Single- and multi-distribution dimensionality reduction approaches for a better data structure capturing

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 4, 2020, accepted November 5, 2020, date of publication November 17, 2020,
date of current version November 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3038460

Single- and Multi-Distribution Dimensionality Reduction Approaches for a Better Data Structure Capturing

LAURETA HAJDERANJ, DAQING CHEN, (Member, IEEE), ENRICO GRISAN, (Senior Member, IEEE), AND SANDRA DUDLEY, (Member, IEEE)
School of Engineering, London South Bank University, London SE1 0AA, U.K.

Corresponding author: Laureta Hajderanj (hajderal@lsbu.ac.uk)

This work was supported by the Joint Scholarship through London South Bank University and Acctive Systems Ltd.

ABSTRACT In recent years, the huge expansion of digital technologies has vastly increased the volume of data to be explored, such that reducing the dimensionality of data is an essential step in data exploration. The integrity of a dimensionality reduction technique relates to the goodness of maintaining the data structure. Dimensionality reduction techniques such as Principal Component Analyses (PCA) and Multidimensional Scaling (MDS) globally preserve the distance ranking at the expense of neglecting small-distance preservation. Conversely, the structure capturing of some other methods such as Isomap, Locally Linear Embedding (LLE), Laplacian Eigenmaps *t*-Stochastic Neighbour Embedding (*t*-SNE), Uniform Manifold Approximation and Projection (UMAP), and TriMap rely on the number of neighbours considered. This paper presents a dimensionality reduction technique, Same Degree Distribution (SDD) that does not rely on the number of neighbours, thanks to using degree-distributions in both high and low dimensional spaces. Degree-distribution is similar to Student-*t* distribution and is less expensive than Gaussian distribution. As such, it enables better global data preservation in less processing time. Moreover, to improve the data structure capturing, SDD has been extended to Multi-SDD*s* (MSDD), which employs various degree-distributions on top of SDD. The proposed approach and its extension demonstrated a greater performance compared with eight other benchmark methods, tested in several popular synthetics and real datasets such as Iris, Breast Cancer, Swiss Roll, MNIST, and Make Blob evaluated by the co-ranking matrix and Kendall's Tau coefficient. For further work, we aim to approximate the number of distributions and their degrees in relation to the given dataset. Reducing the computational complexity is another objective for further work.

INDEX TERMS Dimensionality reduction, global structure, local structure, visualization, structure capturing, manifold learning.

I. INTRODUCTION

High dimensional data are prone to the curse of dimensionality problem, and analysing them can be computationally expensive. Curse of dimensionality occurs when the dimensionality of data increases and the available data become sparse. Conversely, sparse data can be a problem if a machine learning/data mining algorithm to be applied requires that the number of samples be much larger than the data dimensionality to ensure reliable results. To solve this problem, two options could be considered: 1) increase data samples, or 2) reduce the data dimensionality. Increasing the data samples may not always be possible, and as a result, reducing the data dimensionality could be a crucial choice.

Dimensionality reduction is a process of converting data from a high dimensional space to a lower dimensional space with the aim of preserving meaningful information from the original data. Dimensionality reduction can be applied in any field that has high dimensional data (a large number of variables) such as signal processing [1], speech recognition [2], [3], neuroinformatics [4], [5], bioinformatics [6], [7], social media [8], [9], telecoms [10], and computer vision [11], for data visualization, data exploration, noise reduction or as a pre-processing step to support classification models.

The associate editor coordinating the review of this manuscript and approving it for publication was Nilanjan Dey.

An appropriate dimensionality reduction technique is related to the goodness of preserving the geometry (structure) of the data of interest. Maintaining the data structure means that close (far away) points in the original space are embedded closely (far away) in the low dimensional space. Additionally, dimensionality reduction techniques favour either local structure, that means capturing the distances of close points, or the global structure, that means the preservation of the distances of far away points. In general, Principal Component Analyses (PCA) [12] and Multidimensional Scaling (MDS) [13] are linear dimensionality reduction techniques that favour capturing the global structure of the data. By contrast, Sammon mapping [14] is nonlinear dimensionality reduction technique that favour the preservation of the local data structure. Conversely, the scale of data structure to be captured by nonlinear manifold learning methods[1] such as, Isomap [16], Locally Linear Embedding (LLE) [17], Laplacian Eigenmaps (LE) [18], [19], *t*-Stochastic Neighbour Embedding (*t*-SNE) [20], Uniform Manifold Approximation and Projection (UMAP) [21], and TriMap [22] relates to the number of neighbours considered by each method. The smaller the number of neighbours selected means a more local data structure is captured by the method, at the expense of neglecting some global information. Conversely, the higher the number of selected neighbours, the greater the improvement in capturing global structure but at the possible expense of losing local information. Additionally, when tuning the number of neighbours k, where $k : N - 1$ and N represents the number of samples, the user must consider the computational time of each algorithm, as the algorithm needs run $N - 1$ times with the different number of neighbours, to generate the best embedding in terms of structure capturing.

This research aims to present a nonlinear dimensionality reduction (manifold learning) approach, named Same Degree Distribution (SDD), and its extension Multi-Same Degree Distributions (MSDD), to better preserve the data structure using less computational time compared to the other manifold learning methods. SDD and MSDD do not rely on the number of neighbours to tune the scale of the structure to be preserved; but instead, they tune the degree of degree-distribution. The degree of the degree-distribution is responsible for the scale of the data structure to be maintained. By using degree-distribution(s), SDD and MSDD give priority to the local structure of the data. However, a degree-distribution with a low degree is more sensitive to large distances than a degree-distribution with a high degree. In other words, a degree-distribution with a low degree can capture more global structure of data than a degree-distribution with a high degree, but at the expense of losing some local information. Note that a degree-distribution with a high degree can improve the maintenance of the local data structure (small distances); however, it will fail to maintain the global data structure (large distances). As such, to find the best low dimensional data representation in terms of local and global structure capturing, we need to tune the degree of degree-distribution.

There does not exist an upper limit for the degree of the degree-distribution; however, a degree-distribution with degree 15 is acceptable sharp to capture the structure of the data having a large fraction of short distances. Note that the data distances will be scaled by their maximum value, as such, the scaled distance will range between 0 and 1. Because of this, tuning the degree of degree-distribution in the range from 1 to 15 will be sufficient to capture the best structure of data. Therefore, SDD and MSDD require fewer iterations than other manifold learning methods to find the best representation in a low dimensional space in relation to the maintained data structure.

Additionally, as a nonlinear method, SDD (MSDD) is expected to better capture the structure of nonlinear data than linear methods since the low dimensional representation of nonlinear data is located in nonlinear manifolds.[2] Furthermore, SDD employs degree-distribution, which is less expensive than Gaussian distribution, and as a consequence, the proposed approach is expected to be faster than the three other Gaussian distribution-based methods: *t*-SNE, UMAP, and TriMap.

The proposed SDD (MSDD) method has been tested with different datasets to demonstrate its ability to usefully maintain the data structure. It has been shown to outperform eight other dimensionality reduction methods including MDS, PCA, Isomap, LLE, LE, *t*-SNE, UMAP, and TriMap in terms of structure maintain once discovery the best structure within a superior computational time compared with *t*-SNE, UMAP, and TriMap. In datasets with a large number of samples, MSDD better captures the data structure using less computational time compared to all the considered manifold learning methods such as *t*-SNE, UMAP, TriMap, LE, LLE, and Isomap.

In this paper, a high dimensional dataset X is considered with a table of N observations and D attribute (columns). The embedding process requires embedding the dataset X^{NxD} into a new dataset Y^{Nxd}, where $d \ll D$. The i^{th} observation in the high and the low dimensional spaces are represented by x_i and y_i, respectively. $dis(x_i, x_j)$ and $dis(y_i, y_j)$ denote the Euclidean distance between x_i and x_j in the high dimensional space and the Euclidean distance between y_i and y_j in the low dimensional space. Additionally, the term distance indicates the Euclidean distance, n indicates the number of distributions employed, *deg* denotes the degree-distribution degree, and *pr* denotes the perplexity used in *t*-SNE.

The remainder of this paper is organized as follows; Section II presents the related works and offers a detailed discussion on their strengths and limitations with the case identified. The proposed approach is presented in Section III, followed by implementation in Section IV, experimental

[1]Manifold learning methods are the dimensionality reduction methods that try to learn the manifold hidden in high dimensional data [15].

[2]Manifold can be considered as the surface of objects such as a sphere, plane.

settings and results in Section V. Conclusions and further work are provided in Section VI.

II. RELATED WORKS

In this section, we briefly discuss some of the dimensionality reduction methods and underlying causes in terms of data structure capturing.

A. PRINCIPAL COMPONENT ANALYSIS (PCA)

PCA is a standard dimensionality reduction method widely applied in data analysis. Its aims to approximate the data by projecting them on a subspace formed by the largest eigenvectors. PCA makes use of matrix factorization to determine a linear mapping matrix $M \in R^{Nxd}$ (formed by d eigenvectors) to maximize the cost function:

$$max\ trace(M^T\, cov(X)M) \tag{1}$$

where $cov(x) \in R^{NxN}$ is the covariance matrix of the data X. The low dimensional representation of the data is calculated using linear mapping $M \in R^{Nxd}$ and the original data through the formula $Y = XM$. PCA can capture the global structure (i.e., maintain large distances); however, if the fraction of large distances is much higher than small distances, the cost function will maintain the large distances at the expense of the small distance preservation. Furthermore, PCA assumes that the low dimensional representation of high dimensional data lies on a linear submanifold. As a result, PCA does not effectively capture the structure of nonlinear data.

B. MULTIDIMENSIONAL SCALING (MDS)

MDS is also a widely applied dimensionality reduction technique that minimizes the cost function as expressed in (2),

$$min \sqrt{\frac{\sum_{i,j}((dis(x_i, x_j) - dis(y_i, y_j))^2}{\sum_{i,j}(dis(x_i, x_j))}} \tag{2}$$

where $B = XX' = -\frac{1}{2}JD^2J$ and $J = I - \frac{1}{N}11$ and $D^2 = [dis_{ij}^2]$. Determine with d the largest eigenvalues and with B the corresponding eigenvectors. E_d is the matrix of d eigenvectors and $V_d^{\frac{1}{2}}$ is the diagonal matrix of d eigenvalues, whereas the new space calculates through $Y = E_d V_d^{\frac{1}{2}}$. MDS is an excellent method in maintaining global data structure; however, like PCA, it is also prone to neglecting the maintenance of small distances and it is less useful in capturing the structure of nonlinear data.

C. SAMMON MAPPING

The problem caused by MDS has been addressed by the Sammon mapping method, which adapts weight scaling to the classical cost function as in (3).

$$min(\frac{1}{\sum_{i,j} dis(x_i, x_j)} \sqrt{\frac{\sum_{ij}((dis(x_i, x_j) - dis(y_i, y_j))^2}{\sum_{i,j}(dis(x_i, x_j))}}) \tag{3}$$

The main weakness of Sammon mapping is that it boosts the contribution of very close points of the cost function in (3)) [20]. Thus, PCA and MDS are expected to perform better in a dataset with a relevant fraction of large distances among data points. By contrast, Sammon mapping is expected to perform well in datasets with a large fraction of small distances.

D. ISOMAP

Isomap is a method which aims to exploit the geometry of nonlinear data by employing the Geodesic distance, computed as the sum of the shortest path between two data points in the neighbourhood graph [16]. In theory, Isomap has been designed to discover the global structure of the data; however, it requires tuning the number of neighbours, and this exponentially increases the computational time. Additionally, Isomap is prone to produce embedding errors even when there exists a small short-circuit [3] error in the data.

E. LOCALLY LINEAR EMBEDDING (LLE)

LLE is a nonlinear dimensionality reduction method, which embeds high dimensional data points into a lower dimensional space by assuming that every point and its nearest neighbours are located in a linear manifold. Also, each point x_i is defined as a linear combination of its k nearest neighbours [23] as follows:

$$\widehat{x}_i = \sum_{j=1}^{N} w_{ij}x_j \quad subject\ to \sum_j w_{ij} = 1, i = 1 : n. \tag{4}$$

LLE seeks to optimise the weights w_{ij} by solving

$$\widehat{W} = arg\min \sum_{i=1}^{N} \|x_i - \widehat{x}_i\|^2\ subject\ to \sum_j w_{ij} = 1,\ i = 1 : n. \tag{5}$$

The low dimensional representation $\widehat{Y}$ is produced by optimising the following cost function (6) with the weights obtained from (5).

$$\widehat{Y} = arg\min \sum_{i=1}^{N} \left\| y_i - \sum_{j=1}^{N} w_{ij}y_j \right\|^2 \tag{6}$$

The structure capturing of LLE is related to the number of neighbours k. If k is large, then LLE can be considered a linear dimensionality reduction method, as it assumes that every point and its neighbours are located in linear manifolds.

F. LAPLACIAN EIGENMAPS (LE)

LE is a nonlinear dimensionality reduction technique which embeds high dimensional data with a focus to maintain their local structure [18]. The similarity w_{ij} between x_i and x_j has been determined as:

$$w_{ij} = \left\{ \begin{matrix} exp(-\frac{dis(x_i,x_j)^2)}{2\sigma^2})\ if\ x_j \in Neig_i \\ 0\ otherwise \end{matrix} \right\} \tag{7}$$

[3] Distances of data in a neighbourhood are more significant than the distance between folds (regions) in manifolds [20].

Assuming the number of neighbours $k \leq N$ and let $Neig_i^k$ denote the neighbourhood of x_i with k neighbours. Let $D = (d_{ij})$ be a $N \times N$ diagonal matrix with elements $d_{ii} = \sum_{i \in N_i} w_{ij}$. The matrix $L = D - W$ is a symmetric matrix with $N \times N$ dimensions known as graph Laplacian. The low dimensional representation $Y = (y_1, \ldots, y_d)$, is defined by minimising the objective function (8)

$$arg\min trace(YLY^T), \tag{8}$$

where $\sum_i \sum_j w_{ij} dis(y_i, y_j) = YLY^T$. For a small k, since the weight is zero for points outside the neighbourhood, the global data structure is not captured. On the other hand, if the number of neighbours k is large, the method favours the preservation of more global information. Thus, the structure capturing of LE is related to tuning the number of neighbours k, increasing the computational time.

G. t-STOCHASTIC NEIGHBOUR EMBEDDING (t-SNE)

t-SNE is a nonlinear dimensionality reduction technique which calculates the conditional probability $p_{i|j}$ between samples x_i and x_j using the Gaussian distribution, centred at x_j with the variance σ_i as in (9).

$$p_{i|j} = \frac{exp(\frac{-dis(x_i,x_j)^2}{2\sigma_i^2})}{\sum_{k \neq i} exp(\frac{-dis(x_i,x_k)^2}{2\sigma_i^2})} \tag{9}$$

The high dimensional space similarity p_{ij} is calculated as $p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N}$, whereas, the low dimensional similarity is calculated as in (10).

$$q_{ij} = \frac{(1 + dis(x_i, x_j)^2)^{-1}}{\sum_{k \neq l} (1 + dis(x_i, x_j)^2)^{-1}} \tag{10}$$

t-SNE tries to make the low dimensional similarity q_{ij} as similar as possible to its corresponding high dimensional similarity p_{ij}. Consider (9), t-SNE builds n-Gaussian distributions, which are related to density σ_i, and to the distance of each sample x_i to its neighbours. If the distance between the sample x_i and its neighbours is small, then the Gaussian distribution is sharp; otherwise, it broadens. However, there can be some scenarios in which different variables involved in different Gaussian distributions may produce the same probability (similarity), the so-called *confusing samples problem*. In other words, points with different Euclidean distances in the high dimensional space might be mapped in such a way that they have the same Euclidean distance in the low dimensional space, resulting in a failure with regards to data structure capturing. Besides, the goodness of the captured structure of low dimensional data generated by t-SNE relies on perplexity, as an indication of the number of neighbours.

To also capture a more global structure of the data, Zhou and Sharpee [24] presented the Global t-SNE method. Global t-SNE suggests using an exponential distribution in addition to Gaussian distribution. If the Gaussian distribution is sensitive to smaller distances, the exponential distribution is unstable to larger distances because of its heavy tail. However, in the same way to t-SNE, and in addition to the additional time needed to tune the parameter k, Global t-SNE fails to maintain distances of confusing samples.

H. MULTISCALE SNE

One possible solution to the problem mentioned when using t-SNE and Global t-SNE could be to employ multi-perplexities in high dimensional space as in [25] to maintain small and large distances. In fact, Multiscale SNE is an extension of Stochastic Neighbour Embedding (SNE) [26], using Gaussian distributions in high and low dimensional spaces to maintain the data structure, it defines the probabilities as follows:

$$p_{hij} = \frac{exp(\frac{-r_{hi} dis(x_i,x_j)^2}{2})}{\sum_{k \neq i} exp(\frac{-r_{hi} dis(x_i,x_j)^2}{2})} \tag{11}$$

$$q_{hij} = \frac{exp(\frac{-s_{hi} dis(x_i,x_j)^2}{2})}{\sum_{k \neq i} exp(\frac{-s_{hi} dis(x_i,x_j)^2}{2})} \tag{12}$$

$$p_{ij} = \frac{1}{L} \sum_{h=L_{min}}^{L_{max}} p_{hij} \tag{13}$$

$$q_{ij} = \frac{1}{L} \sum_{h=L_{min}}^{L_{max}} q_{hij} \tag{14}$$

where r_{hi} and s_{hi} denote precision in high and low dimensional spaces, respectively, and $1 \leq L_{min} \leq h \leq L_{max}$ where $L = L_{max} - L_{min} + 1$ is considered the number of scales (number of different perplexities employed). In [25] it is suggested using $L_{min} = 2$ and $L_{max} = \log_2 \frac{N}{2}$. Multiscale SNE improves capturing a more global structure, but increases the computational complexity by $\log_2 \frac{N}{2}$. Tuning the scale parameter determines the efficiency of the algorithm, and this makes multiscale SNE more complex and costly.

I. UNIFORM MANIFOLD APPROXIMATION AND PROJECTION (UMAP)

UMAP, a similar method to t-SNE is a useful technique to capture the local structure of the data. For each x_i let define ρ_i and σ_i where

$$\rho_i = \min(dis(x_i, x_j), 1 \leq j \leq k, dis(x_i, x_j) \geq 0) \tag{15}$$

$$\sum_{j=1}^{k} exp(\frac{-\max 0, dis(x_i, x_j) - \rho_i}{\sigma_i}) = \log_2 k \tag{16}$$

and the similarity function is defined as in (17).

$$w_{ij} = exp(\frac{-\max 0, dis(x_i, x_j) - \rho_i}{\sigma_i}) \tag{17}$$

Based on (17), UMAP gives more importance to the local structure capturing than the global structure of the data, and requires tuning the parameter k to generate the best embedding in terms of preserving the data structure.

J. TRIMAP

To capture a more global structure of the data, Amid and Warmuth presented the TriMap method, which considers the similarities of three points (triplets) instead of a pair of points. TriMap defines a set of triplets $T = \{(i, j, k) : p_{ij} > p_{ik}\}$ where the satisfaction probability of the triplet (i, j, k) is defined as in (18).

$$Pr_{ijk} = \frac{q_{ij}}{q_{ij} + q_{ik}} = \frac{1}{1 + \frac{q_{ik}}{q_{ij}}} \tag{18}$$

The low dimensional representation can be calculated by minimising the cost function

$$\min_{\{y_n\}} - \sum_{(i,j,k)\in T} w_{i,j,k} \log Pr_{ijk}, \tag{19}$$

where $w_{ijk} = \frac{p_{ij}}{p_{ik}}$ is the weight of the triplet (i, j, k). The probability p_{ij} in the high dimensional space is calculated as:

$$p_{ij} = exp(-\frac{dis(x_i, x_j)^2}{\sigma_{ij}^2}) \tag{20}$$

where $\sigma_{ij}^2 = \sigma_i \sigma_j$ and σ_i is set to the average distance of x_i to its 10^{th} to 20^{th} nearest neighbours. TriMap proposes using the function in (21) as a function for the similarities calculation in the low dimensional space.

$$q_{ij}^{t'} = \begin{Bmatrix} exp(-dis(y_i, y_j)^2) & if \quad t' = 1 \\ 1 + (1 - t')(-dis(y_i, y_j)^2)^{\frac{1}{1-t'}} & otherwise \end{Bmatrix} \tag{21}$$

TriMap, similarly to t-SNE, employs a Gaussian distribution in the high dimensional space and Student-t distribution in the low dimensional space. As demonstrated with t-SNE, using different Gaussian distributions in the high dimensional space and one Student-t distribution in the low dimensional space cause the so-called confusing sample problem, which also occurres in TriMap.

K. AUTOENCODERS AND RESTRICTED BOLTZMANN MACHINE (RBM)

Autoencoders[4] are neural networks composed of two parts *encoder* and *decoder*. The encoder uses ϕ function (22) to embed the original high dimensional data X to the low dimensional data Y. In contrast, the decoder uses the function ψ (23) to embed the low dimensional data Y to the output data X', where X' is the reconstructed data of the original data X by minimizing the cost function in (24).

$$\phi : X \to Y \tag{22}$$

$$\psi : Y \to X \tag{23}$$

$$\phi, \psi = arg \min_{\phi,\psi} ||X - (\psi \circ \phi)X'||^2 \tag{24}$$

Deep autoencoders [27]–[31] are multi-layered neural networks, where each pair of neighbourhood-layers is considered to be an Restricted Boltzmann Machine (RBM). However, like all neural networks, it is difficult to find the optimal parameters for RBMs; and as such, their selection is heuristic, or based on previous experiments [32]. Above all, most of the methods aforementioned disregard the preservation of the data manifold structure [33]. Hence, to improve RBM and to capture the local data structure, neighbourhood graphs have been used [33]. However, it is complex to implement this approach since it requires to tune not only the number of neighbours, but also the number of hidden layers, the number of nodes in each hidden layer, the number of epochs, and the batch size.

In summary, MDS, PCA, and Isomap concentrate on the maintenance of the global structure of data, whereas LE, LLE, t-SNE, UMAP, and TriMap favour the maintenance of the local structure of data. Furthermore, the scale of maintained data structure by Isomap, t-SNE, LLE, LE, UMAP, and TriMap relates to the number of neighbours considered by each method. Note that, tuning the number of neighbours will inevitably increase the computational time of the methods mentioned above. Contrastingly, PCA and MDS do not require parameter tuning, and therefore save computational time. However, they neglect the maintenance of local data information and fail to capture the structure of nonlinear data. Sammon mapping has been proposed as a nonlinear version of PCA and MDS, but focuses on short-distance preservation, at the expense of global information losses. t-SNE, UMAP, and TriMap have proposed using Gaussian and Student-t distributions to provide a softer border between local and global structure maintenance; however, as mentioned above, they require tuning the number of neighbours to generate the best low dimensional representation in terms of maintained data structure. Multiscale approaches such as Multiscale-SNE attempted to overcome this shortcoming; however, it still is a costly method due to both the multiscale calculations and the utilization of Gaussian distribution, and it is much slower than using Student-t distribution. Overall, the above-mentioned dimensionality reduction techniques favour either local or global data structure. For some methods, parameter tuning, which increases the computational cost and complicates the applicability of the methods, has a significant impact on the maintenance of the data structure.

This paper proposes the Same Degree Distribution (SDD) method for dimensionality reduction, together with Multi Same Degree Distributions (MSDD), aiming to capture the geometry of data by employing the same degree-distribution(s) in the high and the low dimensional spaces. SDD and MSDD use degree-distribution(s), in which degree-distribution ($deg = 1$) is the same as Student-t ($deg = 1$), and for greater degrees, degree-distributions

[4]Autoencoders are neural networks composed of one input layer, one output layer and one hidden layer, whereas deep autoencoders are multi-layered neural networks composed of one input layer, one output layer and many hidden layers.

($deg > 1$) are sharper than Student-t distributions ($deg > 1$). Note that the scale of the maintained data geometry relates to the degree of the degree-distribution. A degree-distribution with a high degree is very sensitive to small distances and the lower the degree, the more sensitive to large distances the degree-distribution becomes, at the expense of losing some local information. To find the best low dimensional representation of data, tuning the degree of degree-distribution is essential. Because we scale the distances of data by their maximum value, tuning the degree of degree-distribution in the range from 1 to 15 will be sufficient to capture the structure of data, even when data has a large fraction of short distances. As a result, SDD (MSDD) requires fewer iterations to find the best representation in the low dimensional space in terms of structure capturing compared to other methods such as Isomap, LLE, LE, t-SNE, UMAP, and TriMap, which require to tune the number of neighbours up to the number of samples (N) $-$ 1. Furthermore, SDD (MSDD) values have a smooth difference between far away and close points, which is an advantage over MDS, PCA, and Isomap, where errors generated by embedding far away points have a higher impact than errors generated by embedding closer points.

However, SDD (MSDD) is expected to perform less favourably with datasets that have high negative skewness in distance distribution, due to a large number of records located in the tail of degree-distribution(s). Note that in this case, the tail of a degree-distribution is not sharp enough, and therefore, large differences between any two large distances are reflected in small differences between the two corresponding degree-distribution similarities.

III. PROPOSED APPROACH

A. SAME DEGREE DISTRIBUTION (SDD) APPROACH

SDD is a nonlinear dimensionality reduction technique with pseudocode shown in Algorithm 1. It employes degree-distribution in the high (27) and the low (28) dimensional spaces to capture the local and global data structure. Degree-distribution is Student-t distribution when the degree of freedom is 1, and for greater degrees, it looks as sharper Student-ts. SDD intends to find a suitable degree to best capture the structure of the data. Degree-distributions are more sensitive to small distances, the greater the distance, the less sensitive degree-distribution becomes, such that, scaling the pairwise distances of high dimensional data into the range between 0 and 1 would be an essential step in the performance of the proposed approach in terms of capturing data structure. As a result, high dimensional space similarities of a degree-distribution will be calculated using the scaled Euclidean distances instead of the Euclidean distances. Kullback-Leibler is the loss function used in SDD to approximate the degree-distribution in the low dimensional space with the degree-distribution in the high dimensional space:

$$C_1 = \sum_{i \neq j} (p_{deg_m})_{ij} \log(\frac{(p_{deg_m})_{ij}}{(q_{deg_m})_{ij}} \tag{25}$$

Algorithm 1 SDD

Require: **Input** :
$X \in R^{NxD}$, number of iterations H, learning rate η, momentum α, number of degree-distributions n, degree deg_m, initial low dimensional data $Y^0 = y_1, \ldots, y_N \in N(0, 10^{-4}I)$.
Step 1 :
Compute the high dimensional space similarities $(p_{deg_m})_{ij}$ using (27).
Step 2 :
Compute the low dimensional space similarities $(q_{deg_m})_{ij}$ using (28).
Step 3 :
Compute the gradient $\frac{\delta C}{\delta\, yi}$ where C_1 is defined in (25).
Step 4 :
Minimize the objective function using the Gradient Descent optimisation algorithm: $Y^h = Y^{h-1} + \eta \frac{\delta C_2}{\delta y_i} + \alpha(Y^{h-1} - Y^{h-1})$.
Output :
Low dimensional space represenation $Y_{bestdeg_m}$.

where deg_m is the degree of degree-distribution m, $m = 1 : n$. SDD intends to minimize the cost function C_1 as (26):

$$loss_1 = \min(C_1) \tag{26}$$

where

$$(p_{deg_{ij}})_{ij} = \frac{(1 + dis(x_i, x_j))^{-deg_m}}{\sum_{k \neq l}(1 + dis(x_k, x_l))^{-deg_m}} \tag{27}$$

$$(q_{deg_{ij}})_{ij} = \frac{(1 + dis(y_i, y_j))^{-deg_m}}{\sum_{k \neq l}(1 + dis(y_k, y_l))^{-deg_m}} \tag{28}$$

However, the minimal loss function value of (26) does not reflect how well the data structure is captured. Thus, to have a better indication of the goodness of a dimensionality reduction method, we propose the use of Kendall's Tau correlation coefficient (τ). This coefficient (τ) measures the correlation between distance rank of the high and the low dimensional data as in (29):

$$\tau = \frac{C - D}{\sqrt{((C + D + T) * (C + D + U))}} \tag{29}$$

where the number of concordant pairs is denoted with C, and the number of discordant pairs is denoted with D, while T and U are the numbers of ties in pairwise distance matrices of the high and the low dimensional spaces DIS and dis, respectively. If a tie occurs for the same pair in both DIS and dis, it will not be added to either T or U, and the input of the data should be in a one-dimensional array. Therefore, the pairwise distance matrixes in both the high dimensional space (DIS) and the low dimensional space (dis) will be flattened to a one-dimensional array. The value of τ ranges between -1 and 1. If τ is close to 1, it means that there is a high

correlation between ranks. On the other hand, if τ is close to -1 or 0, it means there is no relation or negative relation between ranks. Ranks of distances between the high and the low dimensional spaces represent the ranks of neighbours for both spaces, respectively. Consequently, a high value of τ means that the neighbour's rank is captured. In terms of comparison, the best dimensionality reduction method is the method with the highest value of τ. For some datasets, one degree-distribution is not sufficient to capture enough data structure, and therefore more degree-distributions are needed to be applied. To deal with this, we present a multi-distribution-based approach Multi SDD (MSDD), discussed below.

B. MULTI SAME DEGREE DISTRIBUTION (MSDD) APPROACH

MSDD involves multi degree-distributions instead of one degree-distribution (SDD), to better capture the data structure; such that, MSDD can be described as an extension of SDD. The pseudocode of MSDD is demonstrated below in Algorithm 2. MSDD employs n degree-distributions, as such n-objective functions must be optimised. Multi-objective optimisation problems are classically solved using scalarisation techniques [34], [35]. MSDD will be optimised using the composed Kullbak-Leibler (s) as in (30) via the optscalarisation techniques [34]:

$$C_2 = a_1 \sum_{i \neq j} (p_1)_{ij} \log(\frac{(p_1)_{ij}}{(q_1)_{ij}} + \ldots + a_n \sum_{i \neq j} (p_n)_{ij} \log(\frac{(p_n)_{ij}}{(q_n)_{ij}} \tag{30}$$

To simplify the problem, we allocate the same influence to each degree-distribution (weight $a_m = 1, m = 1 : n$) in (30). So, the parameters to be tuned are the number of degree-distributions n and the degree of each degree-distribution $deg_m, m = 1 : n$. The problem can be formulated as below:

$$C_2 = \sum_{m=1}^{n} \sum_{i \neq j} (p_{deg_m})_{ij} \log(\frac{(p_{deg_m})_{ij}}{(q_{deg_m})_{ij}} \tag{31}$$

$$loss_2 = \min(C_2) \tag{32}$$

The performance of SDD and MSDD and other dimensionality reduction methods will be tested and compared using two different quality measures: co-ranking matrix and τ.

C. QUALITY ASSESSMENT

In addition to Kendall's Tau (τ) (29), we also use the co-ranking matrix [36] to measure the quality of each dimensionality reduction method. Let us define $DIS_{N\times N}$ and $dis_{N\times N}$ the matrixes of pairwise distances in the high and low dimensional spaces, respectively. In both spaces the rank matrices R_{NXN} and r_{NXN} of the distance matrixes DIS_{NXN} and dis_{NXN} are calculated as follows:

$$R_{ij} = \left|\{k : DIS_{ik} < DIS_{ij}\}\right| \tag{33}$$

$$r_{ij} = \left|\{k : dis_{ik} < dis_{ij}\}\right| \tag{34}$$

Algorithm 2 MSDD

Require: Input :

$X \in R^{NxD}$, number of iterations H, learning rate η, momentum α, number of degree-distributions n, degree deg_m, $Degrees = bestdeg_m$ from Algorithm I, $\tau_{actual} = \max(\tau)$, initial low dimensional data $Y^0 = y_1, \ldots, y_N \in N(0, 10^{-4}I)$.

Step 1 :

Compute the high dimensional space similarities $(p_{d_m})_{ij}$ using (27).

Step 2 :

Compute the low dimensional space similarities $(q_{deg_m})_{ij}$ using (28).

Step 3 :

Compute the gradient $\frac{\delta C_2}{\delta\, yi}$ where C_2 defined in (31) is reformulated as: $C_2 = \sum_{m \notin Degrees} \sum_{i \neq j} (p_{deg_m})_{ij} \log(\frac{(p_{deg_m})_{ij}}{(q_{deg_m})_{ij}} + \sum_{m \in Degrees} \sum_{i \neq j} (p_{deg_m})_{ij} \log(\frac{(p_{deg_m})_{ij}}{(q_{deg_m})_{ij}}$.

Step 4 :

Minimize the objective function using the Gradient Descent optimisation algorithm: $Y^h = Y^{h-1} + \eta \frac{\delta C_2}{\delta y_i} + \alpha(Y^{h-1} - Y^{h-1})$.

Step 5 :

Add more degrees in cases: $if\, \tau_{new} < \tau_{actual}$, $Degrees = Degrees \bigcup deg_{m\, with\, \tau_{new}}$, $\tau_{actual} = \tau_{new}$.

Output :

Low dimensional space represenation $Y_{Degrees}$.

where $|\cdot|$ defines the set of cardinality. The co-ranking matrix Q is defined by

$$Q_{kl} = \left|\{(i,j) : R_{ij} = k \; and \; r_{ij} = l\}\right| \tag{35}$$

Errors generated by a dimensionality reduction method correspond to off-diagonal entries of the co-ranking matrix [36]. A diagonal co-ranking matrix represents a perfect dimensionality reduction method.

D. COMPLEXITY ANALYSIS

SDD needs to create two matrixes with $N \times N$ to store distances in both high and low dimensional spaces and another matrix that stores the difference $P - Q$ with $N \times N$. In total, the complexity of SDD is $3N^2$. MSDD computational complexity is higher and is related to the number of degree-distributions involved. The computational complexity is $3nN^2$, where n is the number of degree-distributions, and hence it requires n times more than SDD. Because the number of degree-distributions affects the computational complexity, we suggest starting from one degree-distribution and then increasing the number of degree-distributions. The number of degree-distributions used in MSDD will be that number that produces the highest value of the correlation coefficient τ.

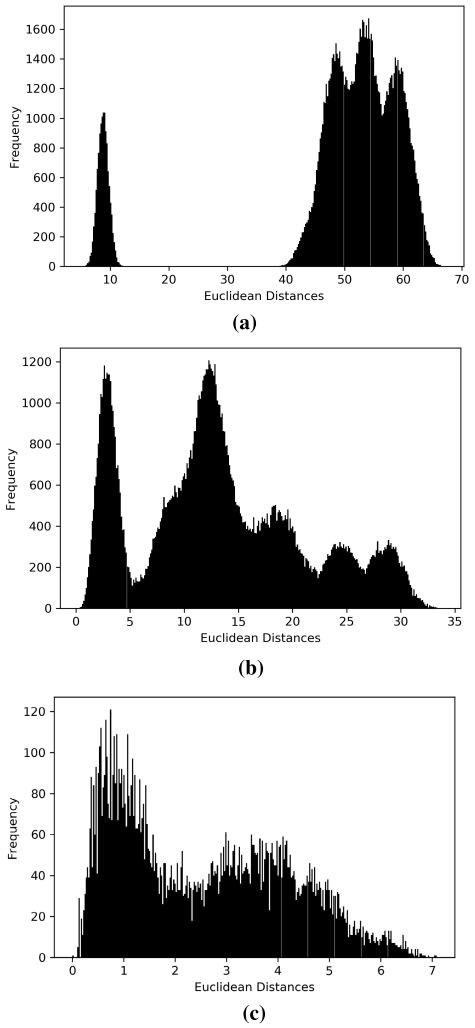


FIGURE 1. Three distance distributions.

IV. IMPLEMENTATION GUIDANCE OF SDD AND MSDD

In this Section, we present some guidance on the implementation of the SDD and MSDD approaches. The performance of SDD is related to the degree(s) of degree-distribution(s), and the selection of the degree of degree-distribution associates with 1) the high dimensional data distance distribution and 2) the dimensionality reduction purpose. In the case of data with a large fraction of large distances and small fractions of small distances, as shown in Fig. 1(a), employing small degree degree-distribution(s) is suggested. Degree-distributions with a small degree (i.e. *deg* 1, 2), has heavy tails, which means high sensitivity to large distances. High degree (*deg*>5) degree-distribution(s) is suggested to be employed in datasets that have a large fraction of small distances (Fig. 1(c)), and medium degree degree-distribution(s) should be employed in datasets with a large fraction of medium distances (Fig. 1(b)). However, this is an intuitive judgement, and the simulations provided later will generate precise results. If for a user, the local structure of the data is more important than the global structure, we suggest employing high degree degree-distribution(s); otherwise, the employment of low degree degree-distribution(s) may be more beneficial. The degree of degree-distribution which captures the best structure of the data we have named *best degree*. If we add degree-distribution(s) with degree(s) far from the best degree, then we might lose some local or global structure of the data. The degree(s) close to the best degree might contribute to maintaining better the data structure by not affecting the actual maintained data structure.

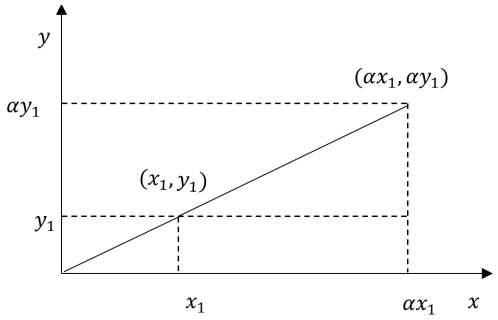


FIGURE 2. Scaled Euclidean distance.

Defining the degree of a degree-distribution also depends on the distance range, and it has to be noted that degree-distributions are less sensitive to large distances. To solve this problem, we propose scaling the distance ranges to the interval range from 0 to 1.

A. SCALING THE DISTANCE RANGE

To scale the pairwise distance range, we propose dividing every single distance on pairwise distances with a decent positive number. The Euclidean distance between x_1, y_1 is calculated as $dist(x_1, y_1) = \sqrt{x_1^2 + y_1^2}$, whereas the Euclidean distance between $\alpha x_1, \alpha y_1$ is calculated: $dist(\alpha x_1, \alpha y_1) = \sqrt{(\alpha x_1)^2 + (\alpha y_1)^2} = \sqrt{(\alpha)^2(x_1)^2 + (\alpha)^2(y_1)^2} = \sqrt{(\alpha)^2((x_1)^2 + (y_1)^2)} = \alpha\sqrt{((x_1)^2 + (y_1)^2)} = \alpha dist(x_1, y_1)$. As such, it is proved that if all sample values are scaled by a positive number α, the Euclidean distance calculated between the scaled samples also scales by the positive number α. Distributions of Euclidean distance and the scaled Euclidean distance can be visually seen in Fig. 3(a) and Fig. 3(b), respectively. In SDD(MSDD), $\alpha = \frac{1}{\max dis(x_i, x_j)}$, due to the high sensitivity of the degree-distribution(s) in the value range between 0 and 1.

V. EXPERIMENTAL SETTINGS AND RESULTS

In this Section, the proposed method and its extension is tested and compared with several benchmark dimensionalitiy reduction techniques; PCA, MDS, Isomap, LLE, *t*-SNE, UMAP, and Trimap, using several typical benchmark datasets including Iris, Breast Cancer, Swiss roll, MNIST, and Make Blob. All algorithms were implemented in Python with the same number of iterations (2000). PCA, MDS, Isomap, LLE, LE, and *t*-SNE, were implemented using their Sklearn

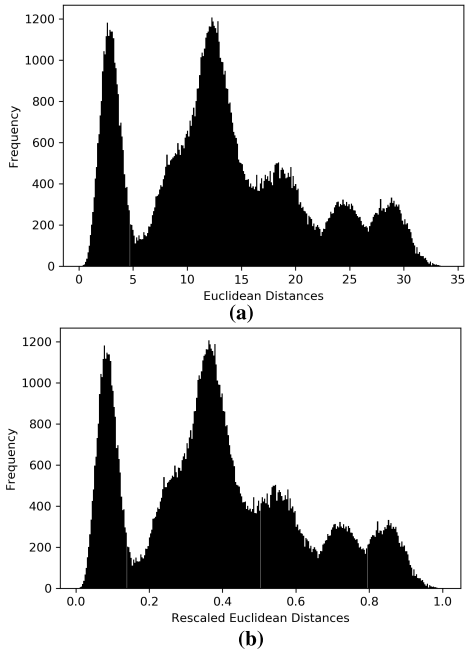


FIGURE 3. Distributions of Euclidean distances(a) and scaled Euclidean distances (b) of Make Blob data with 500 samples.

versions, and for UMAP[5] and Trimap,[6] their GitHub versions were applied. For Isomap, LE, UMAP, and *t*-SNE, the parameter k (pr for *t*-SNE) was tuned in the range $(1, N-1)$, to find an appropriate number of neighbours which could produce the best low dimensional representation in relation to the structure capturing. For LLE, in MNIST dataset, the number of neighbours k was tuned up to 1000, due to the memory problem. TriMap also failed to obtain a number of neighbours of more than 199, so the number of neighbours was tuned up to 198.

The effectiveness of each method was evaluated using the co-ranking matrix and τ (Kendall's Tau). Co-ranking matrix indicates a perfect mapping if the matrix is diagonal, and the off-diagonal entries are the errors. τ takes values between -1 and 1, and when τ is 1, there exists a perfect correlation between ranks corresponding an ideal mapping. The performance of each method in terms of Kendall's Tau τ is presented in Table 1 along with the computational time t (in seconds) and the number of neighbours k (perplexity pr for *t*-SNE). The two-dimensional data representations of the methods are presented in Fig. 9 and Fig. 10, and their co-ranking matrixes are presented in Fig. 11 and Fig. 12.

A. IRIS

The first dataset considered is Iris with 4 dimensions (attributes) and 150 samples, with distance distribution shown in Fig. 4. Based on the distance distribution in Fig. 4, the largest fraction of samples has relatively short and

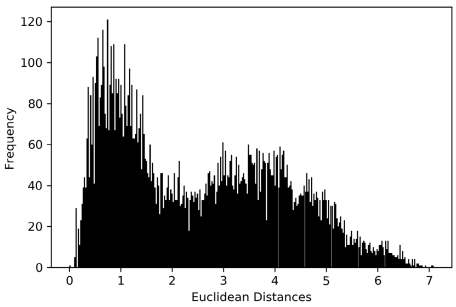


FIGURE 4. Euclidean distance distribution of Iris dataset.

medium distances, in which, our proposed approach is expected to perform better than others. From the simulation results, the best method with the highest τ of 0.967347 (Table 1) was MSDD (*deg*: 8). Its co-ranking matrix is shown in Fig. 11(a) has fewer off-diagonal entries in the top-centre sections, which indicates a good short and medium distance preservations. However, the co-ranking matrix of MSDD (*deg*: 8) has more off-diagonal entries than the co-ranking matrixes of Isomap shown in Fig. 11(p) and PCA showed in Fig. 11(k), in the bottom right sections. Thus, for the Iris dataset, MSDD (*deg*: 8) performed better than the other methods in terms of local structure capturing, and it performed similarly with Isomap and PCA for global structure preserving. Considering the computational time, as shown in Table 1, MSDD was more expensive than PCA, MDS, Isomap, and LE; however, it outperformed *t*-SNE, UMAP, and TriMap. MSDD (*deg*: 8) achieving the highest τ for the Iris dataset, and adding more degree-distributions did not improve the data structure capturing, but instead, could make it worse. As shown in Table 1, MSDD (*degs*: 7 and 8), MSDD (*degs*: 8 and 9), and MSDD (*degs*: 7, 8 and 9) generated lower τ, and as a result, less data structure was maintained.

B. BREAST CANCER

The Breast Cancer dataset[7] with 30 attributes is the second datasets considered. The distance distribution of breast cancer data is shown in Fig. 5, where the majority of samples have relatively short distances, in which MSDD is expected to maintain better the data structure. MSDD (*degs*: 9 and 10) and MSDD (*degs*: 10 and 11) were the dimensionality reduction approaches that produced the highest τ of 0.998125, as shown in Table 1. However, MSDD (*deg*: 10) achieved a similar τ (0.998122) with less computational time. Analyzing the co-ranking matrixes of the Breast Cancer dataset in Fig. 11, and Fig. 12, we can see that MSDD (*deg*: 10) performed better than other methods in maintaining the short, medium and large distance (less off-diagonal entries of the co-ranking matrix of MSDD in Fig. 11(b) than the rest of the co-ranking matrixes). Considering the computational time, MSDD was more expensive than PCA, MDS, LE; however, it was more

[5]https://github.com/lmcinnes/umap
[6]https://github.com/eamid/trimap

[7]Load breast cancer from sklearn, Python.

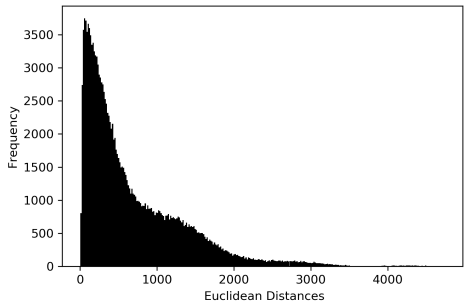
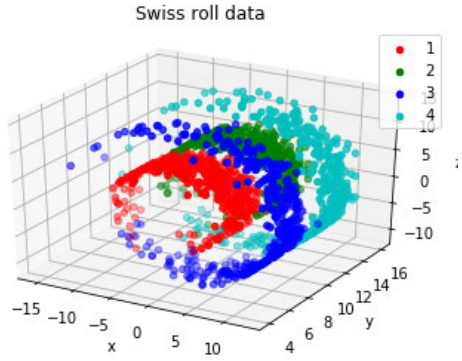
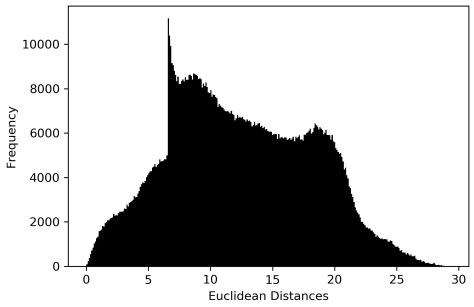


FIGURE 5. Euclidean distance distribution of Breast Cancer dataset.



(a)



(b)

FIGURE 6. Swiss Roll data (a) and its Euclidean distance distribution (b).

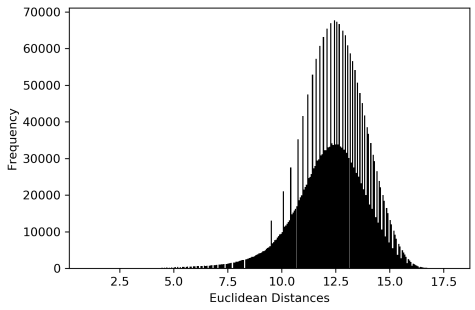


FIGURE 7. Euclidean distance distribution of MNIST data.

useful than *t*-SNE, LLE, and UMAP, TriMap in both, higher structure maintaining and less computational time.

C. SWISS ROLL

Swiss Roll data with 1600 samples and 3 attributes is shown in Fig. 6(a) and its distance distribution is shown in Fig. 6(b),

TABLE 1. The performance of methods (rows) in datasets (columns) in terms of Kendall's Tau coefficient and computational time.

		Datasets (number of attributes (dimensions) is the original space)				
		Iris (4)	Breast Cancer (30)	Swiss Roll (3)	MNIST (784)	Make Blob (40)
	deg_{best}	8	10	1	1	1
	τ	**0.967347**	0.998122	**0.914619**	**0.606540**	0.572387
	t	14.23	278.41	1361.60	8194.18	778
MSDD	deg	8 and 9	10 and 11	1 and 2	1 and 2	1 and 2
	τ	0.967309	**0.998125**	0.914541	0.598577	0.510645
	t	4.06	34.22	167.140	4177.75	44.91
	deg	7 and 8	9 and 10	0 and 1	0 and 1	0 and 1
	τ	0.967316	**0.998125**	0.914574	0.600533	0.503434
	t	3.04	25.82	193.64	3080.225	36.76
	deg	7,8 and 9	9,10 and 11	0, 1 and 2	0, 1 and 2	0,1 and 2
	τ	0.967334	0.998122	0.914541	0.598738	0.513616
	t	7.61	44	250	8086	50
MDS	τ	0.956922	0.997012	0.904167	0.505223	0.593932
	t	1.26	87	194	5387	42
PCA	τ	0.962634	0.997255	0.911537	0.369035	0.580076
	t	0.35	0.21	0.67	4	0.5
Isomap	k	146	515	1447	2222	94
	τ	0.962676	0.997687	0.912133	0.469043	0.599908
	t	2.38	243	15855	77957	608
LLE	k	32	556	975	5	487
	τ	0.681920	0.972837	0.857109	0.243375	0.560708
	t	7	1524	118088	114478	906
LE	k	65	426	1000	1972	290
	τ	0.646008	0.726745	0.812276	0.523077	0.602445
	t	6.72	190	4698	42696	127
***t*-SNE**	pr	135	501	1507	2216	168
	τ	0.925155	0.815006	0.868323	0.549509	**0.651567**
	t	263	5952	75437	295591	86351
UMAP	k	65	5	3	1384	153
	τ	0.871688	0.709309	0.042234	0.307098	0.400109
	t	554	6105	58097	236665	8037
TriMap	k	146	1	12	194	25
	τ	0.852019	0.693643	0.464367	0.364343	0.551976
	t	992	3888	10385	17508	6695

is the third dataset considered. By examining the co-ranking matrixes in Fig. 11 and Fig. 12, it can be seen that MSDD (*deg*: 1), PCA and Isomap performed better than the other methods in preserving the data structure. More specifically, MSDD (*deg*: 1) produced the highest τ was MSDD (*deg*: 1) of 0.914619 followed by Isomap and PCA with τ of 0.912133 and 0.911537, respectively, as shown in Table 1. Although MSDD was more expensive than two linear dimensionality reduction methods PCA and MDS, it performed better than *t*-SNE, Isomap, LE, LLE, TriMap, and UMAP

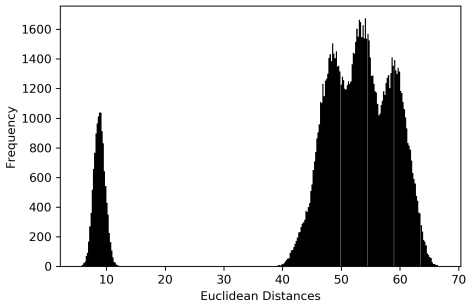


FIGURE 8. Euclidean distance distribution of Make Blob data.

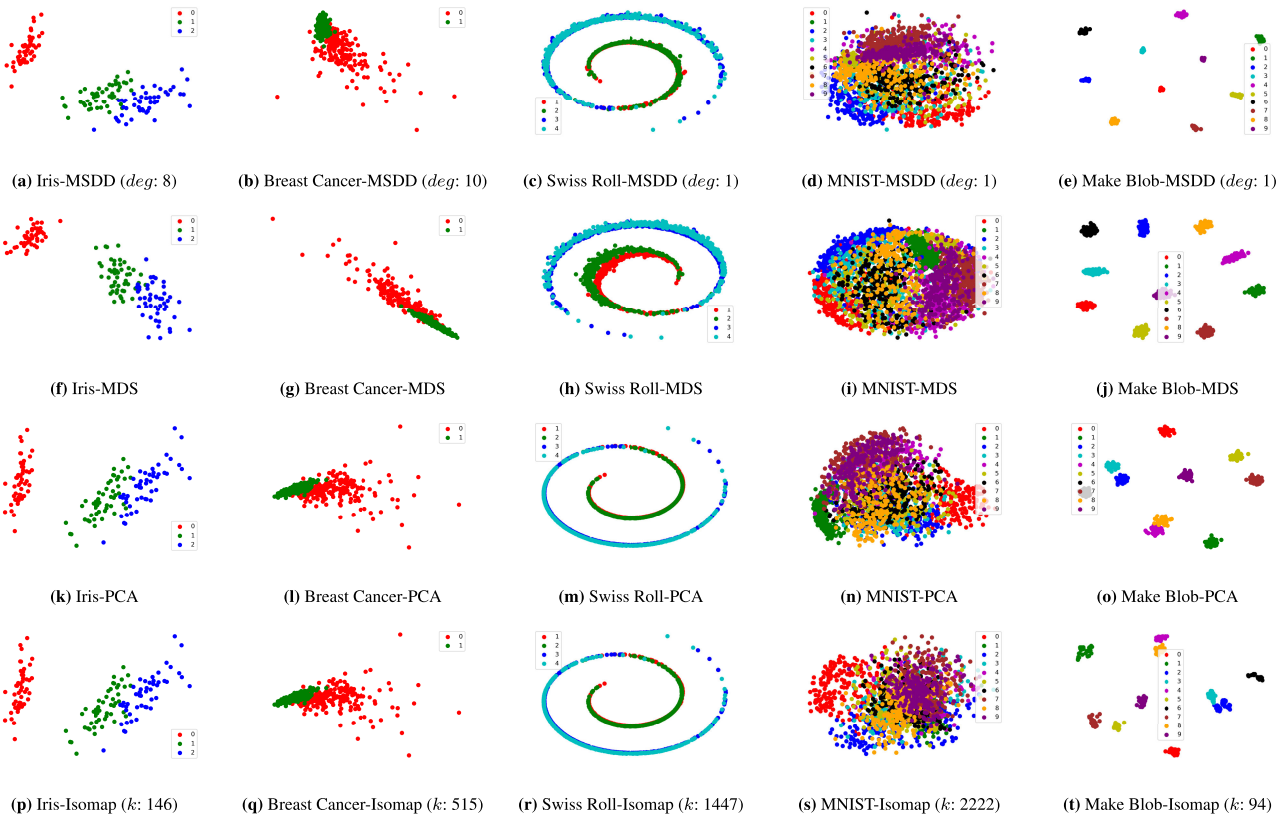
in terms of structure maintaining and computational time, as shown in Table 1.

D. MNIST

MNIST with 2500 samples and 784 attributes is the fourth dataset considered, with distance distribution shown in Fig. 7, dominated by entries with medium, large distances. MSDD (*deg*: 1) achieved the highest τ of 0.606540, followed by *t*-SNE (0.549509) and LE (0.523077), as shown in Table 1. As we can see, MSDD hugely provided better structure preservation over the other considered methods. Furthermore, MSDD was less expensive in relation to computational time compared with Isomap, *t*-SNE, UMAP, Trimap LE, and LLE. Although PCA and MDS were faster than MSDD, their performances in terms of τ were low. Therefore, the usage of MSDD has been beneficial with MNIST dataset in terms of both structure maintaining and computational time.

E. MAKE BLOB

Make Blob with 40 attributes and with distance distribution as shown in Fig. 8 is the last dataset considered. As we can see, the majority of samples have a distance of between 40 to 70, whereas a small fraction has a distance around 10. As we can also see from Table 1, MSDD (*deg*: 1) has been less useful in maintaining the data structure evaluated by a τ of 0.572387. The main cause of that has been the largest fraction of data located in the tail sections of degree-distributions. The method that performed the best in this dataset was *t*-SNE with the highest τ of 0.651567 shown in Table 1 and its respective co-ranking matrix demonstrated in Fig. 11 and Fig. 12, which illustrated fewer off-diagonal entries than other methods. However, *t*-SNE and LE, although they had the highest performances, were the most expensive methods. Furthermore, it should be noted that generating the best representative low dimensional data in terms of structure maintaining relies on the number of neighbours for methods



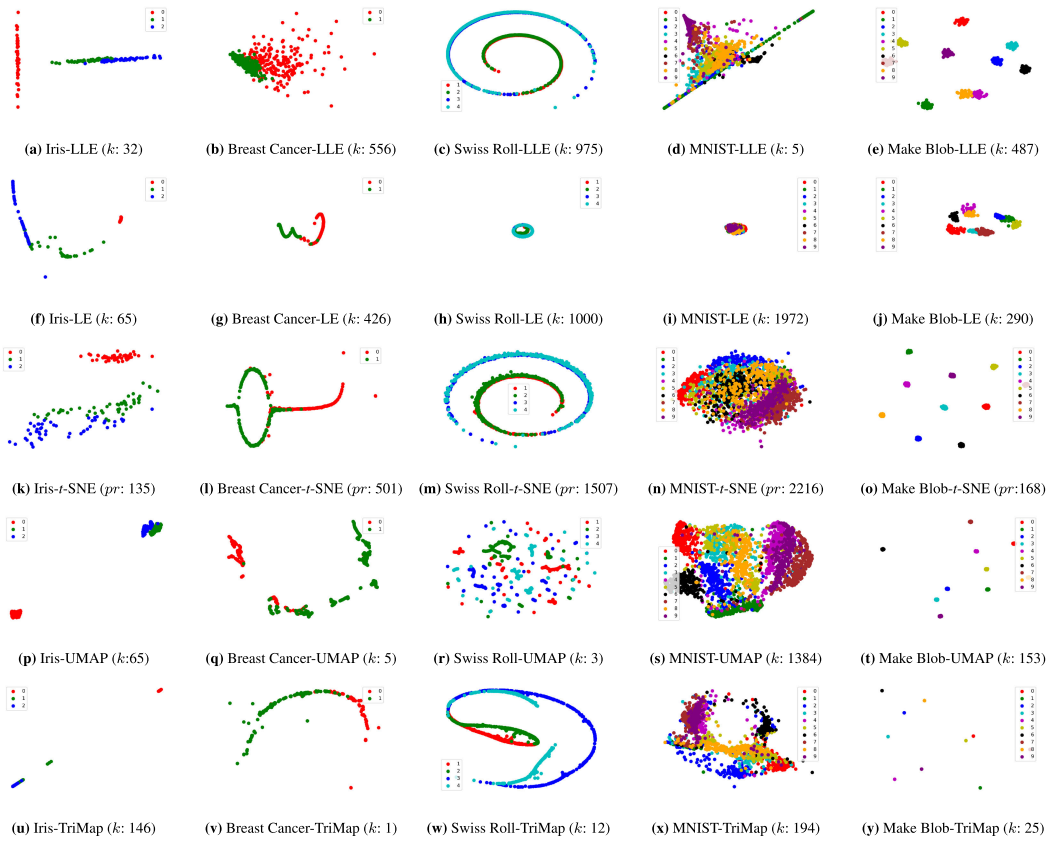
(a) Iris-MSDD (*deg*: 8) **(b)** Breast Cancer-MSDD (*deg*: 10) **(c)** Swiss Roll-MSDD (*deg*: 1) **(d)** MNIST-MSDD (*deg*: 1) **(e)** Make Blob-MSDD (*deg*: 1)

(f) Iris-MDS **(g)** Breast Cancer-MDS **(h)** Swiss Roll-MDS **(i)** MNIST-MDS **(j)** Make Blob-MDS

(k) Iris-PCA **(l)** Breast Cancer-PCA **(m)** Swiss Roll-PCA **(n)** MNIST-PCA **(o)** Make Blob-PCA

(p) Iris-Isomap (k: 146) **(q)** Breast Cancer-Isomap (k: 515) **(r)** Swiss Roll-Isomap (k: 1447) **(s)** MNIST-Isomap (k: 2222) **(t)** Make Blob-Isomap (k: 94)

FIGURE 9. The visualisation of the two-dimensional representation of the Iris (4 attributes), Breast Cancer (30 attributes), Swiss Roll (3 attributes), MNIST (784 attributes) and Make Blob (40 attributes) generated by MSDD, MDS, PCA, and Isomap.



(a) Iris-LLE (k: 32) **(b)** Breast Cancer-LLE (k: 556) **(c)** Swiss Roll-LLE (k: 975) **(d)** MNIST-LLE (k: 5) **(e)** Make Blob-LLE (k: 487)

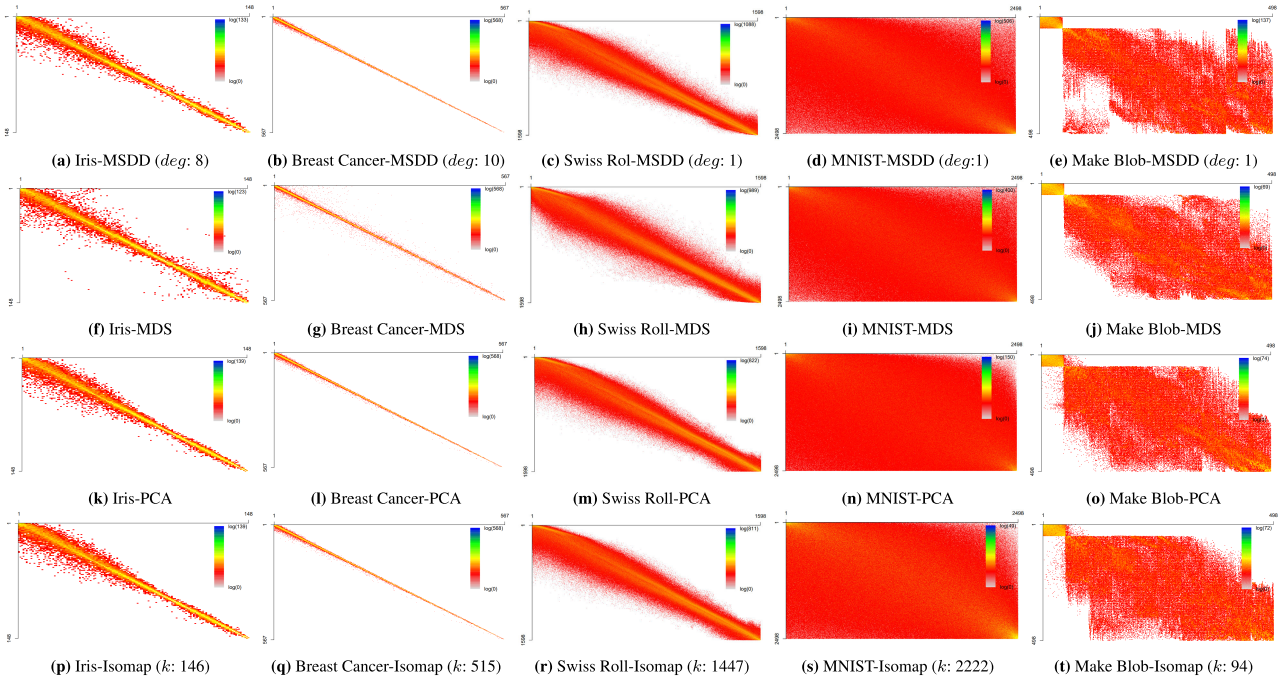
(f) Iris-LE (k: 65) **(g)** Breast Cancer-LE (k: 426) **(h)** Swiss Roll-LE (k: 1000) **(i)** MNIST-LE (k: 1972) **(j)** Make Blob-LE (k: 290)

(k) Iris-t-SNE (pr: 135) **(l)** Breast Cancer-t-SNE (pr: 501) **(m)** Swiss Roll-t-SNE (pr: 1507) **(n)** MNIST-t-SNE (pr: 2216) **(o)** Make Blob-t-SNE (pr:168)

(p) Iris-UMAP (k:65) **(q)** Breast Cancer-UMAP (k: 5) **(r)** Swiss Roll-UMAP (k: 3) **(s)** MNIST-UMAP (k: 1384) **(t)** Make Blob-UMAP (k: 153)

(u) Iris-TriMap (k: 146) **(v)** Breast Cancer-TriMap (k: 1) **(w)** Swiss Roll-TriMap (k: 12) **(x)** MNIST-TriMap (k: 194) **(y)** Make Blob-TriMap (k: 25)

FIGURE 10. The visualisation of two-dimensional representation of the Iris (4 attributes), Breast Cancer (30 attributes), Swiss Roll (3 attributes), MNIST (784 attributes) and Make Blob (40 attributes) generated by LLE, LE, t-SNE, UMAP, and TriMap.



(a) Iris-MSDD (deg: 8) **(b)** Breast Cancer-MSDD (deg: 10) **(c)** Swiss Rol-MSDD (deg: 1) **(d)** MNIST-MSDD (deg:1) **(e)** Make Blob-MSDD (deg: 1)

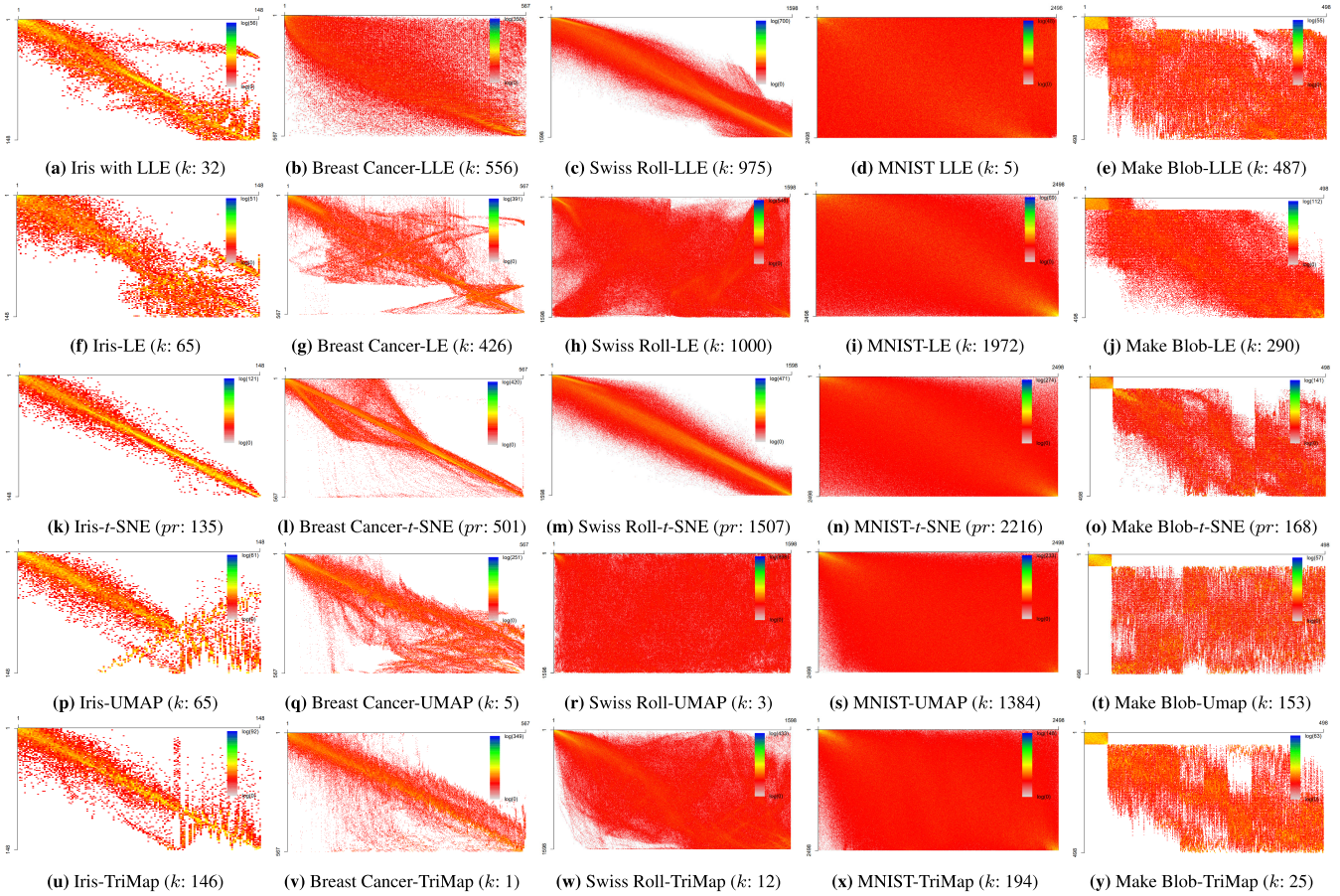
(f) Iris-MDS **(g)** Breast Cancer-MDS **(h)** Swiss Roll-MDS **(i)** MNIST-MDS **(j)** Make Blob-MDS

(k) Iris-PCA **(l)** Breast Cancer-PCA **(m)** Swiss Roll-PCA **(n)** MNIST-PCA **(o)** Make Blob-PCA

(p) Iris-Isomap (k: 146) **(q)** Breast Cancer-Isomap (k: 515) **(r)** Swiss Roll-Isomap (k: 1447) **(s)** MNIST-Isomap (k: 2222) **(t)** Make Blob-Isomap (k: 94)

FIGURE 11. The co-ranking matrixes of the Iris (4 attributes), Breast Cancer (30 attributes), Swiss Roll (3 attributes), MNIST (784 attributes) and Make Blob (40 attributes) generated by MSDD, MDS, PCA, and Isomap.

such as Isomap, LLE, LE, t-SNE, UMAP, and TriMap. As a result, these methods challenged by data with a large number of samples, because they require tuning the number of neighbours from 1 to $N-1$, but to tune the degree on SDD takes a maximum of 15 steps. Although MSDD is more expensive than SDD, its computational complexity does not



(a) Iris with LLE (k: 32) **(b)** Breast Cancer-LLE (k: 556) **(c)** Swiss Roll-LLE (k: 975) **(d)** MNIST LLE (k: 5) **(e)** Make Blob-LLE (k: 487)

(f) Iris-LE (k: 65) **(g)** Breast Cancer-LE (k: 426) **(h)** Swiss Roll-LE (k: 1000) **(i)** MNIST-LE (k: 1972) **(j)** Make Blob-LE (k: 290)

(k) Iris-t-SNE (pr: 135) **(l)** Breast Cancer-t-SNE (pr: 501) **(m)** Swiss Roll-t-SNE (pr: 1507) **(n)** MNIST-t-SNE (pr: 2216) **(o)** Make Blob-t-SNE (pr: 168)

(p) Iris-UMAP (k: 65) **(q)** Breast Cancer-UMAP (k: 5) **(r)** Swiss Roll-UMAP (k: 3) **(s)** MNIST-UMAP (k: 1384) **(t)** Make Blob-Umap (k: 153)

(u) Iris-TriMap (k: 146) **(v)** Breast Cancer-TriMap (k: 1) **(w)** Swiss Roll-TriMap (k: 12) **(x)** MNIST-TriMap (k: 194) **(y)** Make Blob-TriMap (k: 25)

FIGURE 12. The co-ranking matrixes of the Iris (4 attributes), Breast Cancer (30 attributes), Swiss Roll (3 attributes), MNIST (784 attributes) and Make Blob (40 attributes) generated by LLE, LE, *t*-SNE, UMAP, and TriMap.

significantly increase if following the guidance provided in Section IV.

VI. CONCLUSION AND FURTHER WORK

This paper proposes SDD and MSDD for the dimensionality reduction of data to better preserve the data structure using less computational time compared to other manifold learning methods. SDD employs one degree-distribution, whereas MSDD adds various degree-distributions on top of SDD, aiming to improve the data structure capturing. Due to the high sensitivity of degree-distribution(s) in small and medium distance sections, SDD (MSDD) can usefully capture the structure of data with a large fraction of small and medium distances. Conversely, it performs less favorably in datasets with a large fraction of large distances, due to the large number of samples placed in the low sensitivity section(s) (tail(s)) of degree-distribution(s). Overall, SDD (MSDD) outperforms in terms of structure capturing benchmarks methods such as t-SNE, UMAP, Isomap, PCA, MDS, Trimap, LLE, and LE in data which dominates by small and medium distances. Additionally, the structure capturing of SDD (MSDD) does not rely on the number of neighbours, but it instead tunes the degree of degree-distribution, which ranges from 1 to 15 instead of 1 to $N-1$. As a result, SDD (MSDD) can be more useful than other manifold learning techniques to reduce the data dimensionality of datasets having a large number of samples.

In the experiments conducted, employing one degree-distribution has produced the best low dimensional data representation in terms of structure maintaining. The addition of a degree below or above the best degree has resulted in the deterioration of the maintained data structure. For Breast Cancer, where the best result was achieved by the combination of two degree-distributions, the improvement was not notable. In conclusion, we suggest that using one degree-distribution can be efficient in capturing data structure. However, if preserving the data structure is crucial, then we suggest adding more degree-distributions on top of the best degree degree-distribution.

For further work, the authors aim to approximate the number of degree-distributions and their degrees in relation to the data. Reducing the computational complexity is another objective for further work.

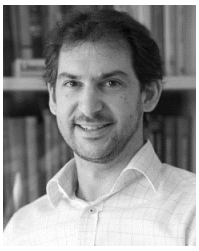
REFERENCES

[1] L. Rui and H. Nejati, "Dimensionality reduction of brain imaging data using graph signal processing," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1329–1333.

[2] S. Zahorian and H. Hu, "Nonlinear dimensionality reduction methods for use with automatic speech recognition," in *Speech Technologies*. London, U.K.: IntechOpen, 2011, pp. 55–78.
[3] S. Sharma, M. Kumar, and P. K. Das, "A technique for dimension reduction of MFCC spectral features for speech recognition," in *Proc. Int. Conf. Ind. Instrum. Control (ICIC)*, Pune, India, May 2015, pp. 99–104, doi: 10.1109/IIC.2015.7150719.
[4] M. Chamberland, E. P. Raven, S. Genc, K. Duffy, M. Descoteaux, G. D. Parker, C. M. W. Tax, and D. K. Jones, "Dimensionality reduction of diffusion MRI measures for improved tractometry of the human brain," *NeuroImage*, vol. 200, pp. 89–100, Oct. 2019.
[5] M. Beyeler, E. L. Rounds, K. D. Carlson, N. Dutt, and J. L. Krichmar, "Neural correlates of sparse coding and dimensionality reduction," *PLOS Comput. Biol.*, vol. 15, no. 6, Jun. 2019, Art. no. e1006908.
[6] S. Ji, "Computational genetic neuroanatomy of the developing mouse brain: Dimensionality reduction, visualization, and clustering," *BMC Bioinf.*, vol. 14, no. 1, p. 222, Dec. 2013, doi: 10.1186/1471-2105-14-222.
[7] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, R. Kaluri, D. S. Rajput, G. Srivastava, and T. Baker, "Analysis of dimensionality reduction techniques on big data," *IEEE Access*, vol. 8, pp. 54776–54788, 2020, doi: 10.1109/ACCESS.2020.2980942.
[8] F. S. Tsai, "Dimensionality reduction techniques for blog visualization," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 2766–2773, Mar. 2011.
[9] C. E. Gutierrez, M. R. Alsharif, H. Cuiwei, M. Khosravy, R. Villa, K. Yamashita, and H. Miyagi, "Uncover news dynamic by principal component analysis," *ICIC Exp. Lett.*, vol. 7, no. 4, pp. 1245–1250, 2013.
[10] M. Alkhayrat, M. Aljnidi, and K. Aljoumaa, "A comparative dimensionality reduction study in telecom customer segmentation using deep learning and PCA," *J. Big Data*, vol. 7, no. 1, p. 9, Dec. 2020.
[11] M. A. Belarbi, S. Mahmoudi, and G. Belalem, "PCA as dimensionality reduction for large-scale image retrieval systems," *Int. J. Ambient Comput. Intell.*, vol. 8, no. 4, pp. 45–58, Oct. 2017.
[12] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *J. Educ. Psychol.*, vol. 24, no. 6, p. 417, 1933.
[13] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, Mar. 1964.
[14] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Trans. Comput.*, vol. COM-18, no. 5, pp. 401–409, May 1969.
[15] G. Zhong, L.-N. Wang, X. Ling, and J. Dong, "An overview on data representation learning: From traditional feature learning to recent deep learning," *J. Finance Data Sci.*, vol. 2, no. 4, pp. 265–278, Dec. 2016.
[16] J. B. Tenenbaum, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
[17] S. T. Roweis, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.
[18] M. M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 585–591.
[19] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, Jun. 2003.
[20] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
[21] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," 2018, *arXiv:1802.03426*. [Online]. Available: http://arxiv.org/abs/1802.03426
[22] E. Amid and M. K. Warmuth, "A more globally accurate dimensionality reduction method using triplets," 2018, *arXiv:1803.00854*. [Online]. Available: http://arxiv.org/abs/1803.00854
[23] J. Isenmann, "Modern multivariate statistical techniques," in *Regression, Classification and Manifold Learning*. New York, NY, USA: Springer, 2008.
[24] Y. Zhou and T. Sharpee, "Using global t-SNE to preserve inter-cluster data structure," 2018, *bioRxiv:331611*. [Online]. Available: https://www.biorxiv.org/content/10.1101/331611v2.abstract
[25] J. A. Lee, D. H. Peluffo-Ordez, and M. Verleysen, "Multi-scale similarities in stochastic neighbour embedding: Reducing dimensionality while preserving both local and global structure," *Neurocomputing*, vol. 169, pp. 26–246, Dec. 2015.
[26] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 857–864.
[27] G. E. Hinton, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.
[28] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.
[29] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 153–160.
[30] R. Salakhutdinov and G. E. Hinton, "Deep Boltzmann machines," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2009, pp. 448–455.
[31] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep Boltzmann machines," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 693–700.
[32] J. Vrábel, P. Pořízka, and J. Kaiser, "Restricted Boltzmann Machine method for dimensionality reduction of large spectroscopic data," *Spectrochimica Acta B, At. Spectrosc.*, vol. 167, May 2020, Art. no. 105849.
[33] D. Chen, J. Lv, and Z. Yi, "Graph regularized restricted bolzmann machine," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2651–2659, Jun. 2018.
[34] K. Miettinen, *Nonlinear Multiobjective Optimization*. Cham, Switzerland: Springer, 2012.
[35] M. T. M. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: Fundamentals and evolutionary methods," *Natural Comput.*, vol. 17, no. 3, pp. 585–609, Sep. 2018.
[36] J. A. Lee and M. Verleysen, "Quality assessment of dimensionality reduction: Rank-based criteria," *Neurocomputing*, vol. 72, nos. 7–9, pp. 1431–1443, Mar. 2009.

LAURETA HAJDERANJ received the M.Sc. degree in mathematics and informatics engineering from the University of Tirana, Tirane, Albania, in 2012, and the M.Sc. degree in internet and database systems from London South Bank University, in 2018, where she is currently pursuing the Ph.D. degree in computer science. From 2013 to 2015, she worked as a Lecturer with the Computer Science Department, University of "Aleksandër Moisiu," Durrës, Albania. Her research interests include high-dimensional data embedding and visualization, structure capturing, and acceleration algorithms.

DAQING CHEN (Member, IEEE) received the bachelor's degree in systems engineering from Northwestern Polytechnical University, Xi'an, China, in 1982, the master's degree in automatics control engineering from the National University of Defense Technology, Changsha, China, in 1990, and the Ph.D. degree in automatics control engineering from Northwestern Polytechnical University, in 1993. From 1994 to 1997, he worked as a Postdoctoral Researcher and then an Associate Professor with the National Key Laboratory of Radar Signal Processing, Xidian University, Xi'an. From 1997 to 1998, he was a Research Associate with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. From 1998 to 1999, he worked as a Research Fellow with the System, Electronics and Information Laboratory, IRESTE, University of Nantes, Nantes, France. Since 1999, he has been working with London South Bank University, where he is currently a Senior Lecturer of Informatics with the School of Engineering. His research interests include deep learning algorithms with applications in lip-reading, medical image diagnosis, high dimensional data embedding and visualization, high-volume data labeling, and business intelligence.



ENRICO GRISAN (Senior Member, IEEE) received the M.Sc. degree in electrical engineering from the University of Padua, Padua, Italy, in 2000, and the joint Ph.D. degree in bioengineering from the University of Padua and City University, London, U.K., in 2004, defending a thesis on automatic analysis of retinal images. In 2005, he was an Intern with Siemens Corporate Research, Princeton, NJ, USA, and then a Postdoctoral Fellow with the University of Padua, where he has been an Assistant Professor of Biomedical Engineering, since 2008. In 2019, he joined London South Bank University, as a Lecturer in Artificial Intelligence. His current research interests include the understanding of medical imaged and identification of relevant biomarkers from medical data, either through classical image processing and analysis or through machine learning, with applications to neuroimaging, confocal microscopy and microendoscopy, and ultrasound. He has served as an Associate Editor for the IEEE ISBI and the IEEE EMBC conferences, and as the General Chair for IEEE ISBI 2019. He has been a member of the IEEE Technical Committee in Biomedical Imaging and Image Processing, since 2015.

SANDRA DUDLEY (Member, IEEE) received the Ph.D. degree in physics from the University of Essex, U.K., in 2004. She is currently a Professor of Communication systems and the Director of Research with the School of Engineering. Her research interests include low power and remote sensing schemes has led to adaptive optical-wireless systems research and the development of smart strategies for inherent physical networks, in particular a world record with BT research on lowest power broadband systems for last mile access broadband systems. She investigates areas ranging from wireless sensor networks, remote sensing, non-wearable technology, and imaging. Additionally, she also carries out research in data processing of the signals from such systems aiming toward complete platforms ready for upscaling. She manages Ph.D.'s and Research Associates in the above areas. She collaborates and leads on a number of U.K., EU, and IUK research grants with applications in remote user monitoring and data processing.