Malik Z, Hussain A & Wu J (2016) An online generalized eigenvalue version of Laplacian Eigenmaps for visual big data, *Neurocomputing*, 173 (Part 2), pp. 127-136.

DOI: 10.1016/j.neucom.2014.12.119

An Online Generalized Eigenvalue Version of Laplacian Eigenmap for Visual Big Data

Zeeshan Khawar Malik, Amir Hussain and Jonathan Wu[1]

University of Stirling, UK and University of Windsor, Ontario, Canada[1]
Email: zkm,ahu@cs.stir.ac.uk and jwu@uwindsor.ca[1]

Abstract

This paper presents a novel online version of laplacian eigenmap termed as generalized incremental laplacian eigenmap (GENILE), one of the most popular manifold-based dimensionality reduction technique performed by solving the generalized eigenvalue problem. We have used swiss roll and s-curve dataset, the most popular datasets used for manifold-based learning techniques, in this paper as artificial datasets. For a real data experiment, we have selected the MNIST digit dataset, the bank-note dataset, and the heart disease dataset for testing and evaluating our novel method and for comparing it with the standard batch isomap method and other manifold-based learning techniques. The experimental results have clearly shown the improvements in terms of classification accuracy by the proposed method in comparison with other techniques.

Keywords: dimensionality reduction, generalized eigenvalue problem, laplacian eigenmap, manifold-based learning

1. Introduction

Most of the traditional techniques used for feature extraction and dimensionality reduction come in both batch and incremental versions. Out of all of the dimensionality reduction methods proposed in the past, manifold-based learning techniques for feature extraction and dimensionality reduction have gained great popularity but most of these run in batch mode. Very few incremental approaches based on manifold-based learning have been proposed in the past, which is not good because the major difficulty arises in scenarios where the input is coming in more than one chunk from time to time, batch

mode algorithms repetitively recalculate the previous chunks again and again at each new input, which becomes computationally very expensive and less efficient.

The general problem of classical dimensionality reduction methods like PCA, MDS, etc., is to produce projections for non-linear data. This has to some extent been solved by methods that generate non-linear maps such as self-organizing maps and other neural network based methods [1]. These methods normally resolve a non-linear optimization problem by using gradient descent methods that do not always reach the global optimum and most of the time produces a local optimum. The ideal solution for this is to explicitly consider the structure of the manifold on which the data reside.

A classical and widely used dimensionality reduction method is principal component analysis (PCA), used primarily for visualization and preprocessing purposes in the area of data mining and machine learning [2], information retrieval [3], multimedia [4], etc. Dimensionality reduction using PCA is performed on the basis of the leading eigenvectors of the data's covariance matrix. One more classical linear method is multi-dimensional scaling, which can only see the flat euclidean structure and can unable to find the non-linear structure in the data [5]. In [6] the authors have proposed a variant of fisher's linear discriminant analysis technique - another classical dimensionality reduction technique which resulted in more than one filters for the two class problem and also produced higher classification accuracy in comparison with other dimensionality reduction techniques. Similarly in [7] the authors have proposed a regularized version of linear discriminant analysis by introducing within cluster scatter matrix generated on the basis of between and within class scatter matrix. In [8] the authors have proposed an L1-norm based multi-linear subspace analysis technique which has proved very robust to outliers as it suppresses the negative effect of outliers on the resulting projection matrix or projection vector. This method is suitable in scenarios where less training data is available. Most of these classical dimensionality reduction methods suffer from difficulties in designing the cost function and are often limited to relatively low-dimensional data sets. Recently, manifold based dimensionality reduction methods have been developed such as isomap [9, 10], local linear embedding (LLE) [11], laplacian eigenmaps [12, 13], hessian eigenmap [14], semi-definite programming (SDE) [15], manifold based charting [16], local tangent space alignment (LTSA) [17] and diffusion maps [18]. Due to these methods' non-linear geometrical nature, they have attracted wide attention in various domains of computational

intelligence.

All the above mentioned algorithms run in batch mode and necessarily require all the data to be available at once for processing. These types of algorithms are computationally very expensive in scenarios where the data is required to be observed sequentially. The reason is the need for repetitively reconsidering the previously processed data: this makes the execution of these batch algorithms computationally very expensive. There are several scenarios for explaining the benefits of incremental learning and how it overcomes the problem of the high computational cost involved in the execution of a batch version. Instead of considering a single new entry, we have considered the most common scenario, where the data is coming in more than one chunk, and in a sequence. The problem of incremental learning of data coming in this way can be stated as follows. Assume $\mathbf{X} = [x_1, x_2, ..., x_{t_1}, x_{t_1+1}, x_{t_1+2}, ..., x_{t_1+t_2}]$ as a sum of two chunks of the whole dataset, where $x_i \epsilon R^{t_1+t_2}$. Suppose the low-dimensional coordinates y_i of x_i representing the first two chunks termed as $t_1 + t_2$ training samples have already been produced. When the third chunk, say $[x_{t_1+t_2+1}, x_{t_1+t_2+2}, ...x_{t_1+t_2+t_3}]$, comes, incremental learning should independently figure out how to project this chunk of information onto the low dimensional space.

In many scenarios, it can be uncommon for all the data to be present before learning, for example social networking site data, online web transaction data, and data received through sensors, due to which, the storage mechanism has also completely changed. These kinds of data are mostly collected and stored in raw form in a distributed file structure storage environment like Hadoop or Cassandra. Analytical programming environments like java, matlab and revolution R extract data from these environment containing storage ranging from terabytes to petabytes and perform learning on these big datasets. The incremental learning technique is best suited to these scenarios because the huge amount of transactional data cannot be learned at once. Instead the best choice is to learn the data in the form of chunks, or more appropriately, one data point at a time in a completely adaptive environment.

Several incremental manifold-based learning methods related to different dimensionality reduction techniques that have been proposed in the literature. In [19], an incremental manifold-based learning method via tangent space alignment is presented. This method works by firstly updating the existing local geometrical information in view of the new input. After that using it with respect to the existing points to perform new estimations and

updating of the whole tangent space. In [20] the authors have proposed an incremental learning multiple threshold based classifier that performs sample by sample incremental learning and then update a pre-defined number of threshold based classifiers without re-training on previous data. Another important characteristic of this algorithm is the optimal determination of threshold and training error of each classifier in a completely close form. In [21], Martin and Anil presented an incremental version of isomap, in which the geodesic is updated every time when a new input comes which leads to the solution of an incremental eigen-decomposition problem. In [22], the author has not changed the cost matrix on a new arrival, as the least eigenvalue is taken for projection. This is always susceptible to change, and then an incremental learning problem of local linear embedding (LLE) is processed by solving a $d_2 \times d_2$ minimization problem considering d_2 as the dimensionality of the low-dimensional embedding space. The author in [23] has presented a generalized common framework for local linear embedding (LLE), multidimensional scaling (MDS), isomap and laplacian eigenmap by proposing a novel nystrom formula for new datapoints. This helps in solving the subset eigendecomposition problem and tries to generalize the dimensionality reduction results for the novel datapoints. Similarly, in [24], a general incremental learning framework capable of dealing with one or more new samples each time for the so-called spectral embedding methods is presented. The authors in that paper have solved the dimensionality reduction problem by recovering the latent manifold as new samples on the basis of the low-dimensional embedding coordinates learned from the previous samples.

The incremental methods presented so far in the literature can be easily divided into two groups.

1. Independent Training: Calculate the low-dimensional embedding of a new chunk from a new class or an existing class like incremental subspace versions of PCA and LDA methods [25][26].
2. Dependent Training: Calculate the low-dimensional embedding of a new chunk by using the existing adjacent information of the previous chunk, like the most recently proposed incremental version of laplacian eigenmap, which is very much dependent on the previously processed information to compute the new information. [27].

The rationale behind deriving a new online version in the presence of an existing extension of laplacian eigenmap already proposed in [27] is to highlight four key findings.

1. For big data computations, the two positive semi-definite matrices produced for learning by solving the generalized eigenvalue problem will be quite big in size and require a large amount of computations, which can only be solved by incrementally learning each vector point by point for both matrices.
2. If the data is online in nature and a light-weight adaptable learning mechanism is required, then too our online version will be a better choice than the standard laplacian eigenmap approach.
3. It is not always important as mentioned in [12, 27] to consider only minimum eigenvalues to produce low-dimensional projections for laplacian eigenmap.
4. The low dimensional embedding can also be calculated incrementally very easily independently in one pass without using the existing adjacent information of the previous chunk as in [27].

2. Manifold-Based Learning

Let us consider the problem of observing some images: there are factors like the view angle, rotation and the lighting angle of the pixel intensities, which means that the data in the high-dimensional space attains a complex non-linear structure. These changes do not occur abruptly and so the data can be reasonably assumed to lie approximately on a (Riemannian) manifold. This is one reason for manifold-based learning techniques' gaining a lot of attention. In this paper, our topic of discussion is to propose an online version of laplacian eigenmap, which is a manifold-based learning technique performed by first building a graph by incorporating the neighborhood information of the dataset. After that using the notion of the laplacian of the graph, computing a low dimensional representation of the dataset that optimally preserves local neighborhood information in a certain sense.

The core idea of laplacian eigenmap [12] is very simple: with minimal computation calculate locally one sparse eigenvalue problem. Since the first part has minimal computation and is calculated only once, the idea for creating an incremental version is to solve the sparse eigenvalue problem incrementally therefore making it computationally more efficient and applicable for deep analysis.

The remainder of this paper is organized as follows: Section 3 reviews the standard laplacian eigenmap technique and in Section 4 we discuss the

Table 1: Important Notations

Notations	Description
X	Input Data
n	Number of training datapoints
t_1, t_2 and t_3	chunks of data from the whole dataset
W	The adjacent weight matrix
D	A diagonal weight matrix, $D_{ij} = \sum_j W_{ji}$
L	Laplacian Matrix
$\mathbf{w}_1$	Eigenvector corresponding to the maximum eigenvalue
$\mathbf{w}_2$	Eigenvector corresponding to the second highest eigenvalue
$\mathbf{y}_1$	Output projection 1
$\mathbf{y}_2$	Output projection 2

existing incremental method for finding the maximum eigenvalue of the generalized eigenproblem. In Section 5 we discuss our own incremental version of the standard laplacian eigenmap technique. In section 6, we describe and implement these methods on artificial and real datasets.

For convenience, we show the important notations used in this paper in Table 1.

3. The Laplacian Eigenmap Algorithm

Given l points $x_1, x_2, ..., x_l$ in $\Re^l$, we construct a weighted graph one for each point connected by the set of edges between neighboring points. The steps involved in the execution of a laplacian eigenmap are stated below.

1. Step 1. [Construct an Adjacency Graph Matrix] Using the K-Nearest Neighbor algorithm on the complete dataset, create an edge between x_i and x_j if i is among the n nearest neighbor of j or j is among the n nearest neighbors of i.
2. Step 2. [Weighting the edges] There are two different variations for weighting the edges.
   (a) Heat Kernel. [$t \in \Re$] if node i is connected with j put

$$W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}. \tag{1}$$

(b) Simple Approach. Set $W_{ij} = 1$ if vertices i and j are connected by an edge and set $W_{ij} = 0$ if vertices i and j are not connected by an edge.

3. Step 3. Construct the objective function. Consider the problem of mapping the weighted graph $\mathbf{G}$ to a lower dimensional space so that the connected points stay as close together as possible. Consider $\mathbf{y} = (y_1, y_2, ..., y_n)$ be such a map. A reasonable criteria of choosing an appropriate map is to minimize the following objective function:

$$\sum_{ij}(y_i - y_j)^2 W_{ij}, \tag{2}$$

The minimization of the objective function is an attempt to remain away from the heavy penalty which can occur if the neighboring points x_i and x_j are mapped far apart. Let $\mathbf{D}$ be a diagonal weight matrix, whose entries are (column or rows as W is a symmetric matrix) sums of $\mathbf{W}$. $D_{ii} = \sum_j W_{ji}$ and the laplacian matrix is $\mathbf{L} = \mathbf{D} - \mathbf{W}$. It turns out that for any y, we can have

$$\frac{1}{2}\sum_{ij}(y_i - y_j)^2 W_{ij} = tr(y^T \mathbf{L}\mathbf{y}). \tag{3}$$

The minimization problem can now be elaborated as arg min tr$(y^T \mathbf{L}\mathbf{y})$ such that

$$y^T \mathbf{D}\mathbf{y} = \mathbf{1}, \tag{4}$$

$$y^T \mathbf{D}\mathbf{1} = \mathbf{0}, \tag{5}$$

The bigger the D_{ii} is, the more important y_i will be. There is a constraint as $y^T \mathbf{D}\mathbf{y} = \mathbf{1}$ where constraint $y^T \mathbf{D}\mathbf{1} = \mathbf{0}$ is to eliminate the trivial solution which collapses all vertices of G onto the real number 1.

4. Step 4. Compute the eigenvalues and eigenvectors by solving the generalized eigenvalue problem

$$\mathbf{L}\mathbf{f} = \lambda \mathbf{D}\mathbf{f}, \tag{6}$$

where $\mathbf{D}$ is a diagonal weight matrix whose entries are the sum of each column of $\mathbf{W}$, i.e., $D_{ii} = \sum_j W_{ij}$, and $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is a laplacian matrix which is always symmetric and positive semi-definite.

4. Generalized Eigenproblems: Incremental Solutions

[28] shows that one method of finding the maximum eigenvalue of the generalized eigenproblem

$$\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}, \tag{7}$$

is to iteratively use

$$\begin{aligned} \Delta\mathbf{w} &= \mathbf{A}\mathbf{w} - f(\mathbf{w})\mathbf{B}\mathbf{w}, \\ \mathbf{w} &= \mathbf{w} + \eta\Delta\mathbf{w}, \end{aligned} \tag{8}$$

where η is a learning rate or step size. In (8) the first term on the right-hand side can be considered as a standard hebbian rule term, and the second term acts to bound the length of the vector $\mathbf{w}$. In (8) $f(\mathbf{w}) = \mathbf{w}^t\mathbf{w}$ becomes the continuous version of oja's algorithm as mentioned by Zhang in [28]. The function $f(\mathbf{w}) : R^n - \{0\} \rightarrow R$ satisfies

1. $f(\mathbf{w})$ is locally Lipschitz continuous
2. $\exists M_1 > M_2 > 0 : f(\mathbf{w}) > \lambda_1, \forall \mathbf{w} : \| \mathbf{w} \| \geq M_1$ and $f(\mathbf{w}) < \lambda_n, \forall \mathbf{w} : 0 < \| \mathbf{w} \| \leq M_2$
3. $\forall \mathbf{w} \in R^n - \{0\}, \exists N_1 > N_2 > 0 : f(\theta\mathbf{w}) > \lambda_1, \forall \theta : \theta \geq N_1$ and $f(\theta\mathbf{w}) < \lambda_n, \forall \theta : 0 \leq \theta \leq N_2$ and $f(\theta\mathbf{w})$ is a strictly monotonically increasing function of θ in $[N_1, N_2]$.

where λ_1 is the greatest generalized eigenvalue and λ_n is the least eigenvalue.

Intuitively, what these criteria mean is that:

1. The function is rather smooth.
2. It is always possible to find values of $\mathbf{w}_i, i = 1, 2$ large enough so that the functions of the weights exceed the greatest eigenvalue.
3. It is always possible to find values of $\mathbf{w}_i, i = 1, 2$ small enough so that the functions of the weights are smaller than the least eigenvalue.
4. For any particular value of $\mathbf{w}_i, i = 1, 2$, it is possible to multiply $\mathbf{w}_i, i = 1, 2$ by a scalar and apply the function to the result to get a value greater than the greatest eigenvalue.
5. Similarly, we can find another scalar so that, multiplying the $\mathbf{w}_i, i = 1, 2$, by this scalar and taking the function of the result gives us a value less than the smallest eigenvalue.
6. The function of this product is monotonically increasing between the scalars defined in 4 and 5.

This method was already used by us in [29] to extract slowly varying features from temporal data. In this paper we will further use this method to derive a new incremental learning method for the laplacian eigenmap.

5. Generalized Incremental Laplacian Eigenmap

In this section, we provide our state of the art purely incremental version of the manifold-based learning technique discussed in [12] for laplacian eigenmap.

Let $\mathbf{L}$ be the laplacian matrix and $\mathbf{D}$ be the diagonal matrix where each value of $\mathbf{D}$ is the sum of each column of $\mathbf{W}$ as explained in [12]. As shown in [28], the optimal weights for a linear projection can be found as the solution of the generalized eigenproblems

$$\mathbf{L}\mathbf{w} = \lambda \mathbf{D}\mathbf{w}. \tag{9}$$

Therefore we can use the method of the previous section to get

$$\begin{aligned} \Delta\mathbf{w} &= \mathbf{L}\mathbf{w} - f(\mathbf{w})\mathbf{D}\mathbf{w}, \\ \mathbf{w} &= \mathbf{w} + \eta\Delta\mathbf{w}, \end{aligned} \tag{10}$$

where $\mathbf{L}$ and $\mathbf{D}$ are both symmetric and semi-definite matrices. Both the matrices are calculated prior to the learning process and then by using the generalized eigenvector solution, the filter $\mathbf{w}$ in (10) finds the eigenvector corresponding to the maximum eigenvalue. The interesting thing to note here is the selection of eigenvector corresponding to the maximum eigenvalue instead of considering the minimum eigenvalue as in [12, 27] which actually looses a lot of variance of the data and the actual overall orientation of data lying on the high dimensional space. Considering smaller variance or least eigenvalues indicates data to be close to the mean but will make no improvement in projecting the neighboring points closer to each other. In other words transforming the data to a different direction and attaining the maximum variance of the data by processing each data point incrementally can produce better results as shown in this paper. Therefore in our case due to our algorithm's incremental nature which learns data point by point of every chunk, ~~so~~ by projecting it on the eigenvector corresponding to the maximum eigenvalue produces better result as compared to the standard version of laplacian eigenmap.

It can be seen from (9) that it uses a mixture of batch and online methods since the whole data is used at any one time and the weights are updated incrementally. For a truly neural solution, by updating the weights in an online mode *and using only one sample at a time*, we can go further and replace the $\mathbf{A}$ and $\mathbf{B}$ matrices with instantaneous values so that

$$\mathbf{L}_i\mathbf{w} = \mathbf{D}_i\mathbf{w}, \tag{11}$$

where $i = 1, 2, 3, ..., n$. Here L_i and D_i means the laplacian matrix $\mathbf{L}$ which is computed as $\mathbf{L} = \mathbf{D} - \mathbf{W}$ and the diagonal matrix $\mathbf{D}$ whose entries are sum of each column of $\mathbf{W}$, i.e., $D_{ii} = \sum_j W_{ij}$ will be learned point by point in a purely incremental manner. In order to find the next filter corresponding to the second highest eigenvalue, we subsequently deflate the matrices $\mathbf{L}$ and $\mathbf{D}$ and again incrementally solve the generalized eigenvector problem to find a second filter:

$$\mathbf{L}^* = \mathbf{L} - \lambda\mathbf{w}\mathbf{L}\mathbf{w}^T, \tag{12}$$

$$\mathbf{D}^* = \mathbf{D} - \lambda\mathbf{w}\mathbf{D}\mathbf{w}^T, \tag{13}$$

with $\mathbf{L}$ and $\mathbf{D}$ as positive semi-definite matrices and $\mathbf{w}$ is the filter corresponding to the highest eigenvalue. In order to find the next filter corresponding to the second highest eigenvalue $\mathbf{L}$ and $\mathbf{D}$ matrices needs to be deflated first by using (12) and (13).

According to (10), the same learning process is conducted using the deflated $\mathbf{L}$ and $\mathbf{D}$ and the next filter corresponding to the second highest eigenvalue is found. The major steps involved in the execution of the incremental algorithm for the next chunk are shown in Table 2.

6. Simulations

6.1. Experiment on an Artificial Dataset

We have used the swiss roll as an artificial dataset for our initial experiment. It consists of 20,000 datapoints and each data point is three in dimensions. Since our method is purely incremental, we divide the data into four different chunks and perform dimensionality reduction on each chunk separately by using the same learned filters $\mathbf{w_1}$ and $\mathbf{w_2}$ of the previous chunk for the next chunk coming ahead. The learning rate was set to 0.00001 and the number of iterations for learning each chunk were 10,000. The learning

Table 2: The Computing Procedure of the generalized incremental laplacian eigenmap (GENILE) Algorithm

Input: The input patterns $\mathbf{X} = [x_1, x_2, x_3, ..., x_{t_2}]$ where $t_2 \, \epsilon$ Next Chunk

Output: The mapping function: $f : R_i^{\mathbf{t_2}} -> R_o^{\mathbf{t_1}+\mathbf{t_2}}$

Step 1:[Construct an adjacency Graph Matrix of the new chunk t_2]
Using the K-Nearest Neighbor Algorithm on the whole chunk t_2 and create an edge between x_i and x_j if x_i is among the K nearest neighbor of x_j or x_j is among the K nearest neighbor of x_i of the chunk t_2.

Step 2::[Weighting the edges independently of the chunk t_2]

Heat Kernel. $[t \epsilon t_2]$ if node i is connected with j put
$W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$,

Simple Approach. Set $W_{ij} = 1$ if vertices i and j are connected by an edge and set $W_{ij} = 0$ if vertices i and j are not connected by an edge.

Step 3:Construct an objective function of the new chunk independently. Consider $\mathbf{y} = [y_1, y_2, y_3, ..., y_{t_2}]$. The criteria to minimize would be similar to the existing method but this time the minimization will be performed independently for each chunk:

$\sum_{ij} (y_{t_{2_i}} - y_{t_{2_j}})^2 W_{t_{2_{ij}}}$,

and independently calculate $\mathbf{L}$ and $\mathbf{D}$ of the new chunk $\mathbf{t_2}$
where $D_{t_{2_{ii}}} = \sum_j W_{t_{2_{ji}}}$ and $\mathbf{L} = \mathbf{D} - \mathbf{W}$

Step 4: Use the updated eigenspace from the previous chunk belonging to the highest eigenvalue by incrementally solving the GEV of the new chunk t_2 to produce updated $\mathbf{w_1}^{\mathbf{new}}$ where $\mathbf{w_1}^{\mathbf{new}} = \mathbf{w_{t_1+t_2}}$ is the updated eigenspace of the first dimension

$\mathbf{L}_i {\mathbf{w_1}}^{t_1+t_2} = \lambda \mathbf{D}_i {\mathbf{w_1}}^{t_1+t_2}$.

Step 5: Use the updated eigenspace from the previous chunk belonging to the second highest eigenvalue by deflating $\mathbf{L_{new}}$ and $\mathbf{D_{new}}$ and incrementally solving the GEV of the new chunk t_2 to produce the updated $\mathbf{w_2}^{\mathbf{new}}$ where $\mathbf{w_2}^{\mathbf{new}} = {\mathbf{w_2}}^{t_1+t_2}$ is the updated eigenspace of the second dimension

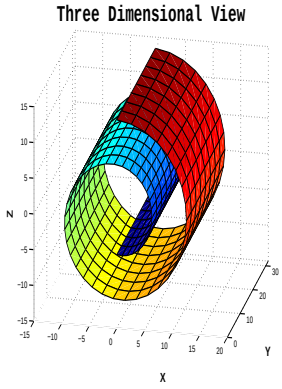

Figure 1: Swiss Roll Dataset

rate and the number of iterations for learning were initialized with the most appropriate values after checking their effect on the output. The results of our experiment conducted incrementally are shown in Figure 2. For our experiments on this artificial dataset we used the weight matrix **W** defined by

$$W_{ij} = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}, \text{if vertices i and j are connected} \\ 0, \text{otherwise} \end{cases}$$

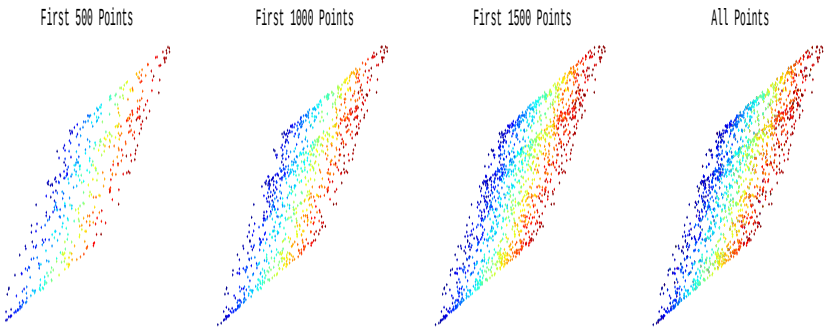

Figure 2: Incremental Laplacian Eigenmap Firstleft: Projections for first 500 datapoints. Secondleft: Projections for first 1000 datapoints. Thirdleft: Projections for first 1500 datapoints. Fourthleft: Projections for all the datapoints

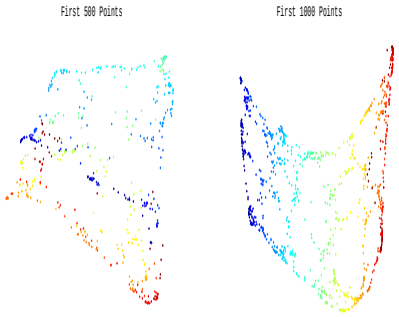

Figure 3: Batch Laplacian Eigenmap Firstleft: Projections for first 500 datapoints. Secondleft: Projections for the first 1000 datapoints.

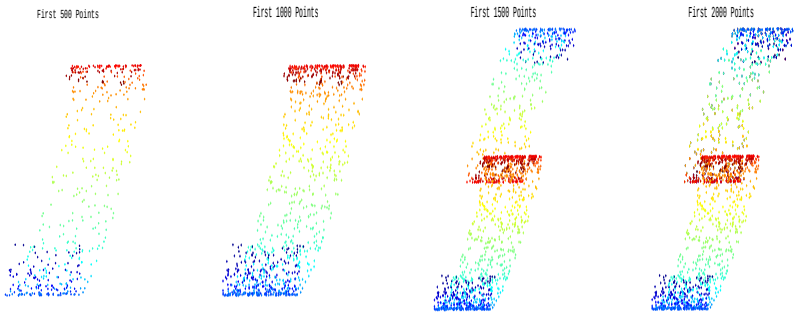

Figure 4: Incremental Laplacian Eigenmap Firstleft: Projections for first 500 datapoints. Secondleft: Projections for first 1000 datapoints. Thirdleft: Projections for first 1500 datapoints. Fourthleft: Projections for all the datapoints

6.1.1. Discussion

It has been commonly remarked that the swiss roll dataset is most of the times used to evaluate algorithms for manifold-based learning techniques [9]. In Figure 2, the projections calculated by using the novel incremental version of the Laplacian eigenmap are shown by splitting the first 2000 datapoints into four chunks assuming the leftmost to be the first chunk of 500 datapoints whose projections are first calculated and displayed. The next 500 datapoints together with the previous chunk are shown in the second sub-figure from the left, and then the other two, including the previously processed datapoints, are shown in the third and fourth sub-figures from the left. It can be easily seen that all the datapoints of the swiss roll dataset are properly revealed in two dimensions with minimum collisions despite the fact that the shape is rather flat than rolled but still all the points are properly clustered in the reduced dimensions.

In Figure 3 we have tested the same swiss roll dataset using batch laplacian eigenmap and tried to learn the manifold of the high dimensional dataset in two separate chunks where the first chunk consists of the first 500 datapoints and the second chunk consists of the first 1000 datapoints. As one can see very easily with the naked eye, the projection produced in both cases are completely different in shape and show no continuity between each other and of course the factor of repetitive re-calculation is always fermenting the computational efficiency in terms of batch processing.

One more interesting point to note is, by projecting the data on minimum eigenvectors in case of standard batch laplacian eigenmap version produces a very sparse kind of shape of a swiss roll with gaps between projected data points in the form of holes. On the other hand projecting the data in a purely adaptive manner on the eigenvector corresponding to the maximum eigenvalue using our proposed method produces very symmetrical results with no gap or sparseness between points in the reduced dimensions.

The simulation results of our newly proposed incremental algorithm are also given on the s-curve dataset in Figure 4. In the case of the s-curve dataset, our approach is able to produce the results in two dimensions that almost represent the shape of a s-curve as compared to the results produced for the swiss roll dataset. According to the results of real dataset shown in the next section, the improvement in clustering, classification and its purely incremental nature as compared to the standard batch laplacian approach are the actual strengths of our incremental approach.

6.2. Experiment on MNIST Digit Dataset

In terms of real data, the performance evaluation of the novel purely incremental approach compared with its standard batch version is firstly demonstrated on the MNIST digit dataset [30]. The MNIST digit dataset consist of 60,000 training patterns containing 0–9 handwritten digits and 10,000 test patterns. Each digit contains 784 pixels. The experiment is conducted by taking 500 datapoints each of the first four digits (0, 1, 2 and 3) and trying to learn the manifold of the high-dimensional data separately using both the standard and the incremental approach. The results are exhibited in Figure 5 in reduced dimensions.

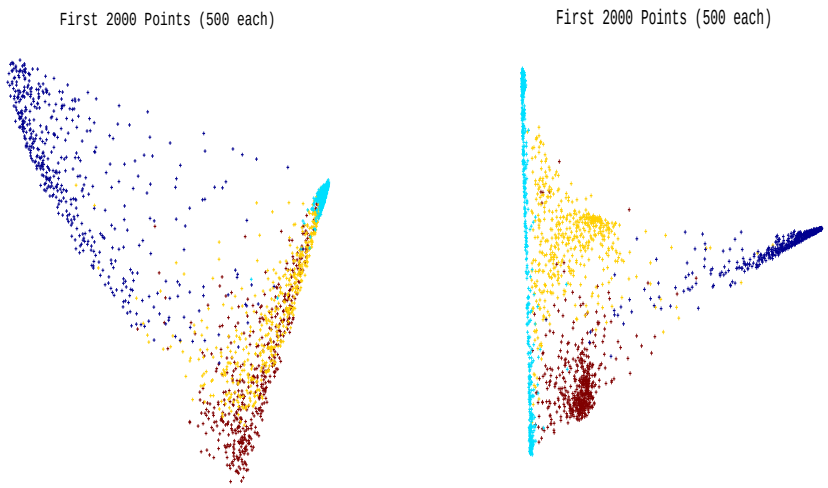

Figure 5: Left: Unfolding first 2000 points (500 each of digits 0,1,2 and 3) using standard LE. Right: Projections of first 2000 points (500 each of digits 0, 1, 2 and 3) using GENILE.

6.2.1. Discussion

The projections visualized in Figure 5 shows the same type of digits placed most of the times closer to each other using both the standard and incremental algorithms. In order to clarify the classifications and misclassifications of all the digits, we ran the k-nearest neighbor (kNN) algorithm on the projections of the data with $k = 5$. This will be assumed as the middle value after considering all the values of k from 1 to 10 in an odd manner. This enables

Table 3: The confusion matrices. Left: Batch Laplacian Eigenmap (LE). Right: GENILE.

	0	**1**	**2**	**3**
0	**480**	1	15	4
1	1	**482**	14	3
2	17	38	**345**	100
3	12	9	218	**216**

	0	**1**	**2**	**3**
0	**490**	0	7	3
1	0	**487**	9	4
2	15	27	**451**	7
3	6	9	28	**457**

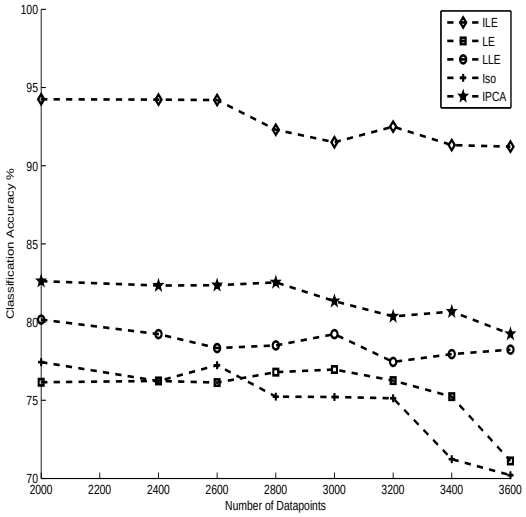

Figure 6: Comparison Between GENILE and other Dimensionality Reduction Methods

these neighbors to vote for the class of the particular datapoint: the most frequent digits among each digit's five neighbors will be considered as that particular digit's group. The results of the kNN algorithm for both the approaches are shown in the form of a confusion matrix in Table 7. According to the confusion matrix, the subtable on the left side shows the results of batch laplacian eigenmap) whereas the subtable on the right shows the results of the generalized incremental laplacian eigenmap (GENILE). It is clear from the results of both the standard and incremental techniques that the latter outperformed the other by showing the correct classification 1885 times out of 2000. The standard approach shows the correct classification 1523 times out of 2000, which is much less than the other. For further clarification, we

Table 4: Classification Accuracy of MNIST Digit Dataset

Algorithn	Classifier	Data Length	Dimensions	Accuracy %
IPCA	KNN	2000	784 (28 x 28)	77.44
LLE	KNN	2000	784 (28 x 28)	80.16
Isometric Projection	KNN	2000	784 (28 x 28)	82.62
LE	KNN	2000	784 (28 x 28)	76.15
GENILE	**KNN**	**2000**	**784(28x28)**	**94.25**

increased the number of digits in each class and tested its impact on the classification accuracy compared with other incremental and manifold-based learning algorithms. The comparative results are shown in Figure 6. There is a clear sight of improvement by our method compared with the existing batch version and other dimensionality reduction mechanism. The classification accuracy produced by our method is always above 90–% whereas with all the other method including incremental principal component analysis (IPCA) [31], local linear embedding (LLE) [11], isometric projection [32] and laplacian eigenmap (LE) [12] the classification accuracy is always below 90 % as shown in Table 4. The reason of the high classification accuracy produced by our method is its point-by-point learning nature and its projection to the eigenvector corresponding to the maximum eigenvalue which actually made a very clear difference of improvement in the classification accuracy compared with the existing batch laplacian eigenmap approach.

6.3. Experiment on the Banknote Authentication Data Set

Next we have chosen the banknote dataset taken from genuine and forged banknote-like specimens [33]. This dataset comprises five attributes: 1) variance of wavelet transformed image, 2) skewness of wavelet transformed image, 4) entropy of image, and 5) class information. The dataset is organized into two classes. We have tried to learn the manifold of this high-dimensional dataset and tried to reduce the dimensions to properly visualize the dataset in two dimensions. The dataset has a total of 1372 instances. The manifold of the whole dataset is learned by using both the standard and our novel incremental approach and the results in reduced dimensions are visualized in Figure 7.

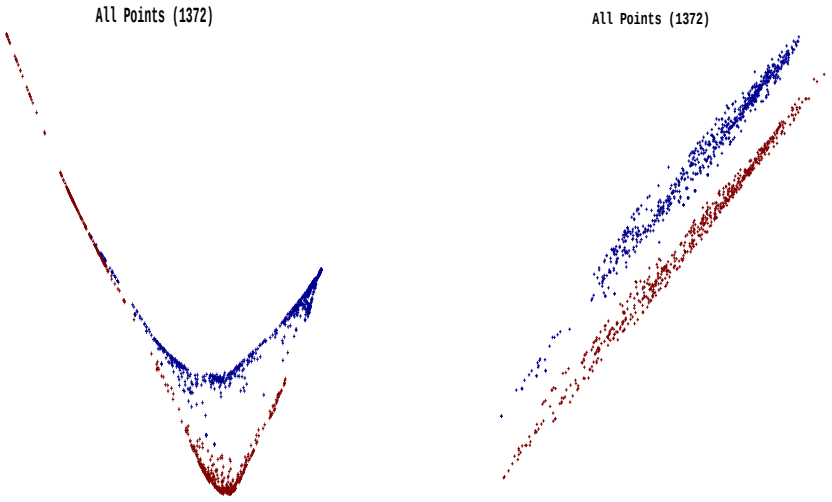

Figure 7: Left: Unfolding all 1327 points using standard LE. Right: Projections of all 1327 points using GENILE.

Table 5: The confusion matrices. Left: Batch Laplacian Eigenmap (LE). Right: GENILE.

	0	**1**
0	**757**	5
1	26	**539**

	0	**1**
0	**762**	0
1	11	**554**

6.3.1. Discussion

Here again the projections visualized in Figure 7 show the same category of notes placed closed to each other in the low dimensional latent space. The classification of both classes is clearly visible in both projections, but in order to find out which algorithm has produced a slightly higher degree of accuracy in terms of classifying the data properly, we again ran the k-nearest neighbor algorithm on the reduced projections produced by both the algorithms. The value of k is again taken as 5. We checked each datapoint's five nearest neighbors and labeled the datapoint based on the maximum number of datapoints from each class. The results of both the experiments are shown in the form of confusion matrix in Table 7, which clearly shows a slightly higher degree of classification again produced by our novel incremental algorithm, something

Table 6: Classification Accuracy of Banknote Dataset

Algorithn	Classifier	Data Length	Dimensions	Accuracy %
IPCA	KNN	1372	5	77.23
Isometric Projection	KNN	1372	5	94.52
LLE	KNN	1372	5	95.21
LE	KNN	1372	5	94.46
GENILE	**KNN**	**1372**	**5**	**95.92**

which is very rarely found in any other incremental version of dimensionality reduction methods as compared to their batch versions. Similarly here too we have compared the classification accuracy of our proposed method with other existing approaches which again includes IPCA [31], LLE [11] and isometric projection [32] as shown in Table. 6. The classification accuracy of our method is still higher compared with the other methods shown clearly in Table 6.

6.4. Experiment on Cardiovascular Disease Dataset

This is a manually collected dataset from one of the co-authors of this paper related to gathering more than one different type of attributes of the patients which helped in designing a framework for labeling the patients as cardiovascular or non-cardiovascular using standard supervised classifiers. In this paper, we used this real time dataset to check the performance of our novel incremental version of laplacian eigenmap in comparison with the traditional approach. The features of the original dataset are reduced to six by using a standard decision tree algorithm [34]. The reduced features are then clustered and classified by using the standard k-means algorithm so that patients of the same type come closer to each other in the higher dimensional space. The total number of patients is 558 and each patient has six attributes. The projections produced by both methods are visualized in Figure 8

6.4.1. Discussion

In Figure 8 the projections produced by both the methods are not very clearly seen, a confusion matrix is created which will enable us to find the

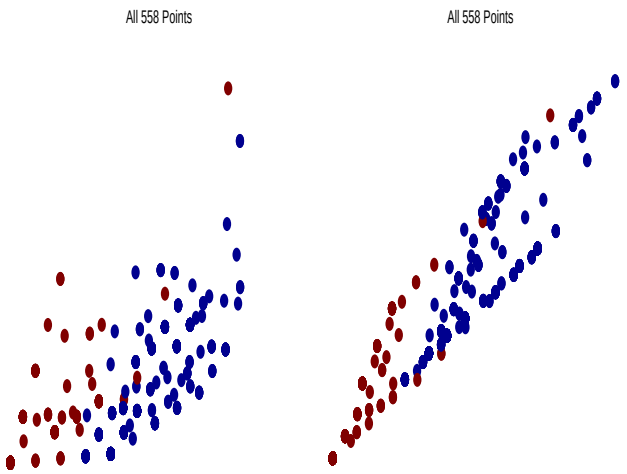

Figure 8: Left: Unfolding all 558 points using standard LE. Right: Projections of all 558 points using GENILE.

Table 7: The confusion matrices. Left: Batch Laplacian Eigenmap (LE). Right: GENILE.

	0	**1**
0	**391**	4
1	7	**156**

	0	**1**
0	**394**	1
1	5	**158**

difference in the classification accuracy between the classes produced by the traditional batch and by our newly proposed incremental method. According to Table 7, the batch version has attained a classification accuracy of 98.02 percent, whereas the new algorithm attained a classification accuracy of 98.93 percent, which is slightly higher than its batch version. The rationale behind using this manually collected dataset by one of the co-author of this paper is to test the practical significance of the proposed method in the paper and further comparing it with the other methods which includes IPCA [31], LLE [11], isometric projection [32] and LE [12] shown in Table 8. The classification accuracy produced by our method is still higher with very close difference with standard LE but very large difference with other manifold-based learning and dimensionality reduction incremental and batch

Table 8: Classification Accuracy of Banknote Dataset

Algorithn	Classifier	Data Length	Dimensions	Accuracy %
IPCA	KNN	558	6	86.43
LLE	KNN	558	6	84.23
Isometric Projection	KNN	558	6	85.32
LE	KNN	558	6	98.02
GENILE	**KNN**	**558**	**6**	**98.93**

techniques.

7. Conclusion

Results concluded that the technique evolved by us in this paper can be termed a purely incremental technique in that it is able to consider each datapoint separately while processing the whole dataset, compared to other incremental methods proposed in the literature which mostly do not work separately on each instance during the calculation. Results have also shown an always slightly higher classification accuracy produced by our method and its strongly adaptable nature as compared to the standard laplacian technique, which is considered inapplicable to online data, especially in scenarios where the data is coming in more than one chunk.

Future work will consider expanded information by mapping the input data on both the feature space and the kernel hilbert space before the calculation of the laplacian matrix and the diagonal matrix. Time series information will also be considered by using an echo state network.

[1] S. Haykin, Neural Networks: A comprehensive foundation, Macmillan, New Jersey, 2010.

[2] S. Papadimitriou, J. Sun, C. Faloutsos, Streaming pattern discovery in multiple time-series, VLDB '05 Proceedings of the 31st international conference on Very large data bases (2005) 697–708.

[3] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, R. Harshman, Indexing by latent semantic analysis, Journal of the American Society of Information Science 41 (6) (1990) 391–407.

[4] B. Moghaddam, Q. Tian, N. Lesh, C. Shen, T. Huang, Visualization and user-modeling for browsing personal photo libraries, International Journal of Computer Vision 56 (1-2) (2004) 109–130.

[5] H. Yin, On multidimensional scaling and the embedding of self-organising maps, Neural Networks 21 (2) (2008) 160–169.

[6] Y. Pang, Y. Yuan, K. Wang, Learning optimal spatial filters by discriminant analysis for brain-computer interface, Neurocomputing 77 (1) (2012) 20–27.

[7] Y. Pang, S. Wang, Y. Yuan, Learning regularized lda by clustering, IEEE Transactions on Neural Network and Learning Systems PP (99) (2014) 1–1.

[8] Y. Pang, X. Li, Y. Yuan, Robust tensor analysis with l1-norm, IEEE Transactions on Circuits and Systems for Video Technology 20 (2) (2010) 172–178.

[9] d. S. V. Tenenbaum, J.B., J. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (5500) (2000) 2319–2323.

[10] V. Silva, J. Tenenbaum, Global versus local methods in non-linear dimensionality reduction, Advances in neural information processing systems, Cambrdge, MA, USA. The MIT Press 15 (2003) 721–728.

[11] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.

[12] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensioanlity reduction and data representation, Neural Computation 15 (6) (2003) 1373–1396.

[13] X. He, S. Yan, Y. Hu, P. Niyogi, H. Zhang, Face recognition using laplacianfaces, IEEE Trans. Pattern Analysis and Machine Intelligence 27 (3) (2005) 328–340.

[14] D. Donoho, C. Grimes, Hessian eigenmaps: New tools for non-linear dimensionality reduction, In Proceeding of National Academy of Science 100 (2003, March) 5591–5596.

[15] K. Weinberger, L. Saul, Unsupervised learning of image manifolds by semidefinite programming, International Journal of Computer Vision 2 (1) (2006) 77–90.

[16] M. Brand, Charting a manifold, Proc. Conf. Advances in Neural Information Processing Systems MIT Press (2002) 985–992.

[17] Z. Zhang, H. Zha, Principal manifolds and nonlinear dimensionality reduction via tangent space alignment, Journal of Shanghai University (English Edition) 8 (4) (2004) 406–424.

[18] B. Nadler, S. Lafon, R. Coifman, I. Kevrekidis, Diffusion maps, spectral clustering and reaction coordinates of dynamical systems, Applied and Computational Harmonic Analysis 21 (1) (2006) 113–127.

[19] X. Liu, J. Yin, Z. Feng, J. Dong, Incremental manifold learning via tangent space alignment, In Artificial Neural Network in Pattern Recognition 4087 (2006) 107–121.

[20] Y. Pang, J. Deng, Y. Yuan, Incremental threshold learning for classifier selection, Neurocomputing 89 (2012) 89–95.

[21] M. Law, A. Jain, Incremental nonlinear dimensionality reduction by manifold learning, Pattern Analysis and Machine Intelligence, IEEE Transactions on 28 (3) (2006) 377–391.

[22] O. Kouropteva, O. Okun, M. Pietikainen, Incremental locally linear embedding, Pattern Recognition 38 (10) (2005) 1764–1767.

[23] Y. Bengio, J. Paiement, P. Vincent, O. Delalleau, N. Le Roux, M. Ouimet, Out-of-sample extensions for lle, isomap, mds, eigenmaps and spectral clustering, Advances in neural information processing systems 16 (2004) 177–184.

[24] H. Li, H. Jiang, R. Barrio, X. Liao, L. Cheng, F. Su, Incremental manifold learning by spectral embedding methods, Pattern Recognition Letters 32 (10) (2011) 1447–1455.

[25] D. Skocaj, A. Leonardis, Weighted incremental subspace learning, In The proceeding of European workshop on Cognitive Vision, Zurich, Switzerland (2002) 19–20.

[26] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, V. Kumar, Idr/qr: an incremental dimension reduction algorithm via qr decomposition, IEEE Transactions on Knowledge and Data Engineering 17 (9) (2005) 1208–1222.

[27] P. Jia, J. Yin, X. Huang, D. Hu, Incremental laplacian eigenmaps by preserving adjacent information between data points, Pattern Recognition Letters 30 (16) (2009) 1457–1463.

[28] Q. Zhang, Y. W. Leung, A class of learning algorithms for principal component analysis and minor component analysis, IEEE Transactions on Neural Networks 11 (2) (2000) 200–204.

[29] Z. Malik, A. Hussain, J. Wu, A novel biologically inspired approaches to extracting online information from temporal data, Cognitive Computation 6 (3) (2014) 1–13.

[30] Y. LeCun, C. Cortes, The mnist database of handwritten digits (1998), The Dataset is available at http://yann.lecun.com/exdb/mnist/.

[31] P. Hall, A. Marshall, R. Martin, Incremental eigenanalysis for classification, In BMVC 98 (1998) 286–295.

[32] D. Cai, X. He, J. Han, Isometric projection, In Proceedings of the National Conference on Artificial Intelligence, MenloPark, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 22 (1) (2007) 528.

[33] K. Bache, M. Lichman, UCI machine learning repository (2013).
URL `http://archive.ics.uci.edu/ml`

[34] L. Rokach, Data mining with decision trees: theory and applications, World Scientific, 2008.