Manifold Learning for Image Processing

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF SCIENCE
IN
ADVANCED COMBINATORICS

by

PANKAJ KUMAR

Moscow Institute of Physics and Technology

Moscow, Russian Federation

June, 2017

Moscow Institute of Physics and Technology

Moscow, Russian Federation

CERTIFICATE

This is to certify that the present work entitled, "**Manifold Learning for Image Processing**" submitted by **Pankaj Kumar** to the Moscow Insitute of Physics and Technology, Russian Federation in partial fulfilment of the requirements for the award of the degree of Master of Science is approved.

.....
Prof. Roman Karasev
Professor
Department of Mathematics
Moscow Institute of Physics and
Technology

Moscow Institute of Physics and Technology

Moscow, Russian Federation

DECLARATION

I hereby declare that the thesis titled, "**Manifold Learning for Image Processing**" being submitted to the Moscow Institute of Physics and Technology, Russian Federation in partial fulfillment of the requirements for the award of the degree of Master of Sceience is a record of bonafide work carried out by me.

The matter embodied in the thesis has not been submitted for the award of any degree in any university or institution.

June, 2017
Moscow, Russian Federation

Pankaj Kumar

Acknowledgments

There are no proper words to convey my deep gratitude and respect for my thesis and research advisor, Prof. Roman Karasev, Moscow Insitute of Physics and Technology, Russian Federation. With his course in discrete geometry, he not only inculcated my interest in manifold learning, but has inspired me to become an independent researcher. He helped me realize the power of critical reasoning, simple, but concrete mathematical representation of idea and guidance to recover when my steps faltered. He taught me how to critically question thoughts and express ideas in simple but scientific way. His insightful thought-provoking comments and constructive criticisms at different stages of my research allowed me to stay focused to core of research ideas. I am grateful to him for holding me to a high research standard and enforcing strict validations for each research result, and thus teaching me how to do research.

There is no way to express how much it meant to me to have been a member and faculty of discrete mathematics department for all meaningful discussion on different plethora of research topics. The brilliant friends, colleagues and and all the other current and former Lab graduates students and visitors that I know inspired me over my stay in Moscow. Also, thanks to active communities of Python and Matlab.

I would like to thank all the people associated with Department of Discrete Mathematics, fellow students and teachers who have been source of thoughts, inspiration and smiles. Acknowledgement will be incomplete without expressing sincere gratitude to Prof. Andrei Raigorodskii for providing excellent infrastructure, guidance and scholastic path to realize our dreams into reality.

Though only some name appears on the cover of this thesis, a great many people have contributed to its production. I owe my deep gratitude to all those people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

In conclusion, I recognize that this research would not have been possible without the financial assistance from Russian Government and MIPT, and express my gratitude to those agencies.

Pankaj Kumar

Abstract

The rapidly progressing digital revolution have infused enormous amount of images and video, which is growing constantly. Dealing with this large scale data calls for software application for enhancing image quality, classification, segmentation and recognition is the topic of Image Processing. In any image processing task an image is usually represented as a vector by concatenating each row or column. The dimension of the image vector is very high and typically embedded on a non-linear dimensional manifold, whose dimension is much smaller than that of the original data space. Manifold learning algorithms extract a submanifold embedded in a high-dimensional space, thus widely used in in the field of computer vision for dimensionality reduction, noise handling, classification and segmentation.

In the first part of this thesis, we provide mathematical notions necessary for under standing the intuition behind manifold learning. After introducing the necessary tools, we review the representative sample of manifold learning, mathematical developments, as well as some interesting applications.

The second part of thesis focuses on application of manifold learning in optical character recognition, in particular MNIST database of handwritten digits recognition. We use representative manifold learning algorithms together with supervised learning model such as KNN, SVM and CNN to classify digit with high accuracy.

In the spirit of manifold learning algorithms, the third part of this thesis tackles the problem of anomaly detection in image. We use dimension reduction property of diffusion maps together with multiscale approach based on Laplacian pyramid representation to detect anomaly from background pixels.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

The rapidly progressing digital revolution have infused enormous amount of images and video, which is growing constantly. Dealing with this large scale data calls for software application for enhancing image quality, which is integral part in Image Processing. Although image processing algorithms are getting accurate day by day, there is constant pressure on algorithms to deal with noise. It is inherent to the acquisition tools. State of the art preprocessing and postprocessing algorithms are being developed to suppress noise without altering the important content. The noise in an image is reduced by convolving the image with a Gaussian function. The above common approach blurs significant features and destroys some of the geometric information in the image [1]. Image processing task are often tackled with geometric methods, which targets to understand the geometric configuration and relation between the observed objects. Manifold learning algorithms are one such geometric methods.

1.2 Manifold Learning

Manifold learning is form of unsupervised machine learning algorithms, which extract low-dimensional structure from high dimensional data. These algorithms typically try to unfold the underlying manifold so that Euclidean distance in the new space is a meaningful measure of distance metrics between any pair of points. Manifold learning assumes that the data lies approximately on a low dimensional surface embedded in a high dimensional space [2]. It can be further illustrated using figure 1.1, where manifold learning algorithms for non-linear data builds an embedding function f mapping $\mathcal{M}$ to $\mathbb{R}^2$.

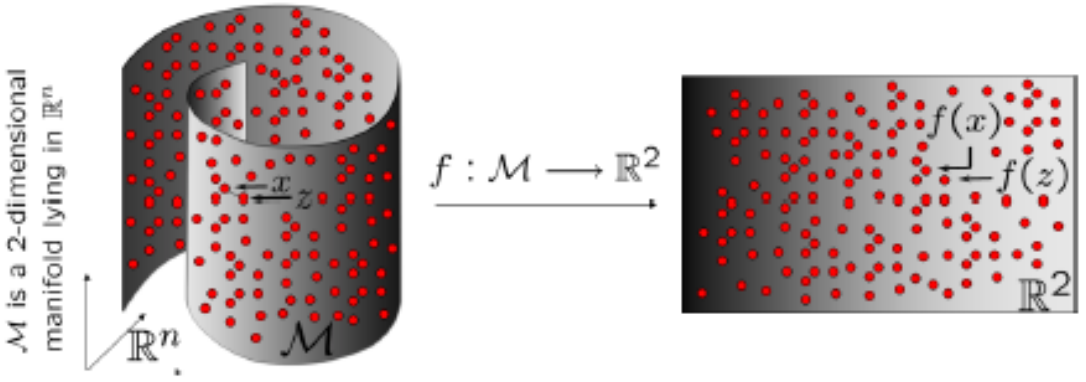


FIGURE 1.1: Manifold Learning [3]

1.3 Motivation

The rapidly progressing digital revolution have infused enormous amount of images and video, which is growing constantly. Dealing with this large scale data calls for algorithms, which can decipher meaningful information from the image of any quality. In any particular system, each image sequence is a point in a space of dimension equal to the number of image pixels. This in turn makes an observation space of possibly thousands/millions of dimensions. The one way to see the hope in the image data is to assume that the data is embedded as sub-manifold in high dimensional space in nonlinear way. There is no harm in assuming that that the number of free parameters remains the same throughout the observations. In line with former assumption, we also assume that there is spatially smooth variation of the parameters, then we have geometric restrictions which can be well modeled as a manifold. Learning this manifold is a natural approach to the problem of manifold learning algorithms, with the advantage of allowing nonlinear dimensionality reduction, clustering and classification.

The problem of classification of the digits is perhaps one of the most widely studied among machine learning communities. A basic requirement for credibility of classification algorithms requires a high accuracy on MNIST datasets. Manifold learning algorithms perform under the lens of curse of dimensionality and can be best tools to classify digits. Adding to the same, the experimental results show that manifold learning algorithms can work pretty well with toy examples rather real world data. The jink associated with manifold learning has to be cleared out.

The stimulating task of anomaly detection have been studied by many researchers over the past decades. Manifold leaning algorithms too got their hands dirty to automate the anomaly detection from background pixels, but few succeed with data driven approach. We too wanted to travel in the quest of solution to the above problem using manifold learning algorithms, searching for best fit with convincing generic approach for anomaly detection.

1.4 Contribution

A through discussion of manifold learning methods require much deeper knowledge of manifolds, topology and differential geometry. In this thesis, we alternatively provides intuitive ideas behind the manifold learning algorithms along with a brief description of their mathematical notions.

The research exploring the comparison of manifold learning algorithms is quite shallow. There are few theoretical comparisons results for these algorithms, but the primary evaluation methodology has been to run the algorithm on artificial data sets and do comparison. We use MNIST datasets with optical character recognition in mind to evaluate the performance of the manifold learning algorithms.

Motivated by recent study, we presents a novel manifold learning approach for high dimensional data, with emphasis on the problem of anomaly detection in image.

1.5 Organization

The remainder of this thesis is organized as follows. Chapter 2 provides mathematical background for understating the intuition behind manifold learning. After suitable mathematical background, Chapter 3 provides review of intuitive ideas behind the manifold learning algorithms along with a brief description of their mathematical notions and application to image data. With necessary theory and algorithms in place, In Chapter 4 we do comparison of representative manifold learning algorithms using MNIST dataset. The objective here is to accurately recognize optical character, which is in the form of handwritten digits. In Chapter 5, we presents a novel manifold learning approach for high dimensional data with emphasis on the problem of anomaly detection in image. Finally Chapter 6 concludes this this thesis summary and future directions.

Chapter 2

Mathematical Background

Overview

This chapter is dedicated to provide mathematical notions necessary for under standing the intuition behind manifold learning. Further, we present the most representative manifold learning algorithms which are used in this thesis to solve specific problems in image processing and anomaly detection. We follow the structure from Nicolas Thorstensen [1] and Boothby [5].

2.1 Metric Space

Use of near and far is intuitive yet rigorous at the same time, which is rare in mathematics. Classifying the relationship between two 'primitives', whether they are close or far apart is some time not useful for the computation. A metric space is the mathematical construct of redefining the near and far primitive idea, which is useful in computation.

Definition 2.1. A metric on an arbitrary abstract set $\mathbb{X}$ is a function $d : \mathbb{X} \times \mathbb{X} \to [0, \infty)$ such that for all $a, b, c \in \mathbb{X}$, the following conditions are satisfied:

1. Non-negativity: $d(a, b) \geq 0 and d(a, b) = 0 \Leftrightarrow a = b$
2. Symmetry: $d(a, b) = d(b, a)$
3. Triangle inequality: $d(a, c) \leq d(a, b) + d(b, c)$

Then we say that d is a *metric* on $\mathbb{X}$ and that $(\mathbb{X}, d)$ is a *metric space*.

A quite known instance of a metric space is the three dimensional Euclidean space $\mathbb{R}^3$ with the Euclidean metric.

Definition 2.2. Let V be a vector space over $\mathbb{R}$ and $N : V \to \mathbb{R}$ a map such that, $N(\mathbf{v}) = \|\mathbf{v}\|$, the following condition holds:

(i) Non-negativity: $\|\mathbf{v}\| \geq 0$ for all $\mathbf{v} \in V$. If $\|\mathbf{v}\| = 0$, then $\mathbf{v} = \mathbf{0}$.

(ii) Linearity: If $\lambda \in \mathbb{R}$ and $\mathbf{v} \in V$, then $\|\lambda\mathbf{v}\| = |\lambda|\|\mathbf{v}\|$.

(iii)Triangle inequality: If $\mathbf{v}_1$, $\mathbf{v}_2 \in V$, then $\|\mathbf{v}_1\| + \|\mathbf{v}_2\| \geq \|\mathbf{v}_1 + \mathbf{v}_2\|$.

Then we call $\| \ \|$ a *norm* and say that $(V, \| \ \|)$ is a *normed vector space*.

Any normed vector space can be made into a metric space by the condition $d(\mathbf{v}_1, \mathbf{v}_2) = \|\mathbf{v}_1 - \mathbf{v}_2\|$.

Definition 2.3. Let V be a vector space over $\mathbb{R}$ and $M : V \times V \to \mathbb{R}$ a map such that, writing $M(\mathbf{v}_1, \mathbf{v}_2) = \langle \mathbf{v}_1, \mathbf{v}_2 \rangle$, the following results hold for $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3 \in V$, $\lambda \in \mathbb{R}$.

(i) $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle \geq 0$.

(ii) If $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle = 0$, then $\mathbf{v}_1 = \mathbf{0}$.

(iii) $\langle \mathbf{v}_1, \mathbf{v}_2 \rangle = \langle \mathbf{v}_1, \mathbf{v}_2 \rangle$.

(iv) $\langle \mathbf{v}_1 + \mathbf{v}_3, \mathbf{v}_2 \rangle = \langle \mathbf{v}_1, \mathbf{v}_2 \rangle + \langle \mathbf{v}_3, \mathbf{v}_2 \rangle$.

(v) $\langle \lambda\mathbf{v}_1, \mathbf{v}_2 \rangle = \lambda\langle \mathbf{v}_1, \mathbf{v}_2 \rangle$.

Then we call $\langle \ , \ \rangle$ an *inner product* and say that $(V, \langle \ , \ \rangle)$ is an *inner product space*.

Working on $\mathbb{R}^n$ to make it vector space, then $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{j=1}^{n} x_j y_j$ is an inner product. We notice that the norm derived from inner product is called Euclidean norm.

2.2 Topology

With the metric structure in hand, the idea of 'near' and 'far' between elements of metric space can be quantified by characterizing the connectivity of sets through small neighborhoods of elements which gives rise to a topology of the sets.

Definition 2.4. Let $(\mathbb{X}, d)$ be a metric space and element $x \in \mathbb{X}$. If $r > 0$, then $B_{open}(x, r) = \{y \ : \ d(x, y) < r\}$ is called as *open ball* with centre $\mathbf{x}$ and radius r. Similarly, $B_{closed}(x, r) = \{y \ : \ d(x, y) \leq r\}$ is called closed ball.

The topology of $\mathbb{X}$ induce by the metric is easy to study.

Definition 2.5. Let $\mathbb{X}$ be a set and τ a collection of subsets of $\mathbb{X}$ with the following properties.

(i) The empty set $\emptyset \in \tau$ and the space $\mathbb{X} \in \tau$.

(ii) If $U_\alpha \in \tau$ for all $\alpha \in A$, then $\bigcup_{\alpha \in A} U_\alpha \in \tau$.

(iii) If $U_j \in \tau$ for all $1 \leq j \leq n$, then $\bigcap_{j=1}^{n} U_j \in \tau$.

Then we say that τ is a *topology* on $\mathbb{X}$ and that $(\mathbb{X}, \tau)$ is a *topological space*.

If $(\mathbb{X}, d)$ is a metric space, then the collection of open sets forms a topology. The topology of a set allows one to study properties such as connectivity.

2.2.1 Connectedness

The metric space $(\mathbb{X}, d)$ is not connected if it is the union of two disjoint open nonempty sets. Similarly, the converse of earlier implies that $(\mathbb{X}, d)$ is connected. Mathematically, It can be defined as:

Definition 2.6. A topological space $(\mathbb{X}, \tau)$ is said to be *disconnected* if we can find non-empty open sets V_1 and V_2 such that $V_1 \cup V_2 = \mathbb{X}$ and $V_1 \cap V_2 = \emptyset$. A space which is not disconnected is called *connected*.

2.2.2 Neighborhoods

Hausdorff was the first one to define topologies in terms of neighborhoods. It always appears to be technically easier to define topologies in terms of open sets as we have been doing it so far. Generally, It can be defined as:

Definition 2.7. Let $(\mathbb{X}, \tau)$ be a topological space. If $x \in \mathbb{X}$, we say that N is a *neighbourhood* of x if we can find $U \in \tau$ with $x \in U \subseteq N$.

2.3 Manifolds

A manifold is a topological space that locally resembles Euclidean space near each point. More precisely, they are spaces that locally look like the much more familiar Euclidean spaces. This allows the well-understood notions on Euclidean spaces to be generalised to manifolds rather directly. Using all the above definition, we now define manifold.

Definition 2.8. A d-dimensional manifold is a topological space 2.5 in which points can be separated by neighborhoods 2.7 and where every point has a neighborhood that is homeomorphically mapped onto an open Euclidean ddimensional ball [1].

As the general definition is not apt to perform vector calculus on a manifold, The the notion of a differentiable manifold comes hand in hand with the concept of coordinate chart 2.1. It can be thought of as assigning a set of coordinates to the points in the neighborhood U.

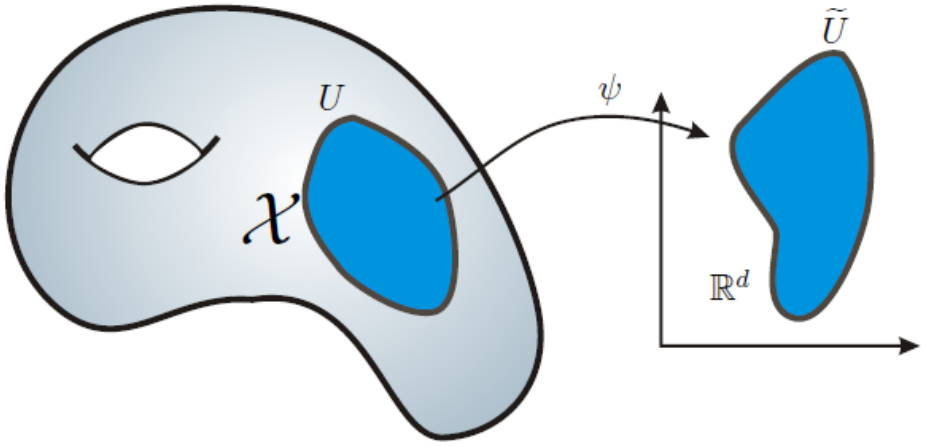


FIGURE 2.1: Coordinate Chart[3]

To provide an entire description of the manifold, several chart over manifold is used rather single. Transition map with two chart, corresponds to a change of coordinates is depicted in the figure 2.2.

2.4 Tangent Spaces

The notion of linear approximation of a surface in vector calculus translates to tangent space in differential geometry. The tangent space to a manifold $\mathcal{X}$ at a point x is the closest flat approximation to $\mathcal{X}$ at that point. If the dimension of $\mathcal{X}$ is n, then the tangent space is an $\mathbb{R}^n$ grazing $\mathcal{X}$ at x, as shown in figure 2.3. For detail discussion please refer work by Nicolas Thorstensen [1] and Boothby [5].

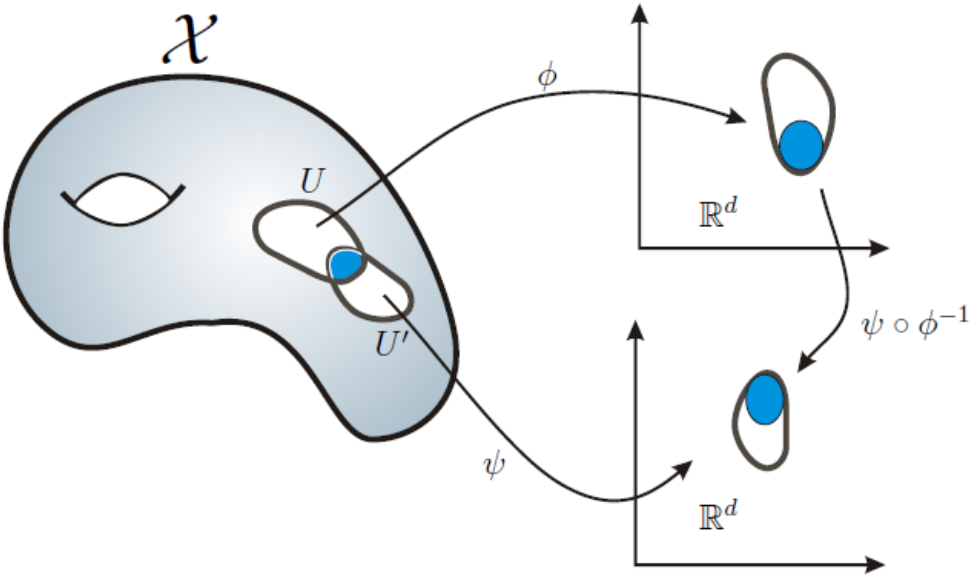


FIGURE 2.2: Transition map.[3]

2.4.1 Embedding

When talking about the relationship between two topological objects, it is interesting to imagine a mapping from one object to the other that somehow preserves its original properties.

Let $\mathbb{X}$ and $\mathbb{Y}$ be two topological objects of the same type, a function $\phi\colon \mathbb{X} \to \mathbb{Y}$ is an embedding of $\mathbb{X}$ into $\mathbb{Y}$ if ϕ is an isomorphism which preserves the original properties of $\mathbb{X}$, where such properties are relative to the type of the objects at hand. Shortly, $\mathbb{X}$ is said to be *embedded* in $\mathbb{Y}$.

Example 2.1 (Embedding of Spaces)**.** *Consider the vector spaces* $\mathbb{R}^p$ *and* $\mathbb{R}^q, p \leq q, p > 0$ *and the isomorphism* $t\colon \mathbb{R}^q \to \mathbb{R}^t \mid t(x) = [x|\bar{0}] = y$*, where* y *is the vector* x *concatenated with* $p - q$ *zeros.* $\mathbb{R}^p$ *is embedded on* $\mathbb{R}^q$*, as the operations sum and scalar multiplication are preserved.*

The same definition applies to manifolds too.

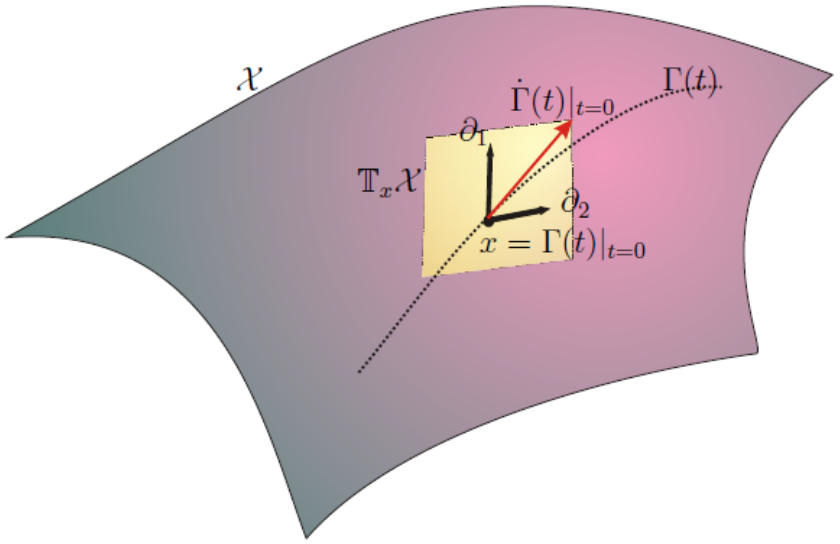


FIGURE 2.3: Tangent Space.[3]

Chapter 3

Manifold Learning

Overview

Manifold learning is an emerging and promising approach in nonparametric dimension reduction widely used in image processing, data mining, signals processing and computer vision. The core theme of manifold leaning is to find the most succinct low dimensional structure that is embedded in a higher dimensional space. Using earlier, the algorithms can work directly in the lower dimensional latent space of the manifold rather high dimensional data space and thus get computationally efficient. In this chapter, we review the representative sample of manifold learning, mathematical developments, as well as some interesting applications.

3.1 Introduction

Most of the datasets encountered in image processing often consist of a large number of samples each of which are themselves of high dimension. Further processing of the datum is done either treating it directly or extracting complex features before processing. Nevertheless, wither its entire datum or a set extracted features, the dimensionality of problems usually remain high. This in turn slows down the processing considerably or sometimes making the treatment even intractable. The task to reducing the computational burden is to reduce the dimensionality of the data. Dimensionality reduction can be thought of as mathematical mapping of high-dimensional data into a meaningful representation of the intrinsic dimensionality. The intrinsic dimensionality of a data set can be defined as the lowest number of variables that can preserve the geometrical structure of the data.

3.2 Problem Definition

Given a set of of high-dimensional training instances $\mathbb{X} = \{x_1, x_2, ..., x_N\}$, where $x_i \in \mathbb{R}^D$. We assume that $\mathbb{X}$ approximately lie on a smooth manifold $\mathcal{X}$. The idea of manifold learning algorithms is to find an embedding set $\mathbb{Y} = \{y_1, y_2, ..., y_N\}$ of $\mathbb{X}$ in low dimension space $\mathbb{R}^d$, where $d < D$. The local manifold structure formed by $\mathbb{X}$ in the original space $\mathbb{R}^D$ is preserved in the embedded space $\mathbb{R}^d$.

Manifold Learning can be mainly divided into linear and non linear methods. Linear methods, which have long been used for analyzing multivariate data, include principal component analysis (PCA) and multidimensional scaling (MDS). A further distinction within non-linear methods is to divide the methods into two classes: one is to preserve the global geometric structure of manifold, such as Isomap [6] and the other one is to preserve the local neighborhood geometric structure, such as LLE [7].

3.3 Linear Methods

3.3.1 Principle Component Analysis

Principal component analysis (PCA) is a standard algorithm to explain the dispersion of a point cloud by projecting data onto a carefully chosen linear subspace. It reduces the dimensionality of the point cloud while keeping the maximum of variance information. The dispersion in linear subspaces of higher dimension is captured by by building an orthonormal basis such that the projection along each axis has a decreasing maximum variance.

We consider n points $\mathbf{X} = \{x_1, x_2, ..., x_n\}$ of dimension D from $\mathbb{R}^D$. Also, let us assume V^* be a d-dimensional subspace of $\mathbb{R}^D$ and let $u = \{u_1, u_2, ..., u_D\}$ be an orthonormal basis of $\mathbb{R}^D$ such that $\{u_1, u_2, ..., u_d\}$ is a basis of V. Then each data point is approximated by:

$$y_n = \sum_{i=1}^{d} a_{n,i} u_i + \sum_{i=1}^{D+1} a_{n,i} u_i \tag{3.1}$$

The below energy function is used to recover $u_1, u_3, ..., u_d$.

$$\mathbb{E} = \frac{1}{N} \sum_{n=1}^{N} \|x_n - y_n\|^2 \tag{3.2}$$

This is equivalent to minimizing the approximation error.

$$x_n - y_n = \sum_{i=d+1}^{D} \{(x_n - \bar{x})^T u_i\} u_i \tag{3.3}$$

The above equation 3.3 can be best visualized in the figure 3.1.

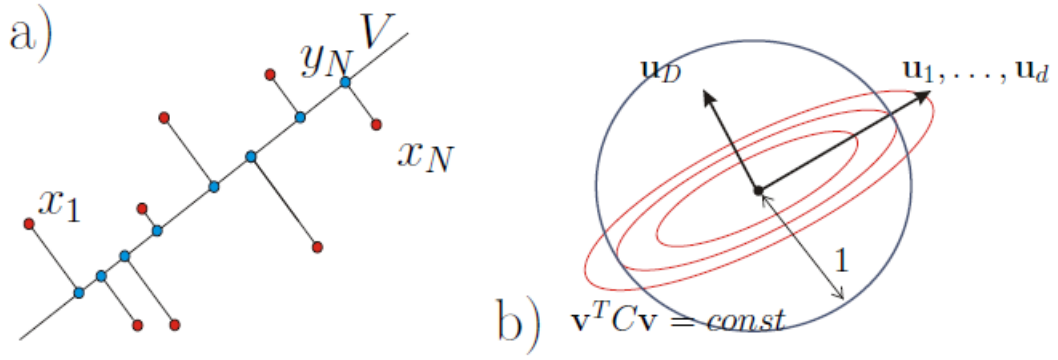


FIGURE 3.1: a) Manifold with original (red) and projected (blue) points. b) Geometry and constraint visualization of optimization function. Pictures taken from Nicolas Thorstensen works [3]

Now, we can write the energy function as

$$\mathbb{E} = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=d+1}^{D} (x_n^T u_i - \bar{x}^T u_i)^2 = \sum_{i=d+1}^{D} u_i^T \mathbf{C} u_i \tag{3.4}$$

$\mathbf{C}$ is defined as covariance matrix, $\mathbf{C} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \bar{x})(x_n - \bar{x})^2$. Our optimization problem look like

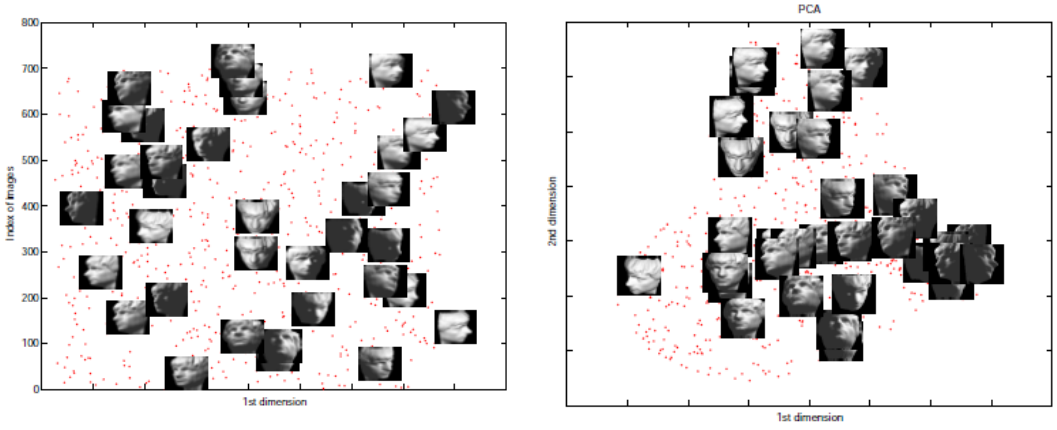
$$\begin{aligned} \underset{D}{\text{minimize}} \quad & \mathbb{E} = u_D^T \mathbf{C} u_D \\ \text{subject to} \quad & u_D^T u_D = 1 \end{aligned} \tag{3.5}$$

Now using Lagrange multipliers, we solve this constraint optimization problem as

$$u_D^T \mathbf{C} u_D + \lambda(1 - u_D^T u_D) = 0 \tag{3.6}$$

The solution off the above equation gives standard form eigenvalue problem $\mathbf{C} u_D = \lambda u_D$. The geometry of the minimization problem is illustrated in figure 3.1 b).

We take an input image 3.2 (A), which is sequence of vector in $\mathbb{R}^{4096}$ representing the brightness values of 64×64 pixel image of the face. The 1st dimension representing embedding correlates with intrinsic-dimension of one, which is left-right pose. The two-dimensional projection of original input image found by PCA is shown in figure 3.2 (B).



(A) Input

(B) Output

FIGURE 3.2: Application of PCA.

3.3.2 Multi-Dimensional Scaling

The Multi-Dimensional Scaling (MDS), is a PCA-like technique that maps the original high dimensional space to a lower dimensional space by preserving pairwise distances. Figure 3.3 shows an example of MDS, where pairwise distance is preserved. MDS [8] is applied when it difficult to calculate covariance matrix from the dataset, when data are, for instance, qualitative or infinite dimensional, when only a point-wise distance is known or also when the size $\mathbb{D}$ of the sample set is lower than the dimension of data [3].

MDS addresses the problem of finding d-dimensional Euclidean coordinates for each data sample ($x_i \in \mathbf{X}$) so that the pairwise distance (d) of their Euclidean coordinates match the original pairwise distance as closely as possible.

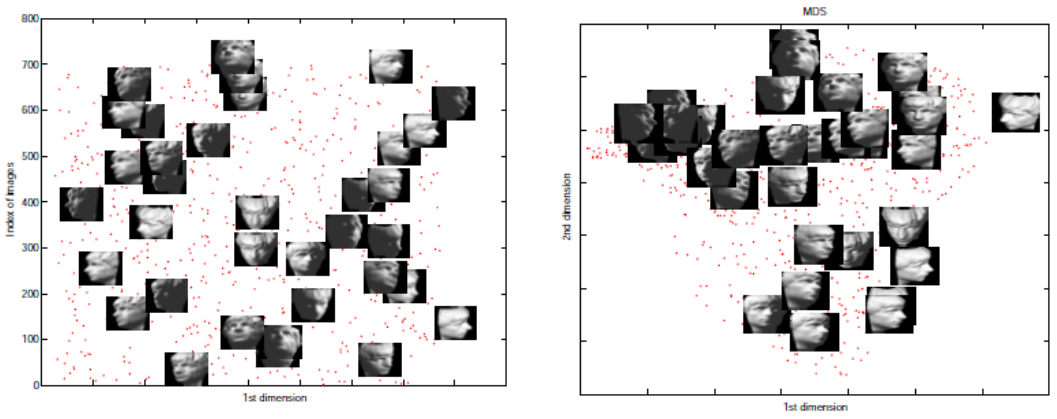
We define $N \times N$ euclidean distance or affinity matrices $\mathbf{D}$ as symmetric, $d_{ii} = 0$ and $d_{ij} > 0$, if $i \neq j$. With $\mathbf{D}$ matrix in hand, the MDS algorithm attempts to find N data points $\mathbf{Y} = y_1, y_2, y_3, ..., y_N$ in $\mathbb{R}^d$ such that distance matrix is preserved as depicted in the figure 3.4.

In particular, we try to optimize following function:

$$\underset{\mathbf{Y}}{\text{Min}} \quad \sum_{i=1}^{N}\sum_{i=1}^{N}(d_{ij}^{\mathbf{X}} - d_{ij}^{\mathbf{Y}})^2 \tag{3.7}$$

where, $d_{ij}^{\mathbf{X}}$ and $d_{ij}^{\mathbf{Y}}$ is normed distance from sample point x_i and x_i respectively. Converting distance $\mathbf{D}^{\mathbb{X}}$ matrix into kernel matrix of inner product yields:

$$\mathbf{X}^T\mathbf{X} = -\frac{1}{2}\mathbf{J}\mathbf{D}^{\mathbb{X}} \tag{3.8}$$



(A) Input (B) Output

FIGURE 3.3: Application of MDS.

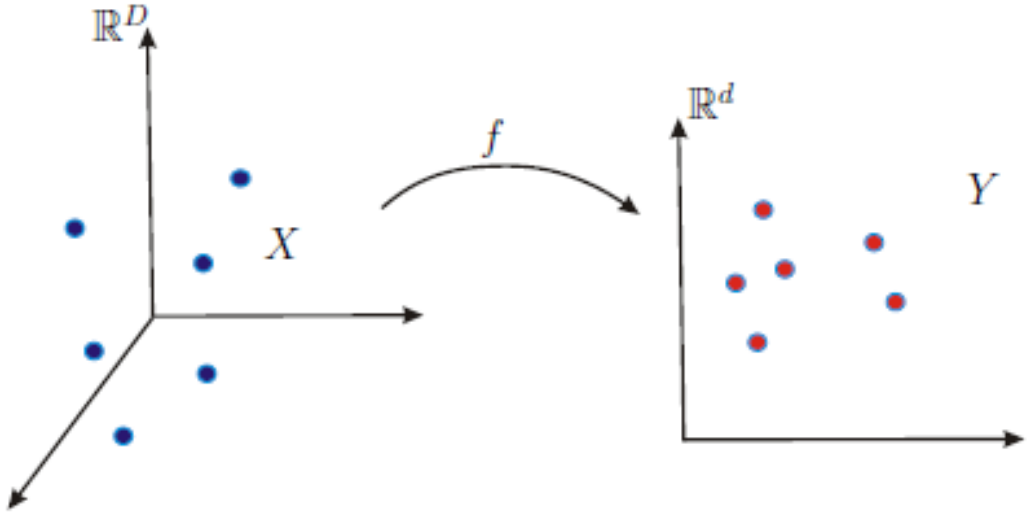


FIGURE 3.4: Multi-Dimensional Scaling

where $\mathbf{J} = \mathcal{I}d_N - \frac{1}{N}\mathbf{1}\mathbf{1}^T$. Now the equation 3.7 can be reduced into

$$\underset{\mathbf{Y}}{\text{Min}} \quad \sum_{i=1}^{N}\sum_{i=1}^{N}(x_i^T x_j - y_i^T y_j)^2 \tag{3.9}$$

Solving equation 3.9 as shown in Cox and Cox paper[8] gives $\mathbf{Y} = \lambda^{\frac{1}{2}}\mathbf{U}^T$. Here, $\mathbf{U}$ is eigenvector of $\mathbf{X}^T\mathbf{X}$ corresponding to top d eigenvalues, and λ is top d eigenvalues of $\mathbf{X}^T\mathbf{X}$.

MDS can be generalized into to geodesic distances between point samples from a non-linear manifold. But a non-convex energy function coming out makes it

tedious to optimize. In the next section we will provide an overview of non-linear manifold algorithms.

3.4 Non-Linear Methods

Linear methods such as PCA and MDS fails to produce desired results when the data is sampled from non linear manifolds [1]. We now review representative non-linear methods in manifold learning in this section with assumption that the data is distributed along a d-dimensional submanifold $\mathcal{X}$ embedded in $\mathbb{R}^D$.

3.4.1 Graph-based Algorithms

Manifold learning based on graph based algorithms often relies on preserving the geometric structure in the neighborhood of some points. The important step is to construct a graph using K-nearest neighbor graph or ε-neighborhood and apply MDS as previously detailed.

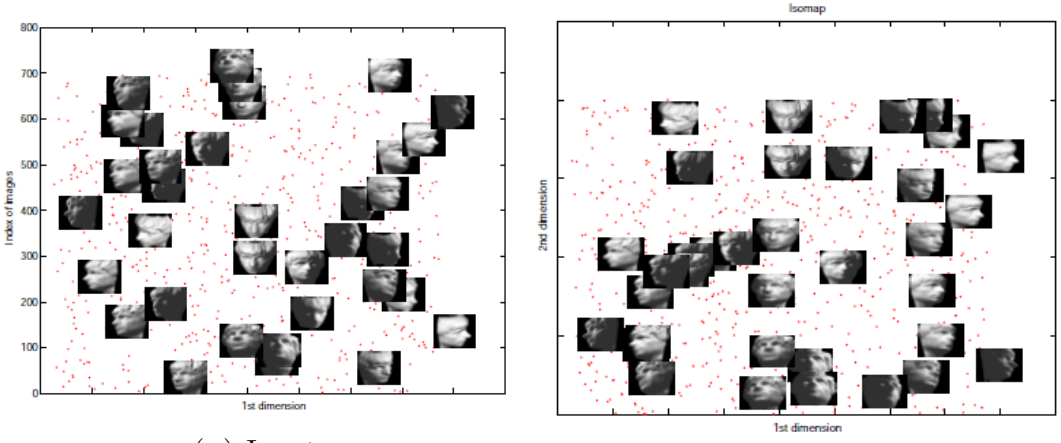
3.4.1.1 Isomap

Isomap was proposed by Joshua Tenenbaum, Vin de Silva and John Langford in Science [6]. It is an extended version of MDS for data lying on smooth non-linear manifold. Unlike MDS, the pairwise distance matrix in Isomap is replaced by the matrix of pairwise geodesic distances approximated by distances in graphs. The algorithm proceeds in three steps. First a distance $d(x; y)$ is considered in the data space. Then, a neighborhood graph based on a ε-neighborhood or k-nearest neighbor points is build with following weights $D_{ij} = d(xi; xj)$ if an edge link exists between points x_i and x_j , otherwise $D_{ij} = 1$. The pairwise distance matrix $\mathbf{D}$ is then calculated using a Dijkstra-like algorithm in the graph. Finally, the data via MDS is embedded so as to preserve these distances. A two-dimensional projection using isomap is shown in the figure 3.5 with k=6.

Precautionary measure should be taken while calculating distances between far apart points on the manifold, which can be very close in the data space as discussed in the figure 3.6.

3.4.1.2 Locally linear Embedding

Locally linear Embedding (LLE) is a manifold mapping dimension reduction algorithm introduced by Roweis & Saul [7]. It aims at recovering the low dimensional geometry of the data by looking at the local interaction between data points. The construction of a nearest neighbor graph recovers the local interactions. Then



(A) Input

(B) Output

FIGURE 3.5: Application of Isomap

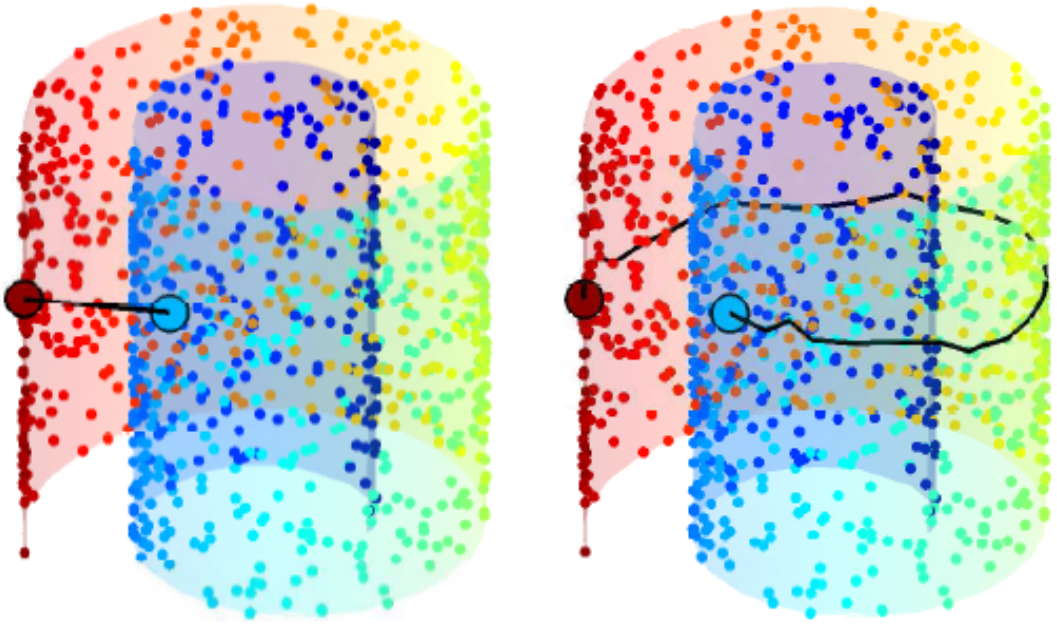


FIGURE 3.6: Multi-Dimensional Scaling. Left: Distance between two points using a distance in the data space. Right: Distance between two points using an approximated geodesic path. Image taken from Patrick Etyngier work [3].

further processing is done to reduce the dimensionality of the data and provide a parametrization in a d- dimensional hyperplane. LLE algorithm can be summarized in the following steps:

1. Neighborhood graph $\mathcal{G}$ is build on the dataset $\mathbb{X}$ using K-nearest neighbour method.

2. Weights W_{ij} are calculated to each of the nearest neighbours pair (x_i, y_i) by minimising the cost function:

$$\varepsilon(W) = \sum_i |x_i - \sum_{j=1}^{k} W_{ij} x_j|^2 \tag{3.10}$$

3. The inner products between each point x_i and each of its nearest neighbours are computed to produce gram positive-semidefinite matrix, $G_{jk} = (x - x_j)^T(x - x_i)$.

4. The reconstruction weights are then computed using:

$$W_{ij} = \sum_k \mathbb{G} - jk^{-1}(G_{jk} + \lambda) \tag{3.11}$$

such that, $W_{ij} = 0$ if x_j is not one of the nearest neighbours of x_i and $\sum_j W_{ij} = 1$)

5. Finally the embedded $\mathbb{Y}$ which will make up the final output data are calculated by minimising a second cost function:

$$\Phi(Y) = \sum_i |y_i - \sum_j W_{ij} y_j|^2 \tag{3.12}$$

We also constraint $\sum_i y_i = 0$ to centre the projection on the origin. The other constraint is imposed in order to avoid degenerate solutions;

$$\frac{1}{n} \sum_i y_i y_i^T = I \tag{3.13}$$

where I is the d×d identity matrix.

6. This cost function now defines a quadratic form containing the N×N symmetric matrix M_{ij}:

$$\Phi(I) = \sum_{ij} M_{ij} y_i y_i^T \tag{3.14}$$

where $M_{ij} = \delta_{ij} - W_{ij} - W_{ij} + \sum_k W_{ki} W_{kj}$ and $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise.

7. The embedding $\mathbb{Y}$ is then given by the the lowest d+1 eigenvectors of the matrix M_{ij}. The bottom eigenvector representing a free translation mode of eigenvalue 0 is left out in order to find the optimum embedding.

The geometry the optimization problem is depicted in figure 3.7.

A two-dimensional projection of the original input by using with k=5 is shown in the figure 3.8.

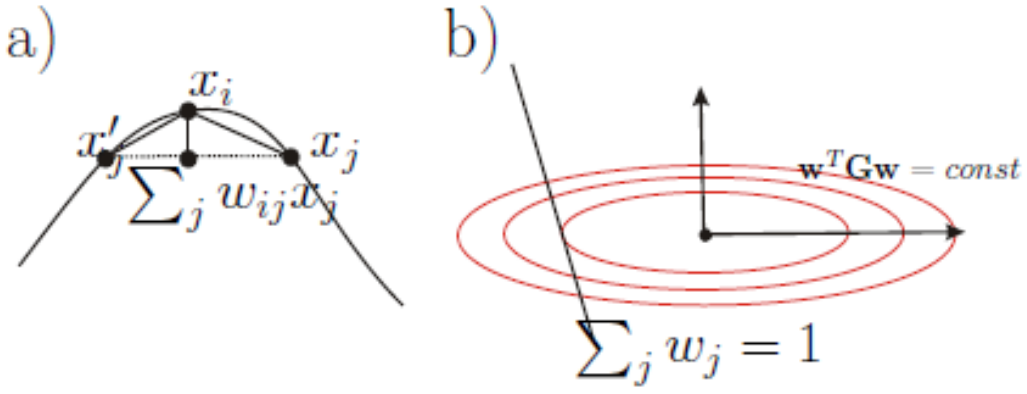
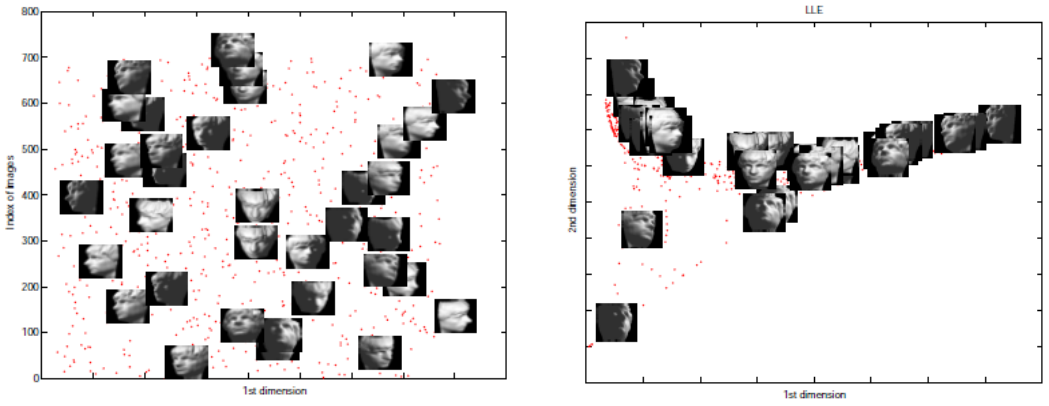


FIGURE 3.7: LLE. a) x_i is appropriated by its neighborhood (x'_j, x_j). b) The black line touching the level set at a single point defines the constraints



(A) Input

(B) Output

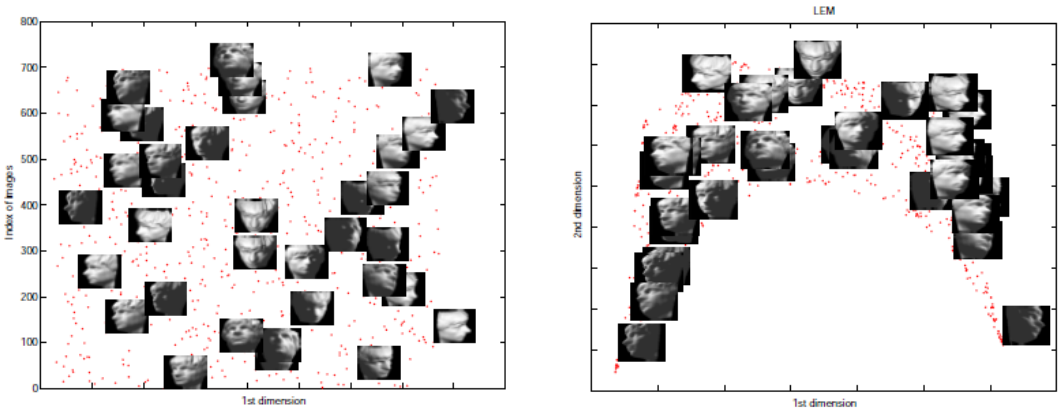
FIGURE 3.8: Application of LLE

3.4.2 Laplacian-based Algorithms

Laplacian based algorithms assume that data have a structure of low dimensional manifold embedded in a higher dimensional space. The maps are constructed into a low dimensional space preserving the local neighborhood topology. In this part, we give outlines of representative laplacian based algorithms.

3.4.2.1 Laplacian Eigenmaps

Laplacian Eigenmaps were introduced by Belkin [4]. Its intent is to embed the observed data $\mathbf{X}$ into $\mathbb{R}^d$ by first constructing the k-nearest-neighbor or ε-graph $\mathcal{G}$ from $\mathbf{X}$. In the k-nearest-neighbor graph, an edge is present between x_i and x_j if x_i is among the k nearest neighbors of x_j or vice versa. In the ε-graph, x_i and



(A) Input

(B) Output

FIGURE 3.9: Application of Laplacian Eigenmaps

x_j are adjacent if $\|x_i - x_j\|^2 < \varepsilon$ for a given threshold parameter ε. The weighted adjacency matrix of $\mathcal{G}$ is defined as $\mathbf{W}$,

$$\mathbf{W}_{ij} = \begin{cases} \mathbf{K}_{ij} & \text{if } \{i,j\} \in E \\ 0 & \text{else,} \end{cases}$$

and let $\mathbf{D} \in \mathbb{R}^{N \times N}$ be the diagonal matrix defined by $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$ for $i \in [N]$.

Then the normalized weighted graph Laplacian of $\mathbf{G}$ [4] is given by $\mathbf{W} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. If we represent eigendecomposition of $\mathbf{W}$ by $\mathbf{W} = \mathbf{U}\Lambda\mathbf{U}^\top$ with the diagonal entries of Λ non-increasing, then Laplacian eigenmaps embeds $\mathbf{X}$ via $\mathbf{U}[:, 2 : d+1]$—the first d nontrivial eigenvectors of $\mathbf{W}$. The local geometry of $\mathbf{X}$ is optimally preserved in a least squares sense by the former.

A two-dimensional projection of the original input by using with k=9 is shown in the figure 3.9.

3.4.2.2 Diffusion Maps

The objective of Diffusion Maps (DM) is to define a metric, named the diffusion distance, that measures the connectivity between points in an arbitrary set. We will follow the construction of DM as described in the paper by Lindenbaum and others [9].

Representing the high dimensional input dataset $\boldsymbol{X}$, the DM framework contains the following steps:

1. A kernel function $\mathcal{K} : \boldsymbol{X} \times \boldsymbol{X} \longrightarrow \mathbb{R}$ is chosen to define an anisotropic transition matrix, represented by $\boldsymbol{K} \in \mathbb{R}^{D \times D}$. It satisfies the following properties for all $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \boldsymbol{X}$.

(a) Symmetry: $K_{i,j} = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathcal{K}(\boldsymbol{x}_j, \boldsymbol{x}_i)$,
(b) Positive semi-definiteness: $\boldsymbol{v}_i^T \boldsymbol{K} \boldsymbol{v}_i \geq 0$ for all $\boldsymbol{v}_i \in \mathbb{R}^D$ and
(c) Non-Negativity $\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$.

2. When we normalize the kernel using $\boldsymbol{M}$; where $M_{i,i} = \sum_j K_{i,j}$, the following matrix elements are computed:

$$P^x_{i,j} = \mathcal{P}(\boldsymbol{x}_i, \boldsymbol{x}_j) = [\boldsymbol{M}^{-1}\boldsymbol{K}]_{i,j}. \tag{3.15}$$

The resulting matrix $\boldsymbol{P}^x \in \mathbb{R}^{D\times D}$ is actually transition kernel of a Markov chain on $\boldsymbol{X}$ such that the expression $[(\boldsymbol{P}^x)^t]_{i,j} = p_t(\boldsymbol{x}_i,\boldsymbol{x}_j)$ describes the transition probability from point $\boldsymbol{x}_i$ to point $\boldsymbol{x}_j$ in t steps.

3. Next, the spectral decomposition is applied to matrix $\boldsymbol{P}^x$ to obtain a sequence of eigenvalues $\{\lambda_d\}$ and normalized eigenvectors $\{\boldsymbol{\psi}_d\}$ that satisfies $\boldsymbol{P}^x\boldsymbol{\psi}_d = \lambda_m\boldsymbol{\psi}_d, d = 0, ..., D-1$;

4. Defining a new representation for the dataset $\boldsymbol{X}$

$$\boldsymbol{\Psi}_t(\boldsymbol{x}_i) : \boldsymbol{x}_i \longmapsto \left[\lambda_1^t\psi_1[i], \lambda_2^t\psi_2[i], \lambda_3^t\psi_3[i], ..., \lambda_{D-1}^t\psi_{D-1}[i]\right]^T \in \mathbb{R}^{D-1}, \tag{3.16}$$

where t is the selected number of steps and $\psi_d[i]$ denotes the i^{th} element of $\boldsymbol{\psi}_d$.
Now the Euclidian distance between two data points is equal to the weighted L_2 distance between the conditional probabilities $p_t(\boldsymbol{x}_i, :)$, and $p_t(\boldsymbol{x}_j, :)$, $i, j = 1, ..., D$ (the i-th and j-th rows of $\boldsymbol{P}^t$), which is referred as the Diffusion Distance

$$\mathcal{D}_t^2(\boldsymbol{x}_i, \boldsymbol{x}_j) = ||\boldsymbol{\Psi}_t(\boldsymbol{x}_i) - \boldsymbol{\Psi}_t(\boldsymbol{x}_j)||^2 = \sum_{d\geq 1} \lambda_d^{2t}(\psi_d[i] - \psi_d[j])^2 = ||p_t(\boldsymbol{x}_i, :) - p_t(\boldsymbol{x}_j, :)||^2_{\boldsymbol{W}^{-1}}, \tag{3.17}$$

where $\boldsymbol{W}$ is a diagonal matrix with elements $W_{i,i} = \frac{D_{i,i}}{\sum_{i=1}^M D_{i,i}}$.

5. The desired accuracy $\delta \geq 0$ is chosen for the diffusion distance defined by Eq. (3.17) such that $s(\delta, t) = \max\{\ell \in \mathbb{N} \text{ such that } |\lambda_\ell|^t > \delta|\lambda_1|^t\}$. By using δ, a new mapping of $s(\delta, t)$ dimensions is defined as
$\Psi_t^{(\delta)} : X \to \left[\lambda_1^t\psi_1[i], \lambda_2^t\psi_2(i), \lambda_3^t\psi_3[i], ..., \lambda_s^t\psi_s[i]\right]^T \in \mathbb{R}^{s(\delta,t)}$.

As discussed above, diffusion distance reflects the intrinsic geometry of the data set defined via the adjacency graph in a diffusion process.

3.5 Other manifold learning algorithms

Many other Laplacian based techniques exist in literature to reduce the dimensionality of a point cloud which were not dicussed in this chapter. For example,

Maximum Variance Unfolding (MVU)[3], which links most of Laplacian-based methods such as LE, LLE and Isomap. Another important approach which is derived from the LLE framework is called Hessian eigenmaps[3]. The methods is widely used when the underlying embedding is not convex. In figure 3.10 taken from Thorstensen work [1] lists the general properties for manifold learning algorithms.

3.6 Intrinsic Dimension

Intrinsic Dimension (ID) is defined as the minimal number of parameters necessary to represent the variability of a data set. It is a a key priori knowledge in computer vision and image processing to improve their performance. Mathematically, a formal definition of the intrinsic dimension is the following, due to [10].

Definition 3.1. A data set $\mathbf{X} \subseteq \mathbb{R}^{\mathbf{D}}$ said to have intrinsic dimension equal to $\mathbf{d}$ if its elements lie entirely, without information loss, within a $\mathbf{d}$-dimensional manifold of $\mathbb{R}^{\mathbf{D}}$, where $\mathbf{d} < \mathbf{D}$.

Most of the existing approach to estimate the intrinsic dimension can be roughly divided into two groups: eigenvalue or projection methods, and geometric methods. Projection or eigenvalues methods, estimate the ID by thresholding the observed eigenspectrum i.e. the spectrum of the eigenvalues output by the global or local PCA. It may be good method for exploratory data analysis, where one might plot the eigenvalues and look for a clear-cut boundary, but not for providing reliable estimates of intrinsic dimension.

The other, geometric methods exploit the intrinsic geometry of the dataset and are most often based on fractal dimensions or nearest neighbor (NN) distances [11].

Finding out the ID of the dimension estimation might become a very complex problem, when data lie on a smooth manifold. In this thesis, we used method proposed by Levina and Bickel [11] to verify the ID computed by projection and geometric methods. They derive ID by applying the principle of maximum likelihood to the distances between close neighbors.

Method	Locality	Linearity
PCA	global structure preserved	linear
MDS	global structure preserved	linear
ISOMAP	global structure preserved	nonlinear
LLE	local structure preserved	nonlinear
HLLE	local structure preserved	nonlinear
Laplacian Eigenmaps	local structure preserved	nonlinear
Diffusion maps	local structure preserved	nonlinear

Method	Strategy	special feature
PCA	preserves variance of data	based on inner product
MDS	preserve inter-point distance	based on distance function
ISOMAP	preserve the geodesics	graph based
LLE	preserve local neighbor	linear neighborhood approximation
HLLE	preserve local neighbor	theoretical guarantees
Laplacian Eigenmaps	preserve local neighbor	laplacian regularization
Diffusion maps	preserve local neighbor	wavelet approach and density independence

FIGURE 3.10: Overview of Manifold Learning Algorithms

Chapter 4

Optical Character Recognition

Overview

Tucked with mathematical preliminary necessary for under standing the intuition behind manifold learning, we discussed the representative manifold learning algorithms in detail. In this chapter, we now move on to some real application called optical character recognition. We motivate the importance of manifold learning algorithms discussed in chapter 5 through dimensionality reduction, clustering and classification of digits from the MNIST database of handwritten digits.

4.1 Introduction

The problem of classification of the digits is perhaps one of the most widely studied among machine learning communities. Among the classification task, optical character recognition have found applications in bank check processing, assistive technology for visually impaired users, automatic number plate recognition and many more. The initial excitement over deep learning methods was triggered by their success on the MNIST handwritten digit recognition problem [15], which was for several years the standard benchmark problem for hard, large dataset machine learning. A basic requirement for credibility of classification algorithms requires a high accuracy on MNIST datasets.

In this chapter 4, we intend to apply various manifold learning algorithms to the MNIST data set of handwritten digit images. The global plan is to evaluate manifold learning algorithms to better understand how the classification algorithms perform under the lens of curse of dimensionality. Our goal is to automatically determine what a handwritten digit image ranging from 0 to 9. The earlier experimental results doing comparison between different manifold learning algorithms show that these methods can work pretty well at least in our toy examples. We

show this results using readily available code from Matab and Python in Appendix A.

4.2 Data

The standardised MNIST database has a training set of 60,000 examples, and a test set of 10,000 examples. The images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. Then after, the images were centered in a 28×28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28×28 field. As a result, each example is of size 28×28, a total of 784 greyscale pixels. If we take training set, we 60,000 rows (images) and 785 (784 pixels and a one label) columns. each Each row component contains a value between one and zero and this describes the intensity of each pixel. On the another hand, the test set is identical to the training set except that it is unlabeled, and so only contains 784 columns instead of 785. The data is quite popular amongst machine learning communities because it requires minimal efforts on preprocessing and formatting. It is the de facto 'hello world' dataset of image processing and computer vision.

4.3 Prepossessing

The first step in any data science project is data visualization. First 50 training examples can be observed in figure 4.1 after dropping labels.

Figure 4.1 revels some interesting facts about the data. Minute but important observation shows that there is a lot of the bordering pixels having black color. We also see that a few of the depicted digits are skewed. Although, MNIST datasets are already preprocessed, but there is always scope for massaging the data better. The obvious choice to vertically align the skewed digit is Hough transform [16]. The purpose of the Hough transform is to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm for computing the Hough transform. We apply Hough transform to training data yielding vertically aligned digits except some, which were not aligned. Figure 4.2 illustrates the former discussion. We are not sure that Hough transform will increase our efficiency, but we just wanted to try considering it might have some positive effect on classification accuracy.

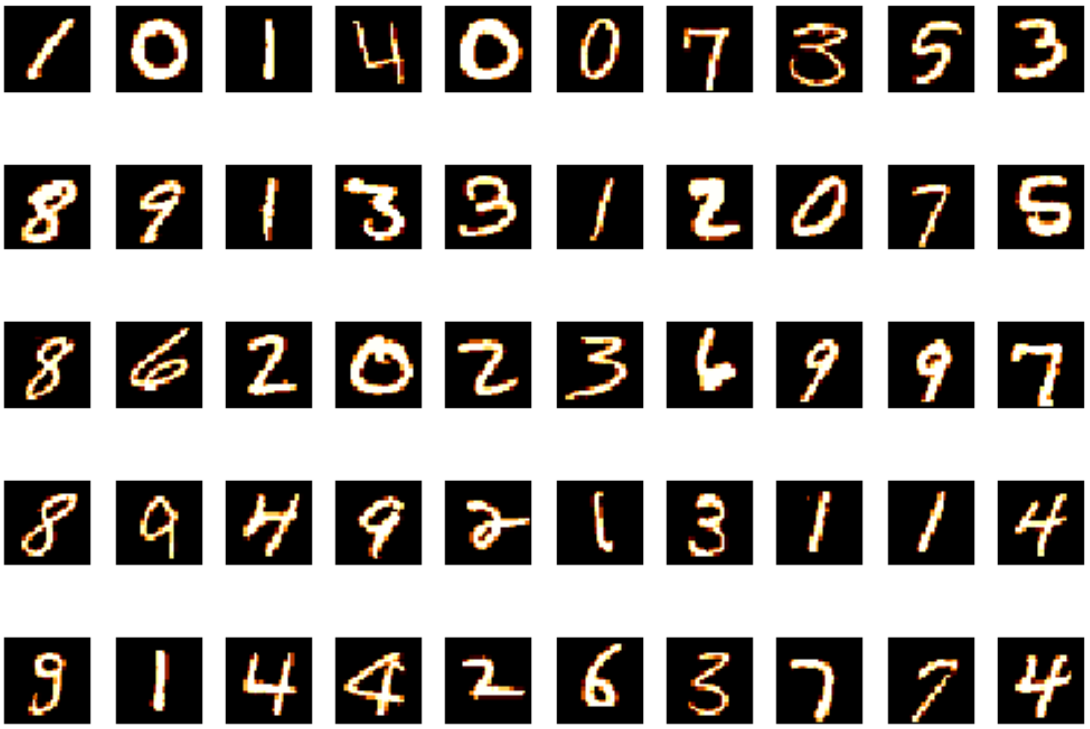


FIGURE 4.1: First look at MNIST data

4.4 Experiments and Results

With prepossessed data in our hand, we go for traditional Pearson Correlation plot in-search of meaningful insight through figure 4.3.

The Pearson Correlation plot didn't give information due to shear size of the data. As the data is still large, we apply dimension reduction.

4.4.1 Principal Component Analysis (PCA)

PCA is a widely used linear technique in image processing. It is used for simplifying a multidimensional dataset to lower dimensions for analysis, which is essential for visualization and image processing. The data is represented in a new coordinate system using PCA, in which basis vectors follow modes of greatest variance in the data. The mathematical details of the same is discussed in 3.3.1. The graphical illustration is provided in the figure 4.4.

After prepossessing of the data, the important information is to observe how the variances look like for the digits in the MNIST dataset. We take the hep of PCA to calculate the eigenvectors and eigenvalues of the covarience matrix. Working with python allows us to use standard library to ease our work. We first standardize

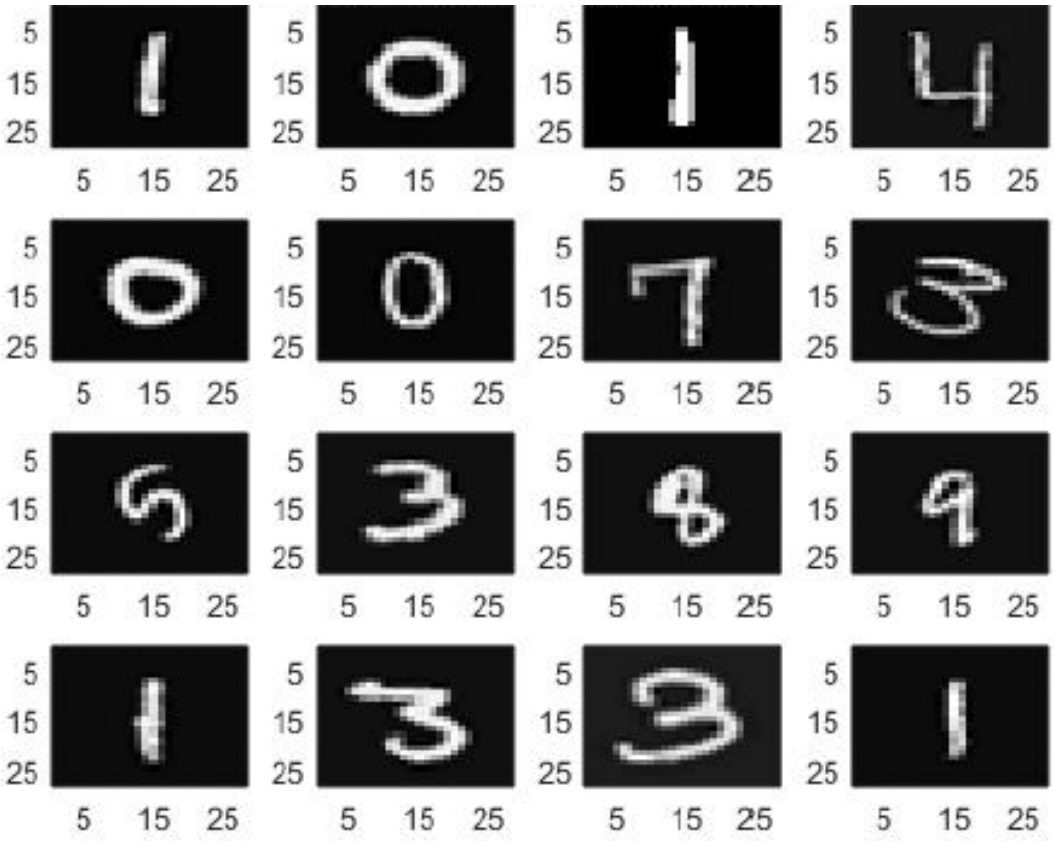


FIGURE 4.2: Hough transform

the data, then calculate the eigenvectors and eigenvalues of the covarience matrix. After shorting it, we calculate explained Variance from the eigenvalues depicted in the figure 4.5.

The distribution of the Individual and Explained variances across all features is shown in smaller picture embedded. The larger plot represents the explained variances only. It is evident from the figure 4.5 that out of our 784 features, only 200 approximately describes 90% of the explained variance. Now fitting data in PCA with 200 eigenvalues, we try to obtain the optimal eigenvectors/direction that captures the most variance. We do visual comparison of the top 5 eigenvalues to some of the other smaller ones using the figure 4.6. It is evident from the figure 4.6 that more complicated directions are being generated in the search to maximise variance in the new feature subspace.

We finally use dimension reduction using PCA in search of existing clusters if there is any. If seek the help from scatter plots to see the embedding using two components. The output is depicted in figure 4.7. As observed from the scatter plot, we don't see any clusters. This means that, for our data, the linear method, PCA is not best fit.

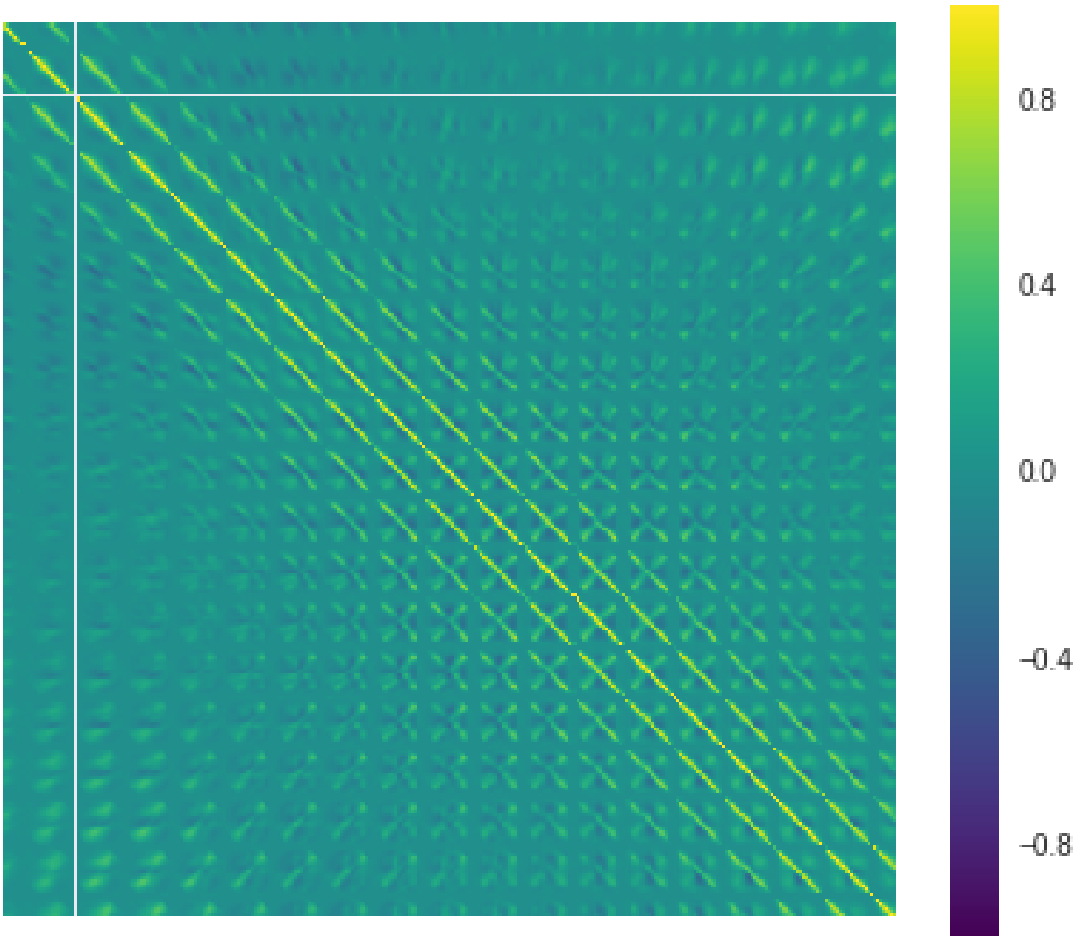


FIGURE 4.3: Pearson Correlation plot 500 observations

4.4.2 Manifold Learning

Manifold learning is widely used in visualization, classification, and information processing of the data embedding in lower dimension. Among all, data visualization is one of the most important method which uses 2D or 3D representation to give further insight into the data structure. This in turn can be used for either interpretation, data reduction and data model selection for extracting meaningful features from cumbersome representations.

For our MNIST dataset, we witnessed that linear method like PCA is not good choice. Manifold learning algorithms as discussed in chapter 3 provides means of data reduction by low dimensional manifold embedded in a high dimensional space. We use t-Distributed Stochastic Neighbour Embedding (TSNE), a manifold learning algorithm introduced by van der Maaten and Hinton [17]. TSNE aims to convert the Euclidean distances between points into conditional probabilities, where student-t distribution is then applied. This serves as metrics to calculate the similarity between one datapoint to another. The scattered plot of the first

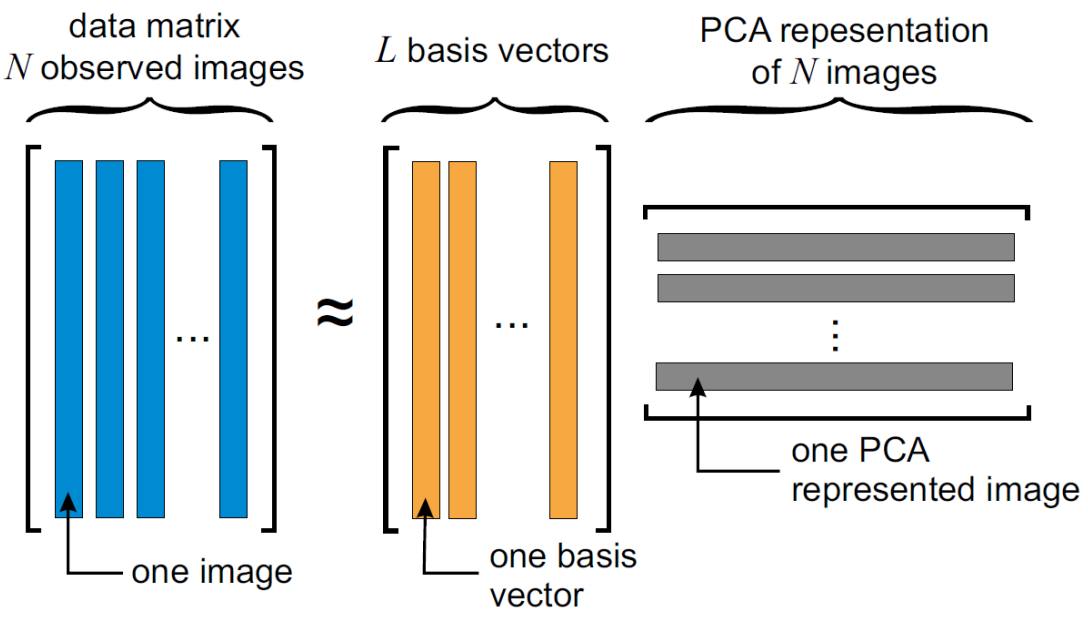


FIGURE 4.4: PCA graphical illustration

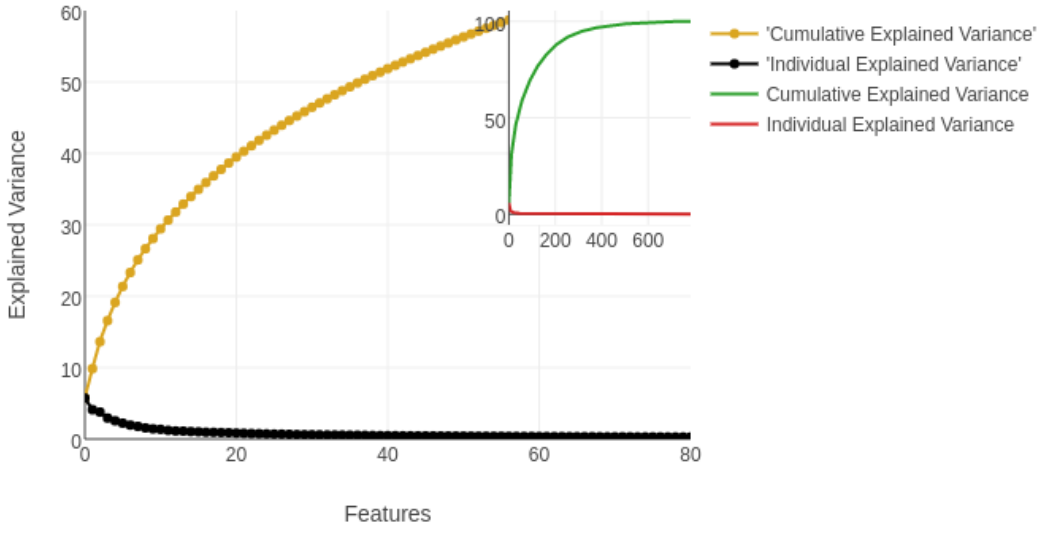


FIGURE 4.5: Explained Variance

two components in new feature space is illustrated in figure 4.8. It clearly identifies clusters in well defined and segregated manner. A good cluster visualisations is prossible because of the topology-preserving attributes of the algorithm.

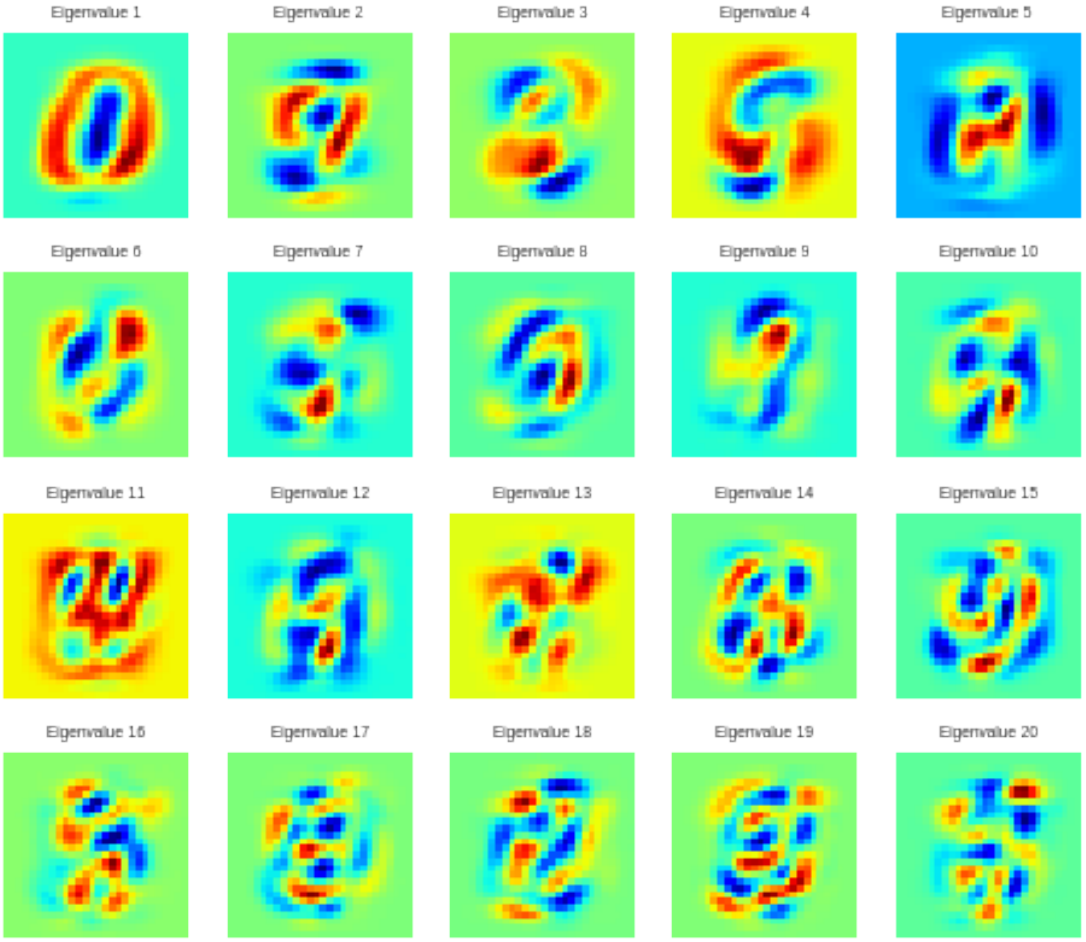


FIGURE 4.6: Top 200 eigenvalues

We now automate the whole process and apply Isomap, Locally Linear Embedding, Spectral Embedding, Local Tangent Space Alignment, Multi-dimensional Scaling, t-distributed Stochastic Neighbor Embedding on the data to see geometric structure embedded in two dimension, represented through figure 4.9. There is problem with the data or data samples we are using, as some methods fails (Local tangent space alignment). Adding to the same, the embedding of the data in lower dimension didn't give much information too.

To see how manifold learning algorithms have performed, we apply out of sample for manifold learning algorithms for arbitrary point, which gives us clear pictures about clusters in the low dimensions. The figure 4.10 clearly indicates that test points represented by colored dots, cross and star are projected on the area concentrated to train points. But the projection of the test points is not so clear. Separation of some numbers are blurred and not separated accurately on the manifold.

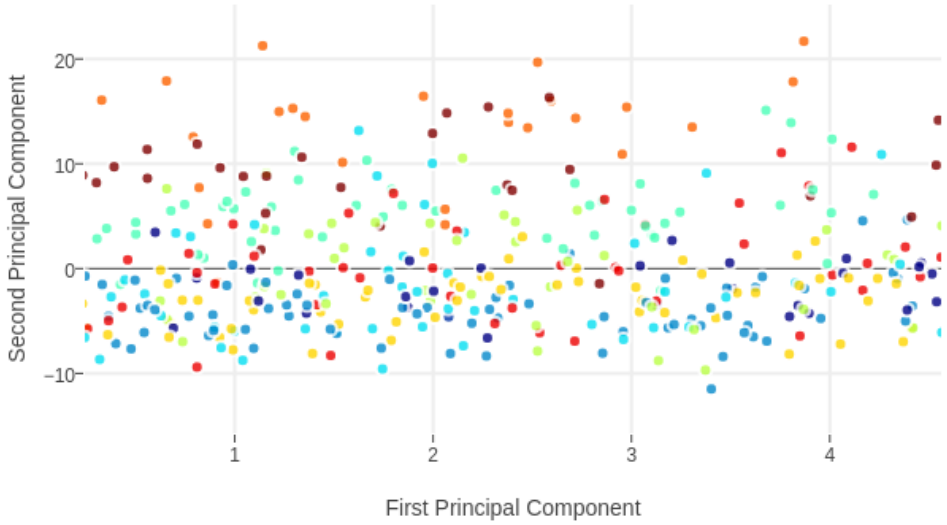


FIGURE 4.7: PCA

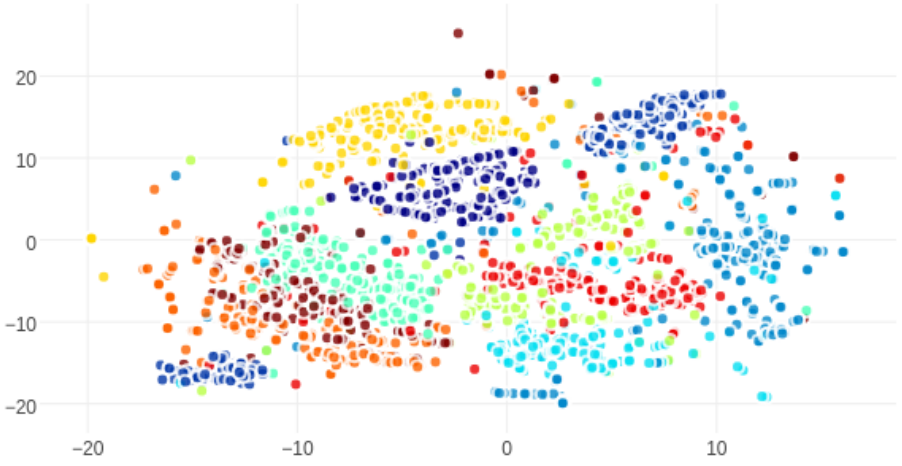


FIGURE 4.8: TSNE (t-Distributed Stochastic Neighbour Embedding)

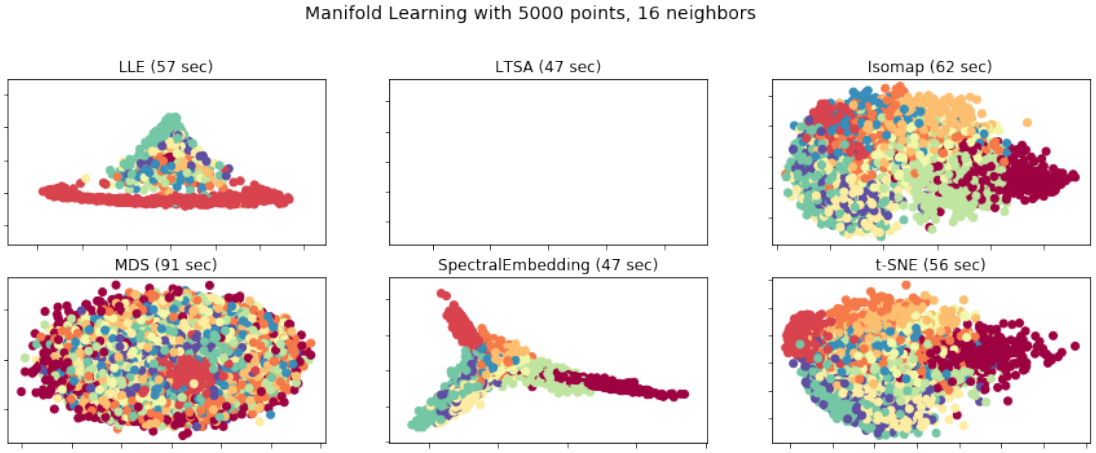
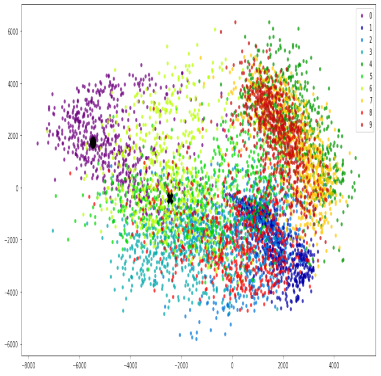
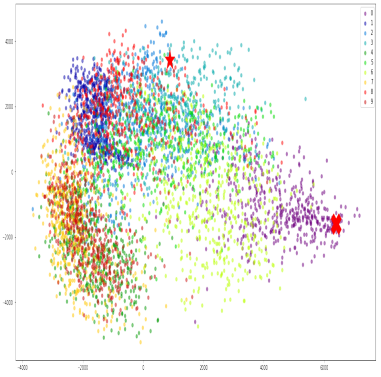


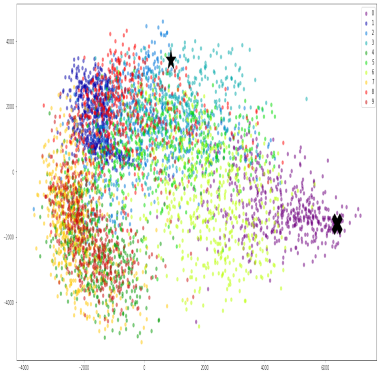
FIGURE 4.9: Manifold Learning Algorithms Output



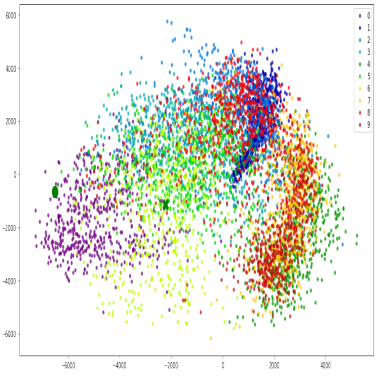
(A) LLE



(B) ISOMAP



(C) SE



(D) TSNE

FIGURE 4.10: Out of sample predication for manifold learning algorithms

4.4.3 Model and Training

As the discussed manifold algorithms didn't perform as per our expectation, we using dimension reduction on the original data. The reduced dimension data

is the trained via simple to complex supervised algorithms. To perform digit recognition task, we seek help of three basic supervised algorithms: K-Nearest Neighbour (KNN) [20] , Support Vector Machine (SVM) [19] and Convolutional Neural Networks (CNN) [18]. All the above module is implemented in python where the packages provides great help.

We also want to compare base model performance with original training data and using reduced dimensions. For example, 200 principle components analysis on our 784 features captures 95% variability in the data. So by PCA, we build the data sets TRAINpca and TESTpca. Similar procedure is followed for manifold learning algorithms too. By extracting n-components from the low dimension, we build train and text datasets corresponding to manifold learning algorithms. For dimension reduction, we take three manifold algorithms namely Isomap, Locally Linear Embedding and t-distributed Stochastic Neighbor Embedding.

The 5-fold cross validation is performed on the training set comprising of original data and data created out of dimension reduction. Then for every model, we choose the parameters yielding highest cross-validation classi cation accuracy. We used the chosen parameters to model each algorithm type. Finally, to obtain a real out of sample performance evaluation, we use the best model to predict on the test set. We also want to clarify that, we didn't optimize our model, its basic version only. What we are interested in accuracy of model with dimension reduction in comparison to original dataset.

4.4.4 Results

Our experiments results in implementing three different types of model for four different data sets. For reducing the computational burden, we took subsample of 15,000 from the original training set of 60,000. All the computation was performed on Dell's 7th Generation Intel Core i3-7100H Processor with 16GB DDR4-2400MHz RAM powered with NVIDIA GeForce GTX 960 graphics card. The model with CNN was performed with tensor-flow backed.

As we wanted to compare our models on four different datasets, we first started with original. The training set accuracy with original datasets is summarized in Table 4.1. We hypothesized that a properly trained CNN model to attain highest accuracy, but SVM performed better than CNN. KNN performed as per the expectation.

TABLE 4.1: Model Summary for **Data**(original)

Model	Data	Accuracy
KNN	original	91.00
SVM	orignal	95.60
CNN	orignal	93.00

Taking the results summarized in table 4.1, we trained our three model on the data, where dimension reduction was performed using manifold learning algorithm, Isomap. The result summarized in table 4.2 failed to our expectation. There was no improvement in accuracy.

TABLE 4.2: Model Summary for **Data**(isomap)

Model	Data	Accuracy (%)
KNN	isomap	92.00
SVM	isomap	95.50
CNN	isomap	93.00

The models with subsample data obtained after reducing dimension by TSNE improved accuracy of SVM by little percentage, but nothing concrete can be said seeing result 4.3.

TABLE 4.3: Model Summary for **Data**(tsne)

Model	Data	Accuracy
KNN	tsne	92.00
SVM	tsne	96.00
CNN	tsne	93.00

The models accuracy was decreased using data from LLE as summarized in table 4.4.

TABLE 4.4: Model Summary for **Data**(lle)

Model	Data	Accuracy
KNN	lle	90.00
SVM	lle	91.20
CNN	lle	91.20

4.5 Conclusions

The out of sample predication for manifold algorithms indicates that test points are projected on the corrects clusters of training datasets. But the projection of the test points is not so clear. Separation of some numbers are blurred and not separated accurately on the manifold. The prepossessing using Hough transform didn't improve the accuracy. Contrary to our exception, the CNN model didn't perform better than SVM. There was no substantial improvement in the accuracy of hand written digit recognition with preprocessed data having dimension reduction.

Chapter 5

Anomaly Detection

Overview

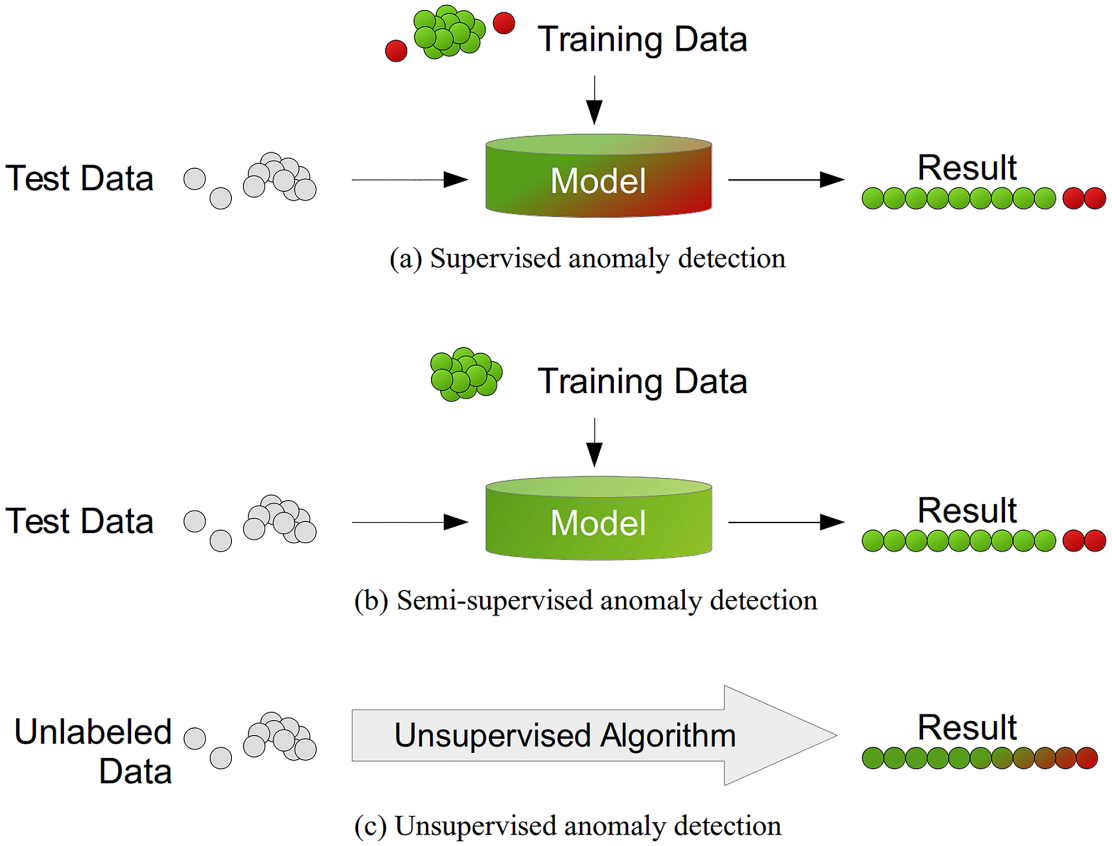
In this chapter, we use clustering property of diffusion maps together with multi-scale approach based on Laplacian pyramid representation for anomaly detection in images. It is expected that diffusion maps clusters the background pixels, and thus in new embedding, the anomaly is far from the clusters. The presented algorithms is tested on simple to complex images to prove its effectiveness or failures.

5.1 Introduction

Anomaly detection refers to the problem of finding patterns in data that do not conform to expected behavior. In computer vision anomaly detection algorithms rely of course on complex image processing methods as a preprocessing step to separate the anomalous points from the background of an image. The large dataset, the presence of noise and high dimensionality of the data makes the anomaly detection in images a challenging task.

The stimulating task of anomaly detection have been studied by many researchers over the past decades. A review paper [12] tries to provide a structured and comprehensive overview of the research on anomaly detection by extracting six approach namely probabilistic, distance-based, domain-based, dimension reduction, reconstruction-based, and information theoretic approach. The dimension reduction and reconstruction based approach is one quite useful in anomaly detection. Such techniques can find a representation which separates the anomaly from the background. Adding to the same, the former being data driven approach doesn't depends on the model, training data and the choice of the distribution. In contrast to the well-known above classification setup, Goldstein and Uchida [14] have classified anomaly detection approaches based on labels available in the

dataset as illustrated in figure 5.1. In our paper, as we have unlabeled data, we use unsupervised algorithms.



(a) Supervised anomaly detection

(b) Semi-supervised anomaly detection

(c) Unsupervised anomaly detection

FIGURE 5.1: Anomaly detection approach based on data labels.

Though, there is girth of approach using manifold learning algorithms, a convincing generic approach for anomaly detection still searching for best fit.

Motivated by the Mishne and Cohen paper [13], we use manifold learning approach, in particular, diffusion maps combining spectral dimensionality reduction and a nearest-neighbor-based anomaly score to detect anomaly in the images. The Laplacian pyramid representation a integral part of our approach which take care of problems associated with interpolative nature of extension methods used by diffusion maps.

5.2 Diffusion Maps

Diffusion map is a manifold learning method that maps high-dimensional data to a low-dimensional diffusion space by constructing the graph Laplacian of the data set.

Considering high dimensional data $\mathbf{X} = \{x_1, x_2, x_3, ..., x_N\} \in \mathbb{R}^{N \times \mathbf{D}}$, we construct a weighted graph. Where, the data points are the nodes and edge weight is similarity measure between the two data points. With suitable weight function, we create random walk by by normalizing the kernel. Now using spectral decomposition of matrix created by random walk, we compute diffusion distance using eigendecomposition. As the spectrum decays really fast, we get reduced dimension embedding. The details mathematical description of diffusion maps is discussed in Chapter 3 (3.4.2.2).

The scale parameters (σ) defined in Gaussian kernels $w(x_i, x_j) = \exp(-\|x_i - x_j\|^2/\sigma^2)$ for constructing the weighted graph is very important. Setting σ to be too small results local neighborhoods of size 1. Large σ connect the anomalies with the cluttered background. In our paper, we use location dependent σ for each data point instead of selecting a single scaling parameter as discussed in [13]. The scale σ is calculated for the neighborhood of point statics x_i, for example, $\sigma_i = \|x_i - x_K\|^2$. Then, the similarity kernels are calculated as $w(x_i, x_j) = \exp(\frac{-\|x_i - x_j\|^2}{\sigma_i \sigma_j})$.

5.3 Function Extension

It is impractical to compute a diffusion map for the large dataset. Instead, a diffusion map is constructed for part of the samples.The embedding is then extended to all points using an out-of-sample extension method. The algorithms for one such out-of-the sample method is done using Laplacian Pyramid [13].

5.4 Laplacian Pyramid

The Laplacian pyramid algorithm constructs a coarse approximation of a function $\mathbf{f}$ for a given scale at each iteration. In the next iteration, we use the difference between $\mathbf{f}$ and the coarse approximation as input, which is approximated as Gaussian kernels at each level with scale getting finer and finer.

We define Gausian Kernels on $\mathbf{X}$ at lowest level as:

$$\mathbf{W}_0^L \overset{\Omega}{=} w_0^L(x_i, x_j) = \exp(-\|x_i - x_j\|^2/\varepsilon_0) \tag{5.1}$$

where $\varepsilon_0 \gg$. Normalizing $\mathbf{W}_0^L$ in equation 5.1, we get smoothing operator $\mathbf{Q}_0^L$

$$\mathbf{Q}_0^L = q_0^L(x_i, x_j) = (q_0^L)^{-1} w_0^L(x_i, x_j) \tag{5.2}$$

where $q_0^L = \sum_j w_0^L(x_i, x_j)$. Similar construction is done for the next level to get $\mathbf{W}_l^L, \mathbf{Q}_l^L$. Using this iteration, we get the Laplacian Pyramid representation $\mathbf{f}$ of $\mathbf{X}$, which is then iterated on finer and finer scales until the difference between

function **f** and approximation is below some error threshold. For mathematical details, please see [13].

5.5 Multiscale Anomaly Detection

The method motivated from base paper[13] overcomes the limitations of random sampling with a multiscale approach. In reality, it is not hard to believe that the anomalies in the image are larger than a single pixel, which is hard to be detected at several resolutions of the image. The situation is much worst at lower resolution. As it is impossible to sample a larger percentage of the lower resolution image, the anomaly detection is bound to fail. Multiscale approach overcome the limitations of random sampling by performing anomaly detection at different resolutions of the image, which gets finer and finer if any anomaly is missed. The Laplacian pyramid representation of the image is constructed starting from coarsest scale giving rise diffusion map using subset of the data or even whole pixel. Anamoly score is used at this level to determine which pixel is anomaly. Using this pixel as input to the next coarser level to detect next anomalous pixel. It is then continued from level to level, with each previous level providing prior information on which samples of the data set are used to construct the diffusion map. In the next section, we present the algorithms in the detail.

5.5.1 Algorithms

1. Input Image: $\mathbf{I}$. Size: 400×400.
2. For $l = 0 : L$ compute Gaussian pyramid $\{\mathbf{G}_i\}_{l=0}^{L}$, where $\mathbf{G}_0$ is the original image and $\mathbf{G}_L$ is the coarsest resolution as discussed in section 5.4.
3. Starting with $\mathbf{G}_L$, sample a subset $\mathbf{I}_s$ from $\mathbf{I}$.
4. Compute Diffusion map using $\mathbf{I}_s$.
5. Extend above step to remaining pixels.
6. Calculate anomaly score $\mathbf{C}_l$ for 400 pixels.
7. Set threshold τ_l to 95th percentile of the anomaly score.
8. If $\mathbf{C}_l > \tau_l$, the anomaly will be sampled more densely. Goto step 2, else
9. output.

We then sample the rest of the pixels randomly. In the anomaly score $\mathbf{C}_{l-1}$ calculated, anomaly gets high score, separating it from the background. The algorithms is illustrated using flowchart 5.2.

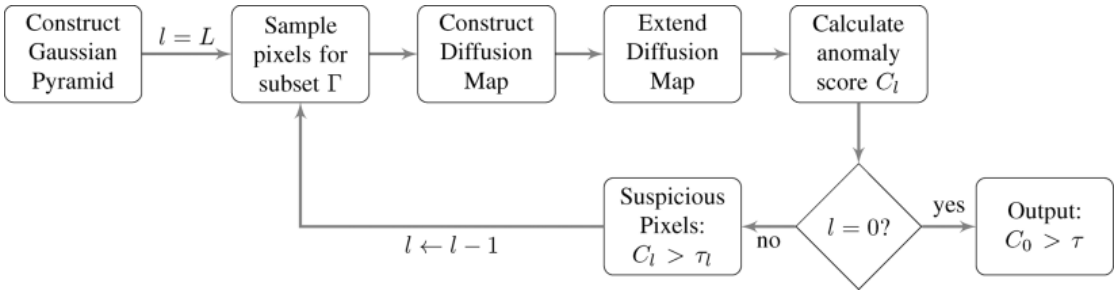


FIGURE 5.2: Multiscale Algorithm.

The affinity matrix at each level is calculated for subset of $\mathbf{X}_l$ using $w(x_i, x_j) = \exp(\frac{-\|x_i - x_j\|^2}{\sigma_i \sigma_j})$ with scaling parameter as explained in section 5.2. The nearest neighbors is used to calculate matrix to reduce computation time and memory requirements. This makes the matrix sparse. The anomaly score for each level is based on a nearest-neighbor approach too.

To determine any pixel as anomaly, we define anomaly score of tested pixels i using below equation.

$$\mathbf{C}_l(i) = 1 - \frac{1}{m} \sum_{j \in N_i} \overline{w}(i, j) \tag{5.3}$$

where m is the number of pixels in N_i.

We compare pixel i to its neighbors $\{j\}$ in the spatial neighborhood denoted by N_i, which is a square window surrounding pixel i of size $2\mathbf{W}+1$ in each dimension. As, we assume that anomaly is larger than a pixel, we mask the inner part of the window surrounding the tested pixel. We use only the pixels in the outer window.

The smoothed estimate of the number of close neighbors i is defines as $\sum_{j \in N} \overline{w}(i, j)$. The notion of closeness is determined by the diffusion distance.

5.6 Data

For our experiments with the proposed algorithms, We take pictures of varied size ranging from 800×800 to 200×200. The pictures are of low resolution to contrast resolution, so that performance of the proposed algorithms can be compared. This also allow us to play with various parameters proposed in the algorithms. The figure 5.3 shows six different input image, which we consider for our experiment.

We treat the red card in subfigure (A) 5.3a, sea-miner in subfigure (B) 5.3b, village girl in subfigure (C) 5.3c, black sheep in subfigure (D) 5.3d, girl in subfigure (E)

(A) (B) (C)

(D) (E) (F)

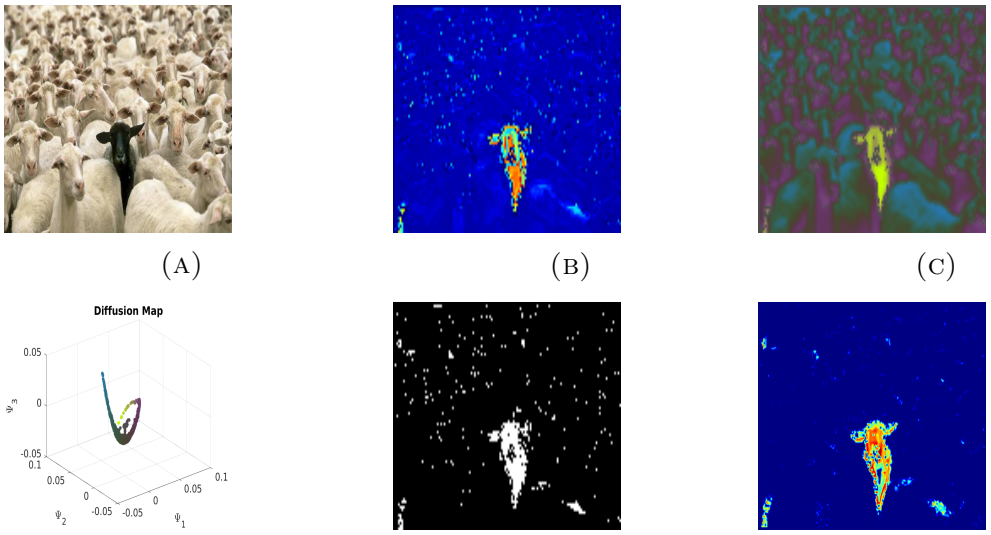
FIGURE 5.3: Input Image

5.3e and duck in subfigure (F) 5.3f as anomalies. Automatic detection of anomalies is a challenging task due to the high variability in the appearance of the anomaly and background images. In the all of the pictures, the anomaly appear as a strong bright region aside a dark region and always smaller than the background region in the image.

5.7 Results

Unlike other anomaly detection algorithms in the images relying on the training set, based on real images and/or synthetic ones, we just use expected size of anomalies as prior information.

The algorithms is evaluated on the all the six images, but one at a time to test the performance and results. As the pixel of images varies from 800×800 to 200×200, we set different parameters for different image. For example, we evaluated our algorithm on a set of 50 ducks in the field 5.3f, each image sized 800×800 pixels. As the size of our images are big compared to image used in the base paper [13], we set Gaussian pyramid level $\mathbf{L} = 5$. Starting from level zero $\mathbf{L} = 0$, we take image size 800×800 pixels for input image, village girl in the field 5.3f, patch size 128×128 pixels, embedding dimension $\mathbf{d} = 24$, percent of pixels in subset as 0.1, window size as 64×64 and mask as 18×18. For efficient computation in all images, we calculate the affinity matrix using exact k-nearest-neighbor search with 16 neighbors for each point, resulting in a sparse weight matrix. For other levels $\mathbf{L} = 1, 2, 3, 4$, we keep decreasing the above parameters by half for each level. For the above figure, the parameters used in the multiscale detector are given in Table 5.5



(A) (B) (C)

(D) (E) (F)

FIGURE 5.4: Anomaly Detection results for multiscale detector

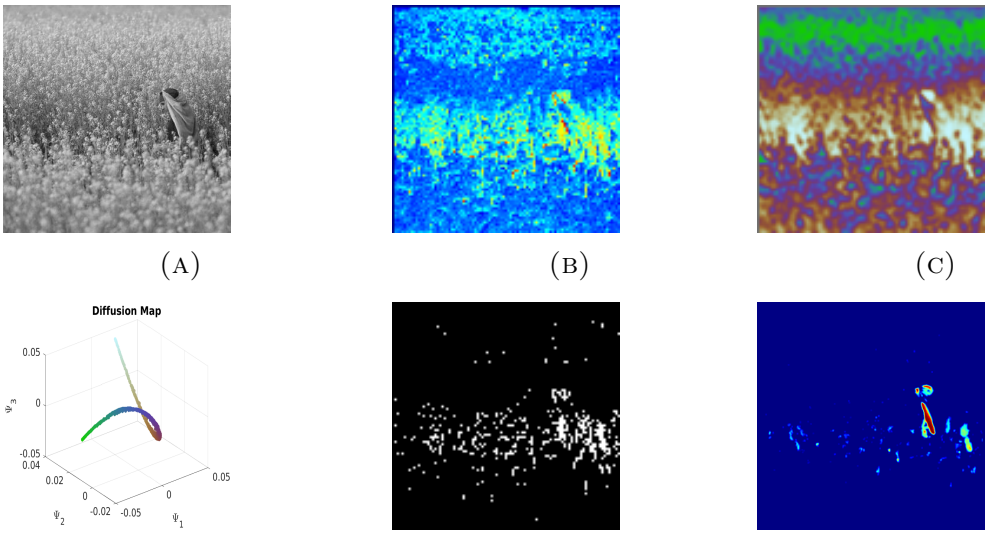
TABLE 5.1: Parameters used in Multiscle algorithms for figure 5.3d (Black Sheep)

L	Image	Patch	Window	Mask	Pixels (%)	**d**
0	200×200	32×32	16×16	5×5	0.10	6
1	100×100	16×16	8×8	3×3	0.20	3
2	50×50	8×8	4×4	3×3	0.30	3

Now, we start with basic anomaly detection for figure 5.3d. In this image, there is black sheep among white sheep. The anomaly detection results for the above input image is illustrated in the figure 5.4. Here, (A) represent the original image. The image detected at level 0 is depicted in (B) and its 3 dimensional coordinates in (C) through Gaussian maps embedding in (D). The (E) represents the suspicious detection leading to anomaly score in (E). The parameters used is summarized in Table 5.1.

With positive results from above naive obvious black sheep image, we progress to other picture 5.3c. In this figure, the village girl is walking in the field. We have to detect her using our algorithms. Figure 5.5 visualizes our algorithms through different stage of algorithms. Corresponding parameters are summarized in Table 5.1.

The image of village girl as anomaly was not contrast, so we take another image of girl standing in the field 5.3e. This image is more contrast and larger in size as compared to village girl image. As the detections are found by thresholding the anomaly score image resulting in a binary image. Correct detection is considered



(A) (B) (C)

(D) (E) (F)

FIGURE 5.5: Anomaly Detection results for multiscale detector

TABLE 5.2: Parameters used in Multiscle algorithms for figure 5.3c (Village Girl)

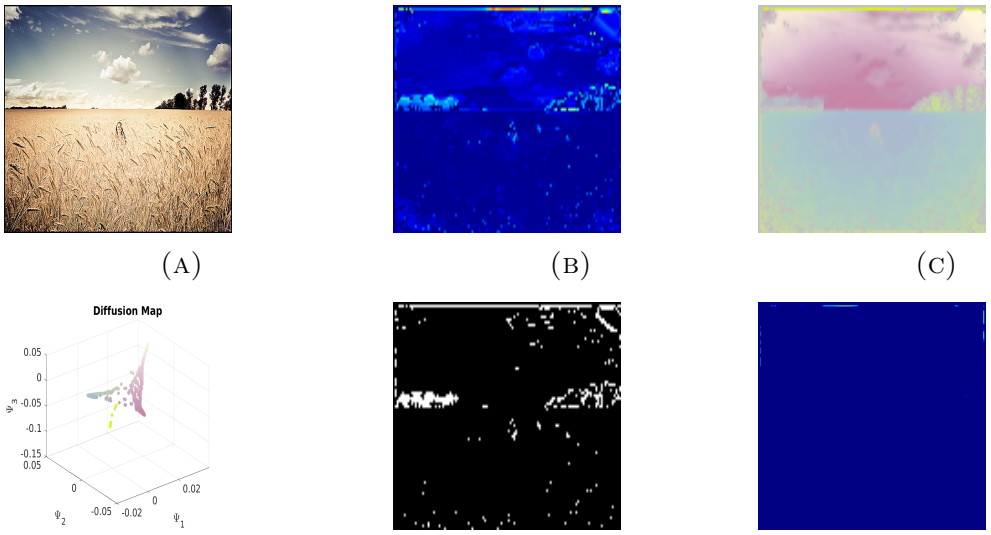
L	Image	Patch	Window	Mask	Pixels (%)	**d**
0	200×200	8×8	40×40	6×6	0.10	6
1	100×100	4×4	20×20	3×3	0.30	3
2	50×50	2×2	10×10	2×2	0.50	3

TABLE 5.3: Parameters used in Multiscle algorithms for figure 5.3e (girl in the field)

L	Image	Patch	Window	Mask	Pixels (%)	**d**
0	400×400	64×64	32×32	8×8	0.10	12
1	200×200	32×32	16×16	4×4	0.20	6
2	100×100	16×16	8×8	2×2	0.40	3
3	50×50	8×8	4×4	2×2	0.50	3

as true positive and incorrect as false alarms. Unlike above two example, the proposed algorithms gives false alarms with the image where girls standing in the field is anomaly. Figure 5.6 discuss the result for the image described. Table 5.3 summarize the parameters.

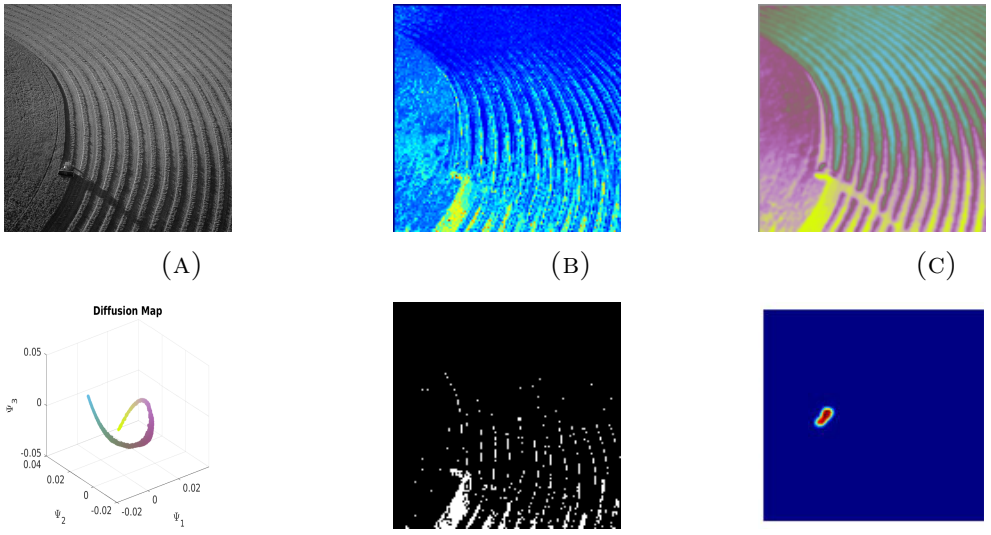
To replicate the results of the base paper[13] for detection of sea-mines in side-scan sonar, we choose parameters as described in the Table 5.4. Figure 5.7 describe



(A) (B) (C)

(D) (E) (F)

FIGURE 5.6: Anomaly Detection results for multiscale detector



(A) (B) (C)

(D) (E) (F)

FIGURE 5.7: Anomaly Detection results for multiscale detector

true positive results of proposed algorithm.

Our algorithm fails to give any results using figure 5.3f. This figure is of size 800×800 with very contrast background with multiple ducks which is hard to see. We have used the parameter as described in Table 5.5.

TABLE 5.4: Parameters used in Multiscle algorithms for figure 5.3a (sea-mines)

L	Image	Patch	Window	Mask	Pixels (%)	**d**
0	200×200	8×8	41×41	9×9	0.10	6
1	100×100	4×4	21×21	5×5	0.33	3
2	50×50	2×2	13×13	5×5	0.50	3

TABLE 5.5: Parameters used in Multiscle algorithms for figure 5.3f

L	Image	Patch	Window	Mask	Pixels (%)	**d**
0	800×800	128×128	64×64	18×18	0.10	24
1	400×400	64×64	32×32	9×9	0.20	12
2	200×200	32×32	16×16	5×5	0.30	6
3	100×100	16×16	8×8	3×3	0.40	3
4	50×50	8×8	4×4	3×3	0.50	3

5.8 Conclusions

The proposed algorithms described in this chapter performs well the images which is not too contrast, but gives false positive on image which is of high resolutions. A future extension of this work would be combine anomaly scores from the different multiscale levels into a single anomaly score. This will enable detecting anomalies of different size in an image, which this algorithms fails to address.

Chapter 6

Conclusion

6.1 Conclusions

1. Deviating from primary evaluation methodology of manifold leaning algorithms based on artificial data sets, we use MNIST datasets for optical character recognition.

2. The out of sample prediction for manifold algorithms indicates that test points are projected on the corrects clusters of training datasets, but separation of some numbers were blurred.

3. The preprocessing of MNIST data by manifold learning algorithms didn't improve accuracy in optical character recognition.

4. For anomaly detection, the proposed algorithms performs well on the images which is not too contrast, but gives false positive on image which is of high resolutions.

6.2 Future Work

1. When answering the questions on evaluating the results of manifold learning, we need to take in account of manifold structure and noises.

2. When there is multiple sub-manifolds of possibly different dimensionalities embedded in high dimensional complex data, it is most unlikely that the existing manifold learning approaches can discover all the interesting lower-dimensional structures. There is need to develop a hierarchical manifolds learning framework to discover a variety of the underlying low dimensional structures.

3. In optical character recognition, there is need to optimize the model when calculating accuracy.

4. A future extension of this work would be combine anomaly scores from the different multiscale levels into a single anomaly score. This will enable detecting anomalies of different size in an image, which this algorithms fails to address.

Appendix A

Manifold learning algorithms for artificial data

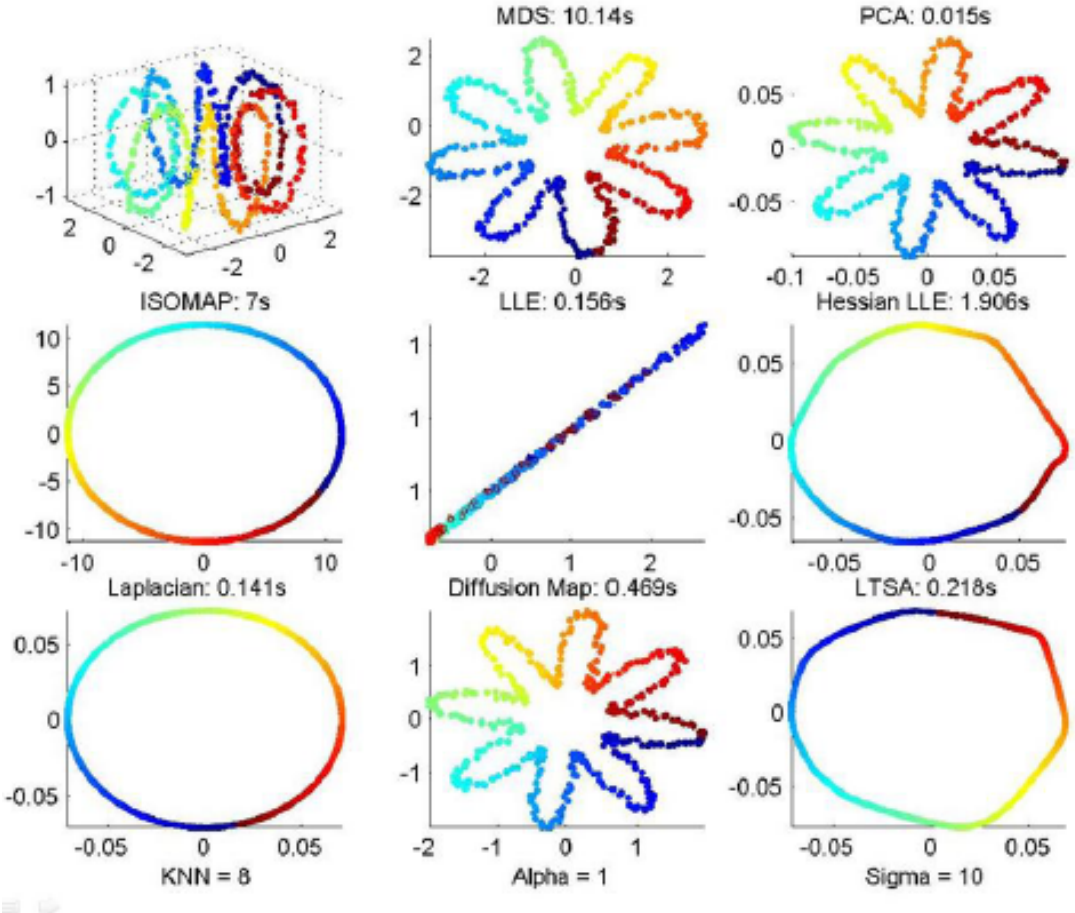


FIGURE A.1: Manifold learning algorithms comparison using Swiss Roll. MDS and PCA fails to unroll Swiss Roll; LLE and Laplacian fails to perform too; Diffusion maps couldn't unroll Swiss Roll.

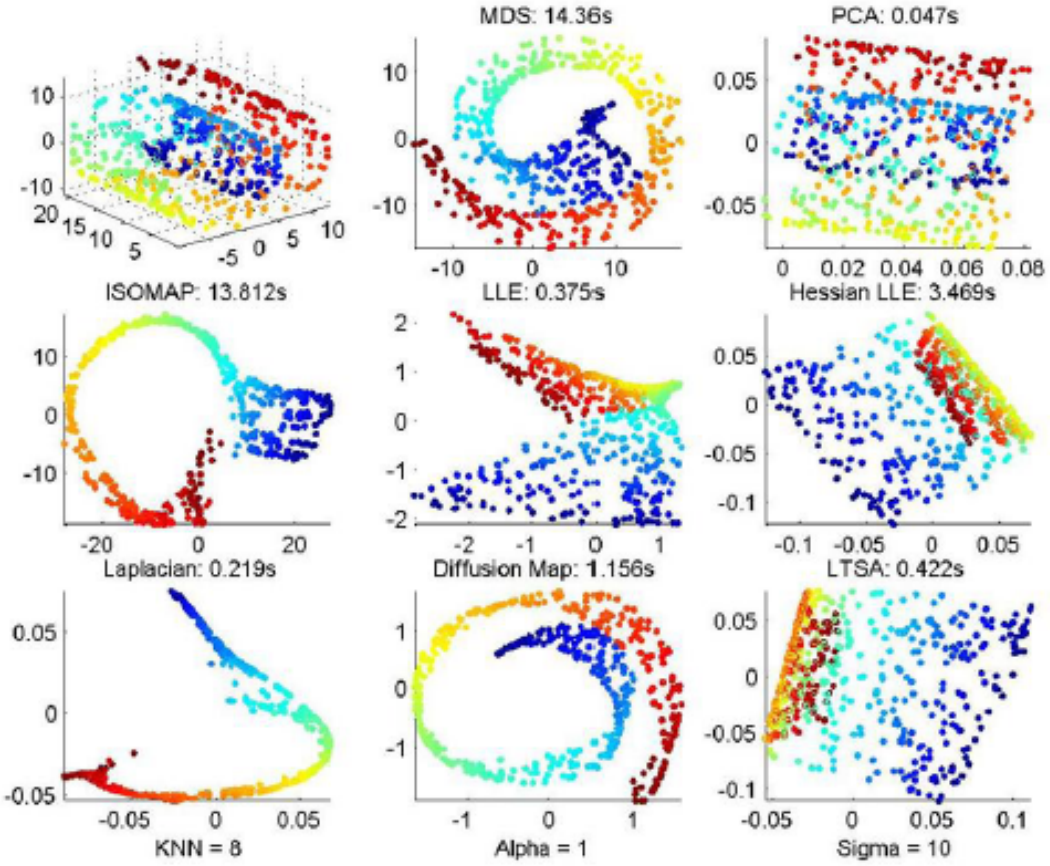


FIGURE A.2: Manifold learning algorithms comparison using Toroidal Helix. ISOMAP, LLE, Laplacian, and Diffusion Map unraveled Toroidal Helix into a circle, which is correct. PCA and MDS fails to perform.

Bibliography

[1] Thorstensen, Nicolas. Manifold learning and applications to shape and image processing. *PhD Thesis, Ecole des Ponts ParisTech*. https://pastel.archives-ouvertes.fr/pastel-00005860/file/Thesis.pdf, 2009.

[2] Talwalkar, A and Kumar, S and Rowley, H. Large-scale manifold learning. *2008 IEEE Conference on Computer Vision and Pattern Recognition*. 1-8, June, 2008.

[3] Etyngier, Patrick . Statistical learning, Shape Manifolds and Applications to Image Segmentation. *PhD thesis, Ecole Nationale des Ponts et Chausees*. http://imagine.enpc.fr/publications/papers/PhD08.pdf, 9, 41, 135, 2008.

[4] Belkin, Mikhail and Partha Niyogi. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. *Advances in Neural Information Processing Systems 14*. http://papers.nips.cc/paper MIT Press, 585–591, 2002.

[5] Boothby, W.M. : An Introduction to Differential Manifolds and Riemannian Geometry. . *Academic Press*, London, 2003.

[6] Tenenbaum, J, De Silva, V and Langford, J. A global geometric framework for nonlinear dimension reduction. *Science*, vol. 290, 2000.

[7] Roweis ST and Saul LK. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[8] Cox, Trevor and Cox, Michael. Multidimensional Scaling. *Second Edition. Chapman & Hall/CRC*, September 2000.

[9] Lindenbaum, Ofir and Yeredor, Arie and Salhov, Moshe and Averbuch, Amir. MultiView Diffusion Maps. *CoRR*. https://arxiv.org/pdf/1508.05550.pdf, 2015.

[10] Camastra, Francesco and Staiano, Antonino. Intrinsic dimension estimation: Advances and open problems. *Inf. Sci.*,328,C, 26-41, 2016.

[11] Levina, Elizaveta and Bickel, Peter. Maximum Likelihood Estimation of Intrinsic Dimension. *Advances in Neural Information Processing Systems 17*,777–784, MIT Press, 2005.

[12] Pimentel, Marco, Clifton, David, Clifton, Lei and Tarassenko, Lionel. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[13] Mishne, Gal and Cohen, Israel. Multiscale Anomaly Detection Using Diffusion Maps. *IEEE Journal of Selected Topics in Signal Processing*, Vol. 7, No. 1, 2013.

[14] Goldstein, Markus and Uchida, Seiichi. A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data. *PLOS ONE*, DOI:10.1371/journal.pone.0152173, 2016.

[15] LeCun Y, Bottou L, Bengio Y and Haffner P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[16] Leandro, Fernandes and Manuel, Oliveira. Real-time line detection through an improved Hough transform voting scheme. *Pattern Recognition*, Volume 41, Issue 9, September 2008.

[17] van der Maaten, L and Hinton, GE. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research* , Volume 9, 2579-2605, 2008.

[18] Ahranjany, S.S, Razzazi, F and Ghassemian, G.H. A very high accuracy handwritten character recognition system for Farsi/Arabic digits using Convolutional Neural Networks. *2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*, pp. 1585-1592, Changsha, 2010.

[19] Steinwart, Ingo and Christmann, Andreas. Support Vector Machines. *Springer Publishing Company, Incorporated* , 2008.

[20] Babu, U.R, Venkateswarlu, Y and Chintha, A.K. Handwritten Digit Recognition Using K-Nearest Neighbour Classifier. *2014 World Congress on Computing and Communication Technologies*, pp. 60-65, Trichirappalli, 2014.