	MRRE	Spearman's Rho
Original LLE	0.6715	9.8036e-04	**0.0184**
Proposed Algo 1	**0.6908**	**9.2215e-04**	**0.0184**
Proposed Algo 2	0.6633	0.001	**0.0184**

Spearman's rho compares all the points and doesn't takes in consideration of the locality for the manifolds, hence most of the points lie almost similarly in the neighborhood map, for the original data and the embedded map. the rest of the neighbors which do fall out of index, can be transformed from one index value to another to subsequent swaps, which causes the changes in the value go null. This being the case, the Spearman's rho values are not considered, as they don't render a opinion which algorithm might perform better that the other.

Like the previous table, in table 4.6, the real datasets are tested. The results show the proposed algorithms do much better than the original algorithm. The proposed algorithms perform in cases as good as the original dataset misclassification rates. That being the case, the multi-class classification data for the datasets aren't much good.

The most application of nonlinear dimensionality reduction techniques has to lie in face, character recognition datasets. MNIST data consist of around 70K data: it wasn't possible to test the dataset for that huge volume, so a random sample of

Table 4.5: Topological measure results for embeddings of Twin Peaks dataset

	KM	MRRE	Spearman's Rho
Original LLE	0.7825	6.4747e-04	**0.0023**
Proposed Algo 1	**0.8097**	**5.2097e-04**	0.0032
Proposed Algo 2	0.78419	6.7728e-04	0.0036

Table 4.6: Misclassification percentages for classifying the lower dimensional embeddings of the separate datasets

	Original Data	LLE Algo	Algo 2	Algo 1
Liver_Disorder	**7.3593**	9.5238	7.7922	**7.3593**
Mamographic	**4.3750**	6.5625	5.6250	**5.1563**
Ionosphere	**2.5532**	4.6809	5.5319	**4.2553**
German_Numeric	7.0465	**6.5967**	7.4963	9.1454
Heart	**6.0773**	8.8398	9.9448	**3.8674**
Diabetes	7.6023	7.7973	8.7719	**7.4074**
Breast_Cancer	**0.6565**	8.0963	9.6280	10.9409

6% uniformly was taken from the each class. the new dataset contained only 4000 samples. For the setup, The images were put through Eigenfaces, after which only the top 250 eigenvalues were taken, hence the MNIST data was previously reduced to 250 features. For the Yale dataset, we also applied Eigenface, and took the top 40% features or the eigenvectors. The classifier used here was k-NN classifier with value of $k = 7$. The classifier is first applied to the original datasets and then to the three embeddings.The results are shown in table 4.7.

The proposed algorithms perform better, on the average as their misclassification rates are much lower than the original data. The observed result agree with

the topological measures results that the proposed algorithms perform better than the original algorithms on an average.

Table 4.7: Misclassification percentages for classifying face and character dataset

	Yale_Faces	Character Datasets(MNIST)
Original Dataset	16.05	9.1972
LLE	18.95	8.8
Proposed Algo 2+LLE	18.85	8.3121
Proposed Algo 1+LLE	**14.6**	**7.8411**
LTSA	**13.97**	8.22
HLLE	19.36	8.65
Laplacian Eigenmaps	20.73	13.21

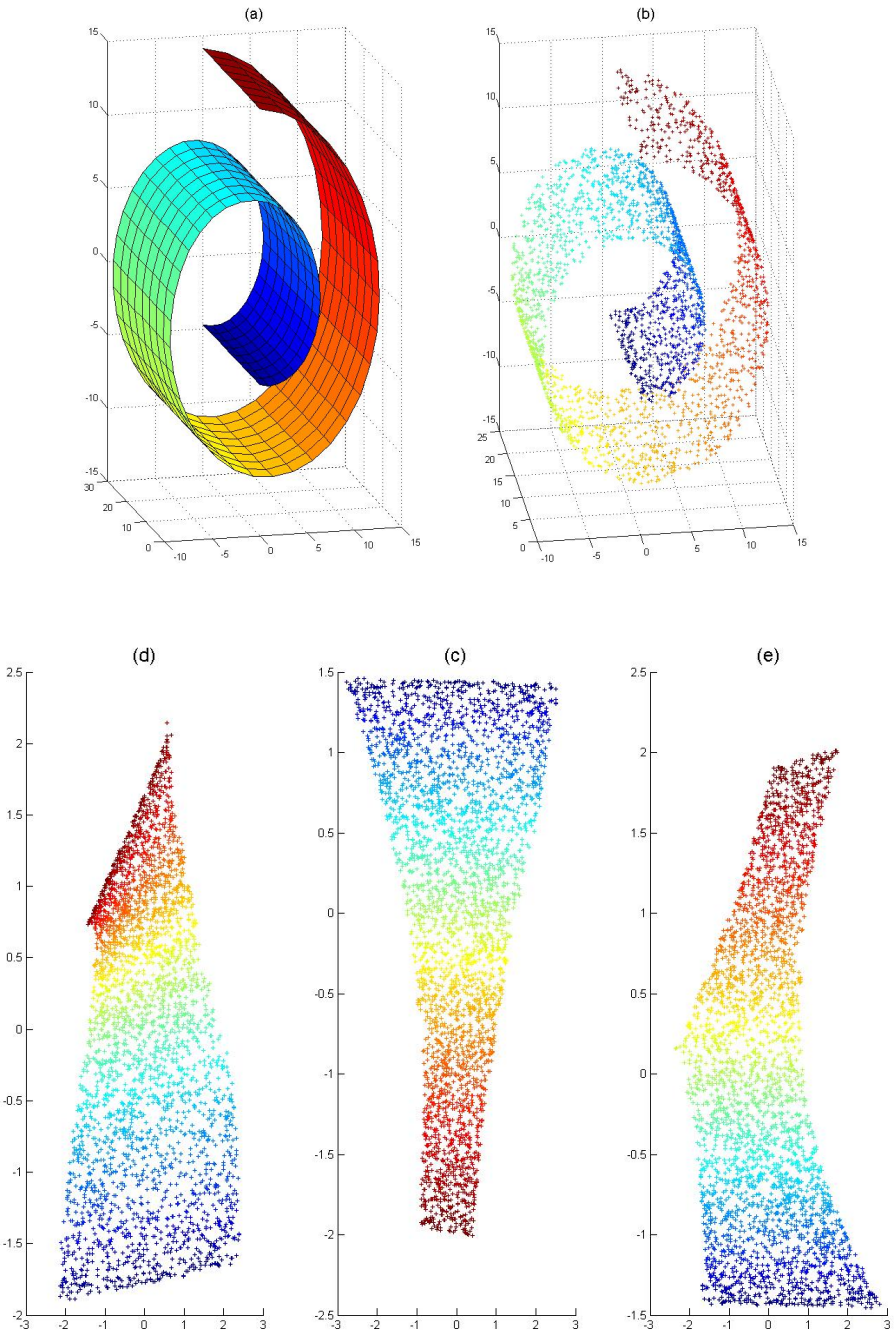


Figure 4.1: Swiss roll embedding: (a) the Swiss roll surface,(b) randomly sampled data points from the Swiss roll, (c) LLE embedding, (d) Proposed algo 2+LLE embedding of the same data, (e) Proposed algo 1+LLE embedding of the same data

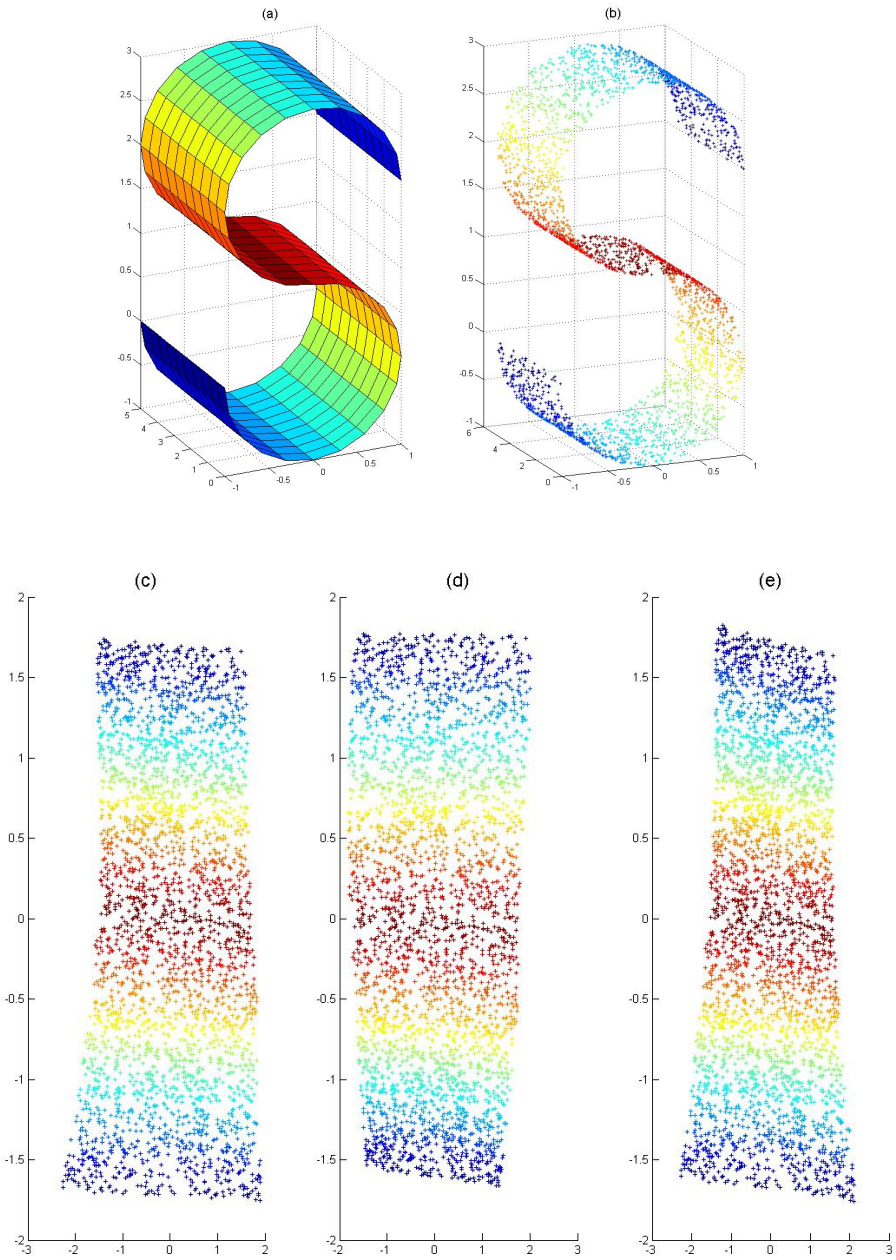


Figure 4.2: S-curve embedding: (a) the Swiss roll surface,(b) randomly sampled data points from the Swiss roll, (c) LLE embedding, (d) Proposed algo 2+LLE embedding of the same data, (e) Proposed algo 1+LLE embedding of the same data

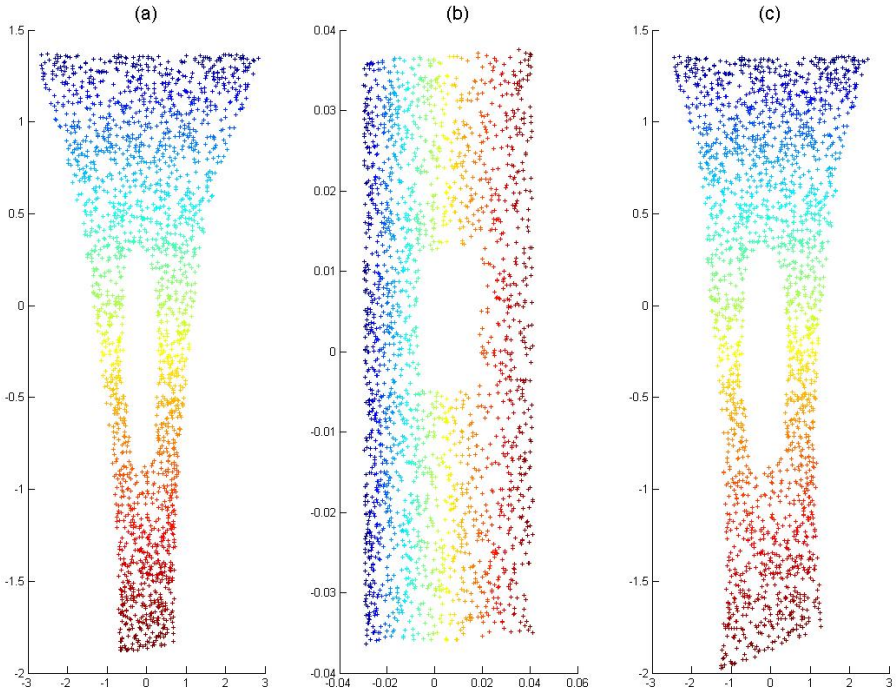


Figure 4.3: Swiss Hole embedding: (a) LLE embedding, (b) Proposed algo 2+LLE embedding of the same data, (c) Proposed algo 1+LLE embedding of the same data

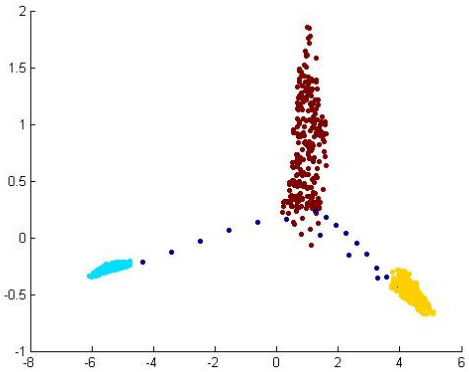
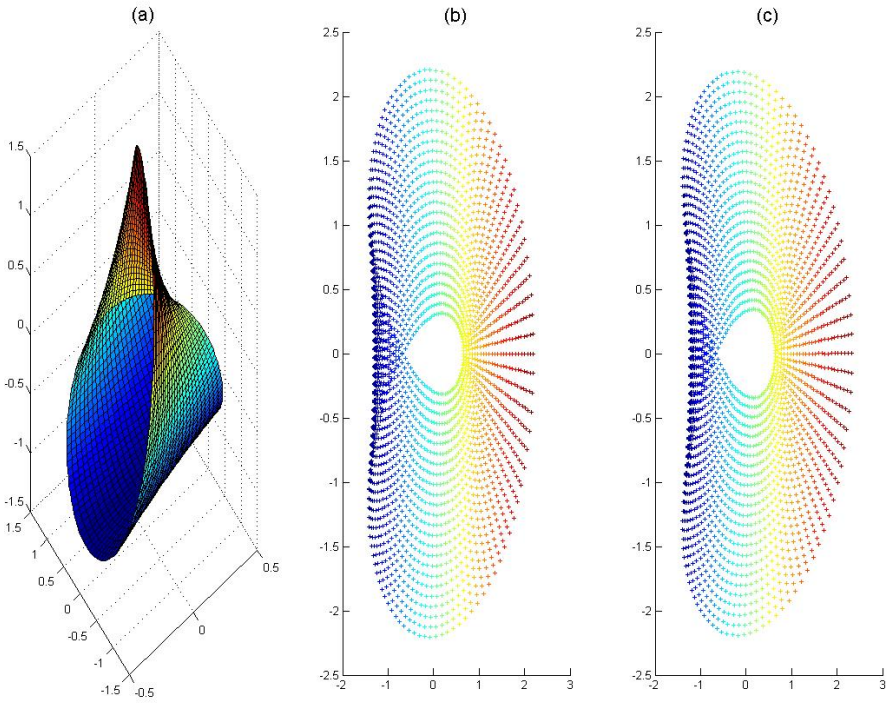
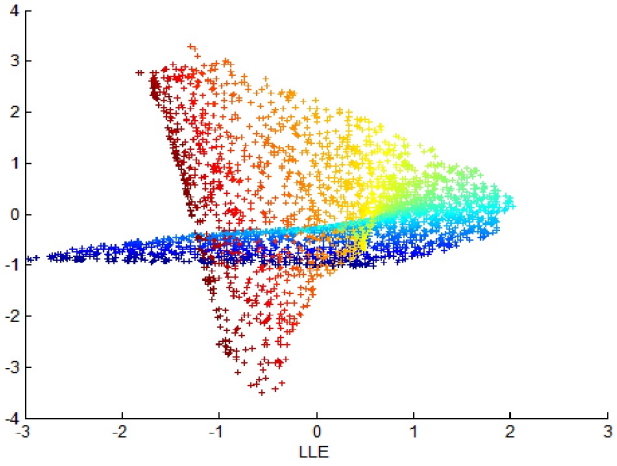


Figure 4.4: 3D cluster embedding

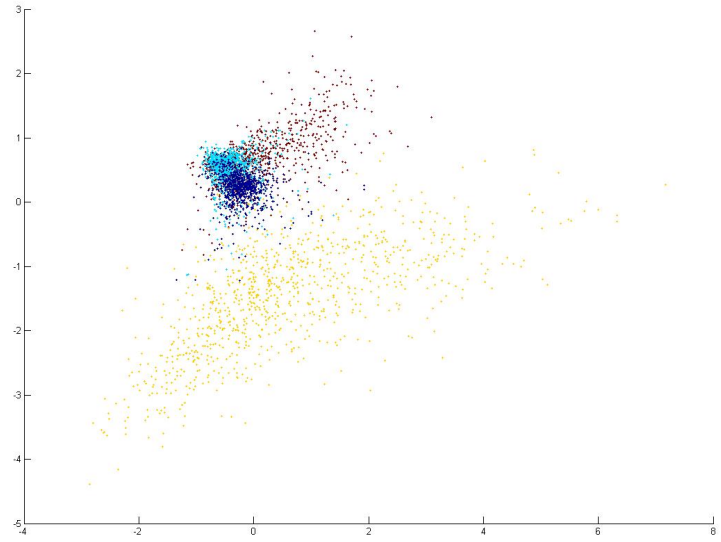


(a) Mobius Strip embedding: (a) Mobius Strip surface,(b)Proposed algo 2+LLE embedding of the same data, (c)Proposed algo 1+LLE embedding of the same data

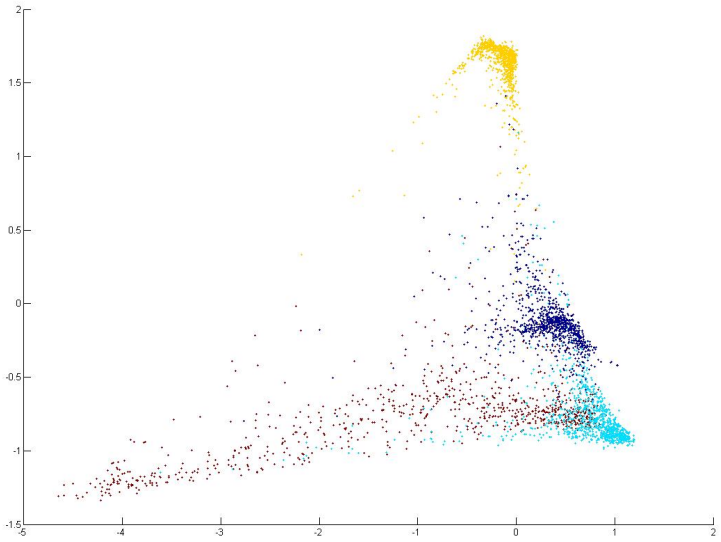


(b) Embedding of swiss Roll is affected by the value of k

Figure 4.5



(a)



(b)

Figure 4.6: MNIST dataset of 3600 samples of class 2,3,4,5:(a) Figure to show embedding of MNIST dataset embedding on 2-dimensions using LLE (b) embedding done using Proposed algo 1

Chapter 5

Conclusion

This thesis has proposed two novel neighborhood selection processes. The key step in nonlinear dimensionality selection is neighborhood selection. The proposed algorithms are added on top of classical LLE as LLE requires a prior neighborhood selection for dimensionality reduction. To test the improvement with the new algorithms, topological preservation techniques are measure which algorithm surpassed the other. The topological measures done on synthetic dataset revealed the proposed algorithm performed better than the original LLE. Spearman's rho resulted same value for each of the algorithms so using Spearman's rho for future validation can deemed unfruitful. For results of torroidal Helix embedding, there is a conflict for win between results of KM and $MRRE$, KM shows Proposed algo 1 as winner while $MRRE$ shows Original LLE as winner. This conflict clearly shows the topological measures aren't foolproof measures for measuring how well an embedding is done for a nonlinear dimensionality reduction. Clearly from the results of the embeddings and the topological measures, Proposed Algo 2 performs almost similar to the Original LLE: in all the experiments, they show close-by results. Though in most cases, it shows an improvement, still it doesn't show a huge improvement over the original LLE. To note, modified versions of the LLE

improves on its timing complexities but none on the neighborhood selection, so once the neighborhood is same for modified versions of the LLE with the original LLE, the topological measures should also show similar results.

For real datasets, the misclassification rate is enough to justify which algorithm performs better: the better algorithm should thereby put lower misclassification rate. In most of the cases, the proposed algo 1 has much lower misclassification rate than the other. But except the Heart dataset, in all the original data misclassification rate is lower, meaning, with loss in dimension the data loses valuable information. In the Heart dataset, the misclassification rate for the Proposed algo 1 is much lower than the other misclassification rates even with original data. So it can be deduced that for the rest of the dataset, though the misclassification rate is lower but it isn't at par with the original data, henceforth losing the importance of dimensionality reduction. For the face dataset and the MNIST, the proposed algorithms perform much better than other nonlinear dimensionality reduction techniques and also original dataset. Face and MNIST datasets are indeed nonlinear dimensional data. Reducing the features of the 28×28 image using PCA is effective as through experimental results, after 250 top eigenvectors, the rest of the vectors don't contribute much to the classification. A clear conclusion can be made that the proposed algorithm performs better than the other nonlinear dimensionality reduction methods that re tested.

5.1 Future Work

As said earlier, the goal of work in this direction is to find a good neighborhood selection method for dimensionality reduction techniques. A good amount of preliminary work for this has been done. Here is a list of future work and challenges :

- Use the algorithm to perform clustering as similar neighborhood selection can be used to find which points belong to which cluster.

- Addressing the issues involved in scaling the model to deal with very huge data. The algorithms used till now work fine on a small dataset. As the dataset size increases, a method like nearest neighbor may no longer be feasible.

- Finding neighborhood using normal distribution also works the same way as finding the entropy in the neighborhood, So using entropy to find an alternate neighborhood selection process. LEGclust or layered entropy graph based clustering is a classic example of entropy based neighborhood selection. This can be used to find neighborhood and later later used for dimensionality reduction or classification purposes.

- To use this algorithm to extend this work of neighborhood selection on other non-linear dimensionality reduction techniques. Some other non-linear dimensionality reduction techniques like Laplacian Eigenmap, Isomap, LTSA and MVU also computes the nearest neighbours in a similar approach to that of LLE. In these techniques, the proposed neighborhood selection methods can be applied and the outcomes can be studied.

- To generalize the value of the number of nearest neighbor parameter i.e. k used in LLE. It is an important parameter as the earlier studies have shown that with the change in value of k, the result varies considerably. So an adaptive approach for selecting the k value dynamically can be tried. In that case, the value of k will be dynamic and will vary for each datapoint based on the local geometry of the dataset.

Bibliography

1. Balasubramanian M, Schwartz EL, Tenenbaum JB, de Silva V, Langford JC (2002) The isomap algorithm and topological stability. Science 295:7

2. Beckmann N, Kriegel HP, Schneider R, Seeger B (1990) The R*-tree: an efficient and robust access method for points and rectangles. In: Proceedings of the 1990 ACM SIGMOD international conference on Management of data, pp 322331 Belhumeur PN, Hespanha JP,KriegmanDJ (1997) Eigenfaces vs. Fisherfaces: recognition using class specific linear. IEEE T Pattern Anal 19:711720

3. Belkin M, Niyogi P (2003) Laplacian eigenmaps for dimensionality reduction and Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput 15(6):13731396

4. Bengio Y, Paiement J, Vincent P, Delalleau O, Roux NL, Ouimet M (2003) Out-of-sample extensions for LLE, Isomap,MDS, Eigenmaps, and Spectral clustering. In: Advances in neural information processing systems, pp 177184

5. Brand M (2003) Charting a manifold. In: Advances in neural information processing systems, vol 15, pp 961968

6. Cai D (2009) Spectral regression: a regression framework for efficient regularized subspace learning, UIUC

7. Cai D, He X, Han J (2007) Spectral regression for efficient regularized subspace learning. ICCV, pp 18

8. Chang H, Yeung D (2006) Robust locally linear embedding. Pattern Recognit 39(6):10531065

9. de La Torre F, Black MJ (2003) A framework for robust subspace learning. Int J Computer Vision 54(13): 117142

10. de Ridder D, Kouropteva O, Okun, Oleg (2003) Supervised locally linear embedding. In: Proceedings of ICANN, pp 333341

11. Donoho D, Grimes C (2003) Hessian eigenmaps: new tools for nonlinear dimensionality reduction. Proc Natl Acad Sci 100:55915596

12. Eftekhari A, Abrishami-Moghaddam H, Babaie-Zadeh M (2009a) k/K-Nearest neighborhood criterion for improvement of locally linear embedding. CAIP, pp 808815

13. Eftekhari A, Babaie-ZadehM, Jutten C, Moghaddam HA (2009b) Robust-SL0 for stable sparse representation in noisy settings. ICASSP, pp 34333436

14. Fienberq S (1985) The analysis of crossclassified categorical data. MIT press, Cambridge

15. Goldberg Y, Ritov Y (2008) LDR-LLE: LLE with low-dimensional neighborhood representation. ISVC, pp 4354

16. GuihuaW, Lijun J, JunW(2008) Kernel relative transformation with applications to enhancing locally linear embedding. IJCNN, pp 34013406

17. Hadid A, Pietikainen M (2003) Efficient locally linear embeddings of imperfect manifolds. MLDM, pp 188201

18. Han PY, Beng Jin AT, Kiong WE (2008) Neighbourhood discriminant locally linear embedding in face recognition. CGIV2008, pp 223228

19. He X, Cai D, Yan S, Zhang H (2005) Neighborhood preserving embedding. IEEE Int Conf Comput Vis2:12081213

20. Holland PW, Welsch RE (1977) Communications in statistics: theory and methods Hou C, Wang J, Wu Y, Yi D (2009a) Local linear transformation embedding. Neurocomputing 72(1012): 23682378

21. Hou C, Zhang C, Wu Y, Jiao Y (2009b) Stable local dimensionality reduction approaches. Pattern Recognition 42(9):20542066

22. Hui K, Wang C (2008) Clustering-based locally linear embedding. ICPR

23. Kadoury S, Levine MD (2007) Face detection in gray scale images using locally linear embeddings. Computing Vis Image Underst 105(1):120

24. Karbauskaite R, Kurasova O, Dzemyda G (2007) Selection of the number of neighbors of each data point for the locally linear embedding algorithm. Inf Technol Control 36(4):359364

25. Kouropteva O, Okun O, inen MPA (2002) Selection of the optimal parameter value for the locally linear embedding algorithm. In: Proceedings of the 1st international conference on fuzzy systems and knowledge discovery, pp 359363

26. Kouropteva O, Okun O, Pietikinen M (2005) Incremental locally linear embedding. Pattern Recognition 38(10):17641767

27. Li B, Zheng C, Huang D (2008) Locally linear discriminant embedding: an efficient method for face recognition. Pattern Recognit 41(12):38133821

28. Li H, Jiang T, Zhang K (2006) Efficient and robust feature extraction by maximum margin criterion. IEEE Trans Neural Netw 17(1):157165

29. Li SZ, Lu J (1999) Face recognition using the nearest feature line method. IEEE Trans Neural Netw 10(4): 439443

30. Lingzhu H, Lingxiang Z,Caiyue C, Min L (2009) Locally linear embedding algorithmwith adaptive neighbors. ISA

31. Pan Y, Ge SS, Al Mamun A (2009) Weighted locally linear embedding for dimension reduction. Pattern Recognitn 42(5):798811

32. Pang YH, Teoh ABJ, Wong EK, Abas FS (2008) Supervised locally linear embedding in face recognition. ISBAST08.

33. Park J, Zhang Z, Zha H, Kasturi R (2004) Local smoothing for manifold learning. CVPR I452-I459

34. Qiu P (2004) The local piecewisely linear kernel smoothing procedure for fitting jump regression surfaces. Technometrics 46:8798

35. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):23232326

36. Saul LK, Rowels ST (2004) Think globally, fit locally: unsupervised learning of low dimensional manifolds. J Mach Learn Res 4(2):119155

37. Tenenbaum JB, De Silva V, Langford JC (2000) A global geometric framework for nonlinear dimension reduction. Science 290(5500):23192323

38. Teng X, Wu B, Yu W, Liu C, (2005) A hand gesture recognition system based on local linear embedding J Visual Lang & Computing Special issue

section on Context and Emotion Aware Visual Interaction - Part I 16(5): 442454

39. Tikhonov AN (1963) Regularization of incorrectly posed problems

40. Valencia-Aguirre J, lvarez-Mesa A, Daza-Santacoloma G, Castellanos-Domnguez G (2009) Automatic choice of the number of nearest neighbors in locally linear embedding CIARP2009, pp 7784

41. Varini C, Degenhard A, Nattkemper TW (2006) ISOLLE: LLE with geodesic distance. Neurocomputing 69(1315):17681771

42. Wang H, Zheng J, Yao Z, Li L (2006) Improved locally linear embedding through new distance computing. ISNN, pp 13261333

43. Wang J (2008) Robust and stable locally linear embedding. FSKD, pp 197201

44. Wang J, Zhang Z (2010) Nonlinear embedding preserving multiple local-linearities. Pattern Recognit 43(4):12571268

45. Wang Y, Wu Y (2010) Complete neighborhood preserving embedding for face recognition. Pattern Recognit 43(3):10081015

46. Watts DJ, Strogatz SH (1998) Collective dynamics of small-world networks. Nature 393(6684):440442

47. Weinberger KQ, Saul LK (2006) Unsupervised learning of image manifolds by semidefinite programming. Int J Comput Vis 70(1):7790

48. Wen G, Jiang L (2006) Clustering-based locally linear embedding. In: Proceedings of 2006 IEEE international conference on systems, man and cybernetics pp 41924196

49. WenG, Jiang L,Wen J, ShadboltNR(2006) Performing locally linear embedding with adaptable neighbourhood size on manifold. J Softw, pp 985989

50. Wu F, Hu ZY (2006) The LLE and a linear mapping. Pattern Recognit 39(9):17991804

51. Xia T, Li J, Zhang Y, Tang S (2008) A more topologically stable locally linear embedding algorithm based on R*-tree. PAKDD, pp 803812

52. Yan Y, Zhang Y (2008) Discriminant projection embedding for face and palmprint recognition. Neurocomputing 71(1618):35343543

53. Yin J, Hu D, Zhou Z (2008) Noisy manifold learning using neighborhood smoothing embedding. Pattern Recognit Lett 29(11):16131620

54. Ying HP, Andrew Teoh BJ, Wong EK (2008) Neighbourhood discriminant embedding in face recognition IEICE Electron Express 5(24):10361041

55. Yulin Z, Jian Z, Sunan W, Xiaohu L (2008) Local linear embedding in dimensionality reduction based on small world principle. CSSE, pp 394398

56. Zeng X, Luo S (2008) Generalized locally linear embedding based on local reconstruction similarity.FSKD2008, pp 305309

57. Zhan D, Zhou Z (2006) Neighbor line-based locally linear embedding. PAKDD, pp 806815

58. Zhang C, Wang J, Zhao N, Zhang D (2004) Reconstruction and analysis of multi-pose face images based on nonlinear dimensionality reduction. Pattern Recognit 37(2):325336

59. Zhang S (2009) Enhanced supervised locally linear embedding. Pattern Recognit Lett 30(13):12081218

60. Zhang Z, Wang J (2007) .MLLE: modified locally linear embedding using multiple weights Adv Neural Inf Process Syst 19:15931600

61. Zhang X, Liu Y, Gao C, Liu J (2008) An efficient algorithm of learning the parametric map of locally linear embedding. IITA, pp 5256

62. Zhang Z, Zha H (2005) Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. SIAM J Sci Comput 26(1):313338

63. Zhang Z, Zhao L (2007) Probability-based locally linear embedding for classification. FSKD, pp 243247

64. Zhao D (2006) Formulating LLE using alignment technique. Pattern Recognit 39:22332235

65. Zhao L, Zhang Z (2009) Supervised locally linear embedding with probability-based distance for classification. Comput Math Appl 57(6):919926

66. Zhao Q, Zhang D, Lu H (2005) Supervised LLE in ICA space for facial expression recognition. ICNNB05, pp 19701975

67. Zhou CY, Chen YQ (2006) Improving nearest neighbor classification with cam weighted distance. Pattern Recognit 39(4):635645

68. Rasa Karbauskait E, Gintautas Dzemyda (2009) Topology Preservation Measures in the Visualization of Manifold-Type Multidimensional Data, INFORMATICA, 2009, Vol. 20, No. 2, 235254

69. R. A. Jarvis And Edward A. Patrick (1973) Clustering Using a Similarity Measure Based on Shared Near Neighbors, Ieee Transactions On Computers, Vol. C-22, No. 11, November 1973, 1025-1034

70. John A. Lee and Michel Verleysen (2008) Rank-based quality assessment of nonlinear dimensionality reduction, ESANN 2008 proceedings, 49-54

71. Konig, A. (2000). Interactive visualization and analysis of hierarchical neural projections for data mining. IEEE Transactions on Neural Networks, 11(3), 615624.