mathematics

MDPI

Article

Comparative Analysis of Manifold Learning-Based Dimension Reduction Methods: A Mathematical Perspective

Wenting Yi [1], Siqi Bu [1,2,*], Hiu-Hung Lee [1,*] and Chun-Hung Chan [1]

[1] Centre for Advances in Reliability and Safety (CAiRS), Hong Kong SAR 999077, China; wenting.yi@cairs.hk (W.Y.); howard.chan@cairs.hk (C.-H.C.)

[2] Department of Electrical and Electronic Engineering, The Hong Kong Polytechnic University, Hong Kong SAR 999077, China

* Correspondence: siqi.bu@polyu.edu.hk (S.B.); rainbow.lee@cairs.hk (H.-H.L.)

Abstract: Manifold learning-based approaches have emerged as prominent techniques for dimensionality reduction. Among these methods, t-Distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP) stand out as two of the most widely used and effective approaches. While both methods share similar underlying procedures, empirical observations indicate two distinctive properties: global data structure preservation and computational efficiency. However, the underlying mathematical principles behind these distinctions remain elusive. To address this gap, this study presents a comparative analysis of the subprocesses involved in these methods, aiming to elucidate the mathematical mechanisms underlying the observed distinctions. By meticulously examining the equation formulations, the mathematical mechanisms contributing to global data structure preservation and computational efficiency are elucidated. To validate the theoretical analysis, data are collected through a laboratory experiment, and an open-source dataset is utilized for validation across different datasets. The consistent alignment of results obtained from both balanced and unbalanced datasets robustly confirms the study's findings. The insights gained from this study provide a deeper understanding of the mathematical underpinnings of t-SNE and UMAP, enabling more informed and effective use of these dimensionality reduction techniques in various applications, such as anomaly detection, natural language processing, and bioinformatics.

Keywords: manifold learning; dimension reduction; spectral embedding; fuzzy topology; stochastic gradient descent

MSC: 68Q25; 68W40; 68T09; 68T20; 68T01

check for updates

Citation: Yi, W.; Bu, S.; Lee, H.-H.; Chan, C.-H. Comparative Analysis of Manifold Learning-Based Dimension Reduction Methods: A Mathematical Perspective. *Mathematics* **2024**, *12*, 2388. https://doi.org/10.3390/math12152388

Academic Editor: Antanas Cenys

Received: 14 June 2024
Revised: 25 July 2024
Accepted: 29 July 2024
Published: 31 July 2024

1. Introduction

Dimension reduction techniques aim to reduce the dimensionality of high-dimensional data while preserving its essential structure [1–3]. These techniques are generally categorized into linear and nonlinear methods. Linear techniques, like Principal Component Analysis (PCA), project data onto a lower-dimensional subspace by minimizing variance [4,5]. While PCA is computationally efficient, it assumes linearity in the data, limiting its ability to capture complex nonlinear structures. To address this limitation, nonlinear manifold learning techniques, such as t-Distributed Stochastic Neighbor Embedding (t-SNE) [6–8] and Uniform Manifold Approximation and Projection (UMAP) [9–11], have gained prominence. These methods uncover the underlying nonlinear structure of data by constructing low-dimensional representations that preserve intrinsic relationships. Manifold learning techniques are particularly useful in industrial applications like mechanical structure fault detection and anomaly analysis, where high-dimensional data can be efficiently processed and analyzed for improved decision-making [12,13].

t-SNE is a widely used manifold learning technique for dimension reduction and visualization [7,14]. It reveals the structure of the data by creating low-dimensional representations that preserve similarity relationships between data points. This is achieved by constructing probability distributions over pairs of high-dimensional data points and their low-dimensional counterparts, then minimizing the divergence between these distributions. t-SNE excels in retaining local structures and extracting nonlinear patterns, making it valuable in fields such as image analysis [15,16], predictive maintenance [17,18], and bioinformatics [19–22]. However, t-SNE's computational demands and sensitivity to hyperparameters can limit its scalability and reliability for large datasets.

UMAP, another popular manifold learning technique, shares a common objective with t-SNE. Both methods aim to uncover the underlying structure by creating a low-dimensional representation that preserves data point relationships [9,23]. However, UMAP distinguishes itself by constructing a fuzzy topological representation of the data, approximating the manifold structure. By utilizing a combination of local and global optimization objectives, UMAP achieves an embedding that effectively balances the preservation of both local and global structures. While t-SNE and UMAP share similar goals, UMAP presents several distinct advantages [24,25]. Firstly, UMAP demonstrates faster computation and superior scalability, making it highly efficient for handling larger datasets. Additionally, UMAP provides greater flexibility for parameter tuning, granting increased control over the resulting embedding [26,27]. Furthermore, UMAP outperforms in preserving the global structure of the data, resulting in more accurate and meaningful visualizations. Despite its advantages over t-SNE, the underlying principles of UMAP's performance are not fully understood, warranting further investigation.

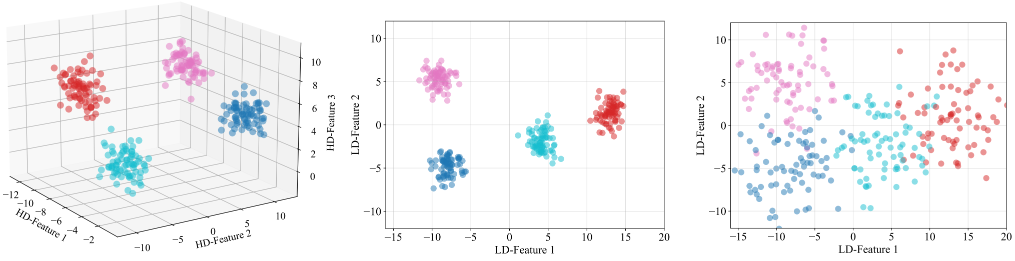
Understanding the mechanisms of manifold learning methods is crucial for their effective use. The main motivation behind choosing t-SNE and UMAP lies in their similar algorithmic structures and objectives, yet distinct mathematical formulas and principles. By comparing these two techniques, valuable insights into their strengths and limitations can be obtained, aiding in method selection, parameter optimization, and results interpretation. This comparative analysis is expected to identify potential enhancements for these techniques and improve their application across various domains. Specifically, for UMAP, which possesses a robust and intricate mathematical foundation, a comprehensive exploration and comprehension of its mechanisms from a mathematical perspective is essential. Despite ongoing efforts, a comprehensive interpretation of these methods remains elusive.

To address the aforementioned limitations, this paper presents a comprehensive comparative analysis of t-SNE and UMAP from a mathematical perspective. The contributions of this paper are as follows:

- The algorithmic mechanisms of both t-SNE and UMAP are firstly systematically deconstructed into five key subprocesses: the high-dimensional probability function [28,29], the low-dimensional probability function [30], the spectral embedding [31], the loss function [32], and the optimization process [33]. This deliberate deconstruction enables a detailed and comprehensive comparison analysis, and serves as the foundation for a thorough examination of the execution process and functional results.
- Through comprehensive analysis of the mathematical formulas and the distinctions between the subprocesses of t-SNE and UMAP, the reasons behind UMAP's ability to preserve the global structure of the data and achieve computational efficiency are clearly revealed and presented from a mathematical perspective.
- A lab experiment designed to mimic real-life situations is conducted, and the resulting dataset is made available for further research to facilitate validation and verification.
- A comprehensive blade study is performed to assess the impact of different UMAP subprocesses on computational time and accuracy. Statistical quantitative results are obtained and presented to provide evidence-based validations.
- A detailed discussion on the application of t-SNE and UMAP across various scenarios is provided. Based on the revealed mathematical principles, scenario-specific guidance is presented to inform the optimal selection of these dimensionality reduction methods.

2. Problem Statement

As indicated in the introduction, manifold learning techniques such as t-SNE and UMAP offer significant advantages over traditional linear methods like PCA when dealing with high-dimensional data that exhibit complex nonlinear structures. Linear methods often fail to capture these intricate relationships, leading to the distortion of essential data structures during dimensionality reduction. Consequently, manifold learning techniques are crucial for preserving the integrity of data with nonlinear patterns, making them indispensable for accurate analysis and subsequent applications. This concept is further illustrated in Figure 1 for better understanding.



(**a**) High-dimensional data (**b**) Low-dimensional data (ML) (**c**) Low-dimensional data (non-ML)

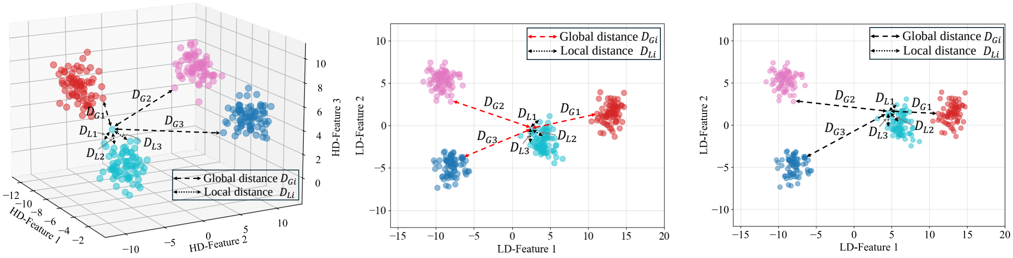
Figure 1. Illustration of high-dimensional data and low-dimensional data with manifold learning (ML) and non-manifold learning (non-ML) techniques. Four colors represent four different clusters.

Figure 1a illustrates a high-dimensional dataset (represented as 3D data) with nonlinear structures. When manifold learning techniques are applied, the low-dimensional projection (2D in this case) effectively preserves the clusters present in the original high-dimensional data, as shown in Figure 1b. In contrast, non-manifold learning techniques can distort these clusters, as depicted in Figure 1c, leading to the loss of critical information embedded in the data. This distortion can result in erroneous conclusions and ineffective applications. Therefore, manifold learning techniques are essential for maintaining data integrity during dimensionality reduction, particularly when dealing with datasets that have inherent nonlinearities.

Given the importance of manifold learning techniques in preserving data structure, it is essential to understand the specific capabilities and limitations of different methods within this category. t-SNE and UMAP are two prominent manifold learning techniques that, despite their similarities in adopting nonlinear transformations and preserving local neighborhood relationships, exhibit differences in their performance and application.

- Firstly, t-SNE and UMAP differ in their ability to preserve global structures in the data. While t-SNE is effective at preserving local relationships, it often fails to maintain global distances, which can result in a misrepresentation of the overall data structure in the low-dimensional space. In contrast, UMAP is designed to preserve both local and global structures, making it more robust in capturing the true nature of the data. This concept is further illustrated in Figure 2 for better understanding: Figure 2a shows the global and local distances in high-dimensional space (3D data), with global distances marked by dashed lines (D_{Gi} such as D_{G1}, D_{G2}, D_{G3}) and local distances by dotted lines (D_{Li} such as D_{L1}, D_{L2}, D_{L3}). Figure 2b demonstrates that t-SNE preserves local distances in the low-dimensional space (2D), i.e., D_{Li} in high dimension is equal to D_{Li} in low dimension, but fails to maintain global distances, i.e., D_{Gi} in high dimension does not equal to D_{Gi} in low dimension (marked with a red dashed line). Conversely, UMAP effectively preserves both local and global distances, i.e., all pairs of D_{Gi} and D_{Li} are equal in high and low dimensions.
- Secondly, computational efficiency is a crucial consideration, particularly when managing large-scale datasets. t-SNE is recognized for its computational inefficiency, which constrains its scalability and limits its applicability to large datasets. In contrast,

UMAP demonstrates superior computational efficiency, rendering it more suitable for processing and analyzing extensive datasets effectively.



(**a**) High-dimensional data (**b**) Low-dimensional data (t-SNE) (**c**) Low-dimensional data (UMAP)

Figure 2. Illustration of global and local structures in high dimension and low dimension with t-SNE and UMAP. Four colors represent four different clusters.

Despite recognizing the differences between t-SNE and UMAP, a comprehensive mathematical understanding of the underlying causes remains insufficiently explored. Existing research has predominantly been based on empirical observations without a solid theoretical foundation. To address this gap, this study aims to provide an in-depth comparative analysis and theoretical investigation of t-SNE and UMAP. By elucidating the fundamental factors that contribute to their divergent behaviors, this research seeks to enhance the theoretical understanding of these techniques. Through a detailed examination grounded in mathematical principles, this study provides deeper insights into their respective strengths and limitations. This understanding will facilitate more informed decisions regarding method selection, parameter optimization, and application across various domains.

3. Mathematical Comparison of Two Manifold Learning-Based Methods

3.1. Mathematical Notations and Symbols

To enhance the readability of mathematical derivation and illustration, the mathematical notations and symbols used in the following sections are given in the following Table 1.

The general algorithmic structure for both t-SNE and UMAP is illustrated on the left side of Figure 3. The primary steps of both algorithms can be summarized as follows: (1) calculate similarity scores in the high-dimensional space; (2) initialize the low-dimensional embedding; (3) calculate similarity scores in the low-dimensional space; and (4) assess whether the discrepancies between the high-dimensional and low-dimensional similarity scores are below a specified threshold. If the errors are within the threshold, the process ends; if not, adjust the positions of the low-dimensional points to minimize the error, and repeat the iteration.

While these steps outline the algorithmic workflow, each step is underpinned by specific mathematical formulas that are crucial to the algorithm's performance and functionality. To provide a thorough understanding of these techniques, this study decomposes the algorithm into distinct subprocesses for detailed mathematical analysis:

- **Subprocess 1** : design a probabilistic function to model the structural patterns between pairs of high-dimensional data points.
- **Subprocess 2**: design an initialization function for assigning initial low-dimensional coordinates.
- **Subprocess 3**: design a probabilistic function to model the structural patterns between pairs of low-dimensional data points.
- **Subprocess 4**: design a loss function to minimize the discrepancy, guiding the model towards better performance by adjusting its parameters during the training process.
- **Subprocess 5**: design an optimization algorithm for updating the parameters and minimizing the dissimilarity between high-dimensional data points and their corresponding low-dimensional representations.

Table 1. Mathematical notations and symbols.

x_i	Data point i in high-dimension dataset
y_i	Data point i in low-dimension dataset
$p_{i\|j}$	Similarity score of data point x_i to data point x_j
$p_{j\|i}$	Similarity score of data point x_j to data point x_i
σ_i	The variance of the Gaussian process for node i
$d(x_i, x_j)$	Distance between data points x_i and x_j
ρ_i	Distance from x_i to its nearest neighbor
k	Preassigned parameter k in UMAP algorithm
$-\sum_j p_{j\|i} \log_2 p_{j\|i}$	Shannon entropy
$\hat{q}_{ij}$	Low-dimensional probability for data point y_i and y_j when applying t-SNE
$\left\\|y_i - y_j\right\\|^2$	Square of the distance between data point y_i and y_j in low dimensions.
$\tilde{q}_{ij}$	Low-dimensional probability for data point y_i and y_j when applying UMAP
α, β	Controllable parameters in UMAP
min_dist	Hyperparameter for determining α and β in UMAP
Ψq_{ij}	Probability calculated from non-linear least-square fitting
$\boldsymbol{A}, \boldsymbol{D}, \boldsymbol{L}$	Graph matrix, Degree matrix, Laplacian matrix
$\mathcal{C}_{KL}$	Kullback–Leibler loss function
$\mathcal{C}_{CE}$	Cross entropy loss function
s	1-simplex
$\mathcal{S}$	Set of all possible 1-simplices
$w_h(s)$	Weight of the 1-simplex s from high-dimensional manifold approximation
$w_l(s)$	Weight of the 1-simplex s to be discovered for low-dimensional representation
w_t	Parameter value at time t in gradient descent
η_t	Learning rate, the step size in each iteration
∇	Gradient vector of the function evaluated at the current parameter values
$i_t \in \mathcal{N}$	Random index selected from $\mathcal{N}$ with equal probability
z_{i_t}	Parameter for the stochastic mini-batch determined by i_t
ξ	Non-negative hyperparameter that controls the regularization strength
$d_{h,ij}$	High-dimensional distance between data points i and j
$d_{l,ij}$	Low-dimensional distance between data points i and j
$P(d_{h,ij})$	Joint probability for data points i and j in high dimension
$Q(d_{l,ij})$	Joint probability for data points i and j in low dimension

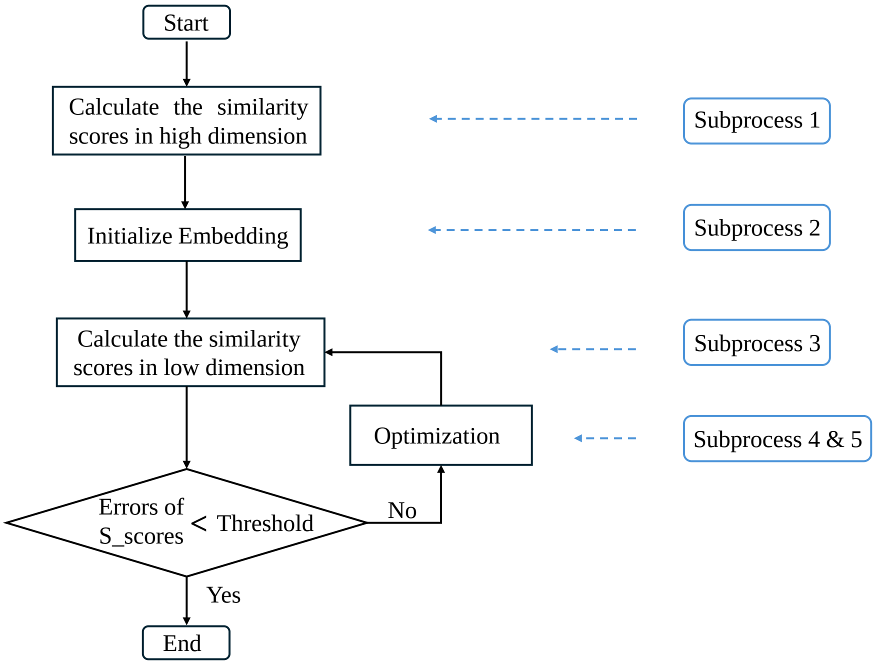

Start
Calculate the similarity scores in high dimension
Initialize Embedding
Calculate the similarity scores in low dimension
Optimization
Errors of S_scores < Threshold
No
Yes
End
Subprocess 1
Subprocess 2
Subprocess 3
Subprocess 4 & 5

Figure 3. The general algorithm structure of manifold learning techniques.

The relationships and dependencies between various subprocesses are presented on the right side of Figure 3 and are summarized as follows:

- The probabilistic function determined in **Subprocess 1** is used to calculate the similarity scores between each pair of points in the high-dimensional dataset.
- The initialization function determined in **Subprocess 2** is used for the embedding initialization process in the algorithm.
- The probabilistic function determined in **Subprocess 3** is used to calculate the similarity scores between each pair of points in the low-dimensional dataset.
- The loss function determined in **Subprocess 4**, along with the optimization algorithm in **Subprocess 5**, is used in the optimization process to minimize the errors in the similarity scores between high and low dimensions.

Each subprocess will be examined in the subsequent sections to elucidate the underlying principles and mechanisms.

3.2. Modeling the Fuzzy Topological Structure in High Dimension

Table 2 illustrates the modeling formulas for t-SNE (second column) and UMAP (third column). Both methods comprise three closely linked components: calculation of conditional probabilities, calculation of joint probabilities, and setting of controllable parameters for constraints. However, they utilize different formulas within each component, resulting in their distinct characteristics:

- ***Conditional probability calculation***: For t-SNE (Equation (A.1)), $p_{j|i}$ represents the similarity of data point x_j to data point x_i. It quantifies the conditional probability that x_i would select x_j as its neighbor if neighbors were chosen proportionally to their probability density under a Gaussian centered at x_j. The variance of the Gaussian process σ_i depends on the perplexity parameter (Equation (A.3)) [7]. In UMAP (Equation (B.1)), $d(x_i, x_j)$ represents the distance between data points x_i and x_j. The parameter ρ_i denotes the distance from x_i to its nearest neighbor and, like the perplexity parameter in t-SNE, σ_i for UMAP is determined by a preassigned parameter k (Equation (B.3)). Notably, the parameter ρ_i determines the local connectivity of the manifold, resulting in a locally adaptive exponential kernel for each data point.
- ***Joint probability calculation***: For t-SNE, the joint probability is calculated using a simple formula (Equation (A.2)) that satisfies the symmetry rule. In contrast, UMAP adopts a fuzzy union operation (Equation (B.2)) due to its theoretical framework, topology, and foundation in fuzzy sets. In other words, after the conditional probability calculation, UMAP applies Equation (B.2) to all data pairs, ensuring symmetrical representation through a fuzzy union approach rather than a simple average, as in t-SNE.
- ***Controllable parameter calculation***: For t-SNE, a binary search is performed to determine the value of σ that produces a probability distribution with a fixed perplexity, specified by the user (Equation (A.3))), where the exponential part $-\sum_j p_{j|i} \log_2 p_{j|i}$ is the Shannon entropy. This perplexity parameter serves as a measure of the effective number of neighbors, providing a smooth measure of connectivity. In contrast, UMAP uses the number of nearest neighbors (Equation (B.3)) instead of perplexity (Equation (A.3)), which avoids utilizing the $log_2()$ function.

It is worth noting that the high-dimensional modeling formulas of t-SNE and UMAP can also provide insights into graph connectivity. In t-SNE, if a data point i has a large distance from all other data points, all $p_{i|j}$ values approach zero, resulting in an unconnected graph. On the other hand, in UMAP, the introduction of ρ_i ensures the presence of at least one $p_{i|j}$ value equal to one, guaranteeing graph connectivity. In addition, formula Equation (B.1) indicates that the output $p_{i|j}$ in UMAP is primarily determined by $d(x_i, x_j)$, and the elimination of the regularization term (denoted by the denominator in Equation (A1)) can decrease the computational burden when utilizing formula Equation (B.1).

Table 2. Comparison of t-SNE and UMAP in high-dimensional structural pattern modeling.

	t-SNE	UMAP
Conditional probability	$p_{j\|i} = \frac{e^{-\left\|x_i - x_j\right\|^2/2\sigma_i^2}}{\sum_{k \neq i} e^{-\left\|x_i - x_j\right\|^2/2\sigma_i^2}}$ (A.1)	$p_{i\|j} = e^{-\frac{d(x_i, x_j) - \rho_i}{\sigma_i}}$ (B.1)
Joint probability	$p_{ij} = \frac{p_{i\|j} + p_{j\|i}}{2N}$ (A.2)	$p_{ij} = p_{i\|j} + p_{j\|i} - p_{i\|j} p_{j\|i}$ (B.2)
Controllable parameter	Perplexity = $2^{-\sum_j p_{j\|i} log_2 p_{j\|i}}$ (A.3)	$k = 2^{\sum_i p_{ij}}$ (B.3)

3.3. Modeling the Fuzzy Topological Structure in Low Dimension

After capturing the structural patterns between pairs of high-dimensional data points, the subsequent task involves designing a low-dimensional probability function that effectively preserves these structural patterns from the high-dimensional space. The probability function utilized in t-SNE for low-dimensional representation is designed based on the Student's t-distribution, which can be expressed as

$$\hat{q}_{ij} = \frac{(1 + \left\|y_i - y_j\right\|^2)^{-1}}{\sum_{k \neq l} (1 + \left\|y_k - y_l\right\|^2)^{-1}} \tag{1}$$

where $\hat{q}_{ij}$ represents the low-dimensional probability for data points y_i and y_j when applying t-SNE and $\left\|y_i - y_j\right\|^2$ represents the square of the distance between data points y_i and y_j in low dimensions.

The preference for the Student's t-distribution over the Gaussian distribution in low-dimensional embedding is motivated by addressing the Crowding Problem [34]. By utilizing the Student's t-distribution, distances between points are amplified in low-dimensional space, alleviating the issue of point "crowding" and preventing excessive convergence of points in lower dimensions.

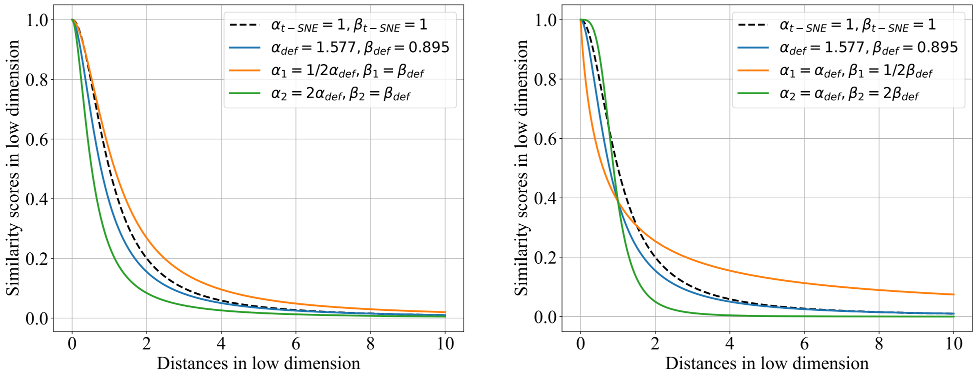
Unlike t-SNE, UMAP adopts the family of curves represented by $\frac{1}{1+\alpha y^{2\beta}}$ to model distance probabilities in low dimensions, where the function is defined as follows:

$$\tilde{q}_{ij} = \frac{1}{1 + \alpha (y_i - y_j)^{2\beta}} \tag{2}$$

where $\tilde{q}_{ij}$ represents the low-dimensional probability for data points y_i and y_j when applying UMAP, and α and β are two controllable parameters. In practice, α and β are determined from non-linear least-square fitting to the piecewise function with the hyperparameter *min_dist* as

$$\Psi q_{ij} \approx \begin{cases} 1, & \text{if} \left\|y_i - y_j\right\|_2 \leq min\_dist \\ e^{-(y_i - y_j) - min\_dist}, & \text{otherwise.} \end{cases} \tag{3}$$

The impacts of parameters α and β on the function curve, which describe the relationship between low-dimensional distance and similarity scores, are depicted in Figure 4a and Figure 4b, respectively. The default values for α and β are 1.577 and 0.895 (blue line). Notably, when α = 1 and β = 1 (black dash), Equation (1) corresponds to the t-SNE low-dimensional probability function. In Figure 4a, varying α values result in sunken curves (closer to the origin). Similarly, Figure 4b shows different β values, with larger β values causing a plateau at small distances in low dimensions. This indicates tight connections among data points below the UMAP hyperparameter *min_dist*. As the similarity score function behaves like a Heaviside step function, UMAP assigns nearly identical low-dimensional coordinates to closely related points. In addition, the *min_dist* parameter leads to the formation of tightly packed clusters commonly observed in UMAP dimensionality reduction plots.



(**a**) Parameter α (**b**) Parameter β

Figure 4. The impact of the parameters α and β on similarity scores in low dimension.

The parameters α and β are determined by solving the optimization problem with the constraints defined in Equation (3). Figure 5 visually presents the original points (blue dots) and the fitted curve (orange line) for different values of the hyperparameter *min_dist*: 0.5, 1, 1.5, and 2. The corresponding α and β values for these four scenarios are computed and provided in Table 3. In Figure 5, it can be observed that as the *min_dist* value increases, the plateau region of the curve becomes wider. A quantitative analysis from Table 3 reveals that α exhibits a positive correlation with *min_dist*, indicating that larger values of *min_dist* lead to larger β values. On the other hand, α demonstrates a negative correlation with *min_dist*, suggesting that higher values of *min_dist* correspond to smaller α values.

Table 3. Results for optimal parameters α and β with different *min_dist*.

	$min\_dist = 0.5$	$min\_dist = 1$	$min\_dist = 1.5$	$min\_dist = 2$
α	0.5743	0.1201	0.0193	0.0025
β	1.3714	1.8813	2.3993	2.9222

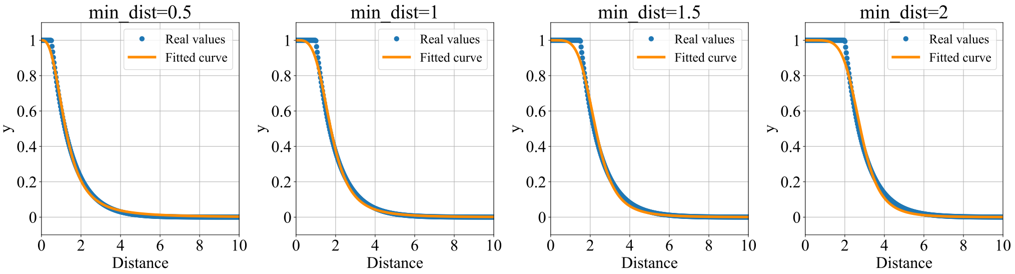


Figure 5. The curve-fitting results with different values of *min_dist*.

3.4. Initialization in Low Dimension

UMAP utilizes spectral embedding as its initialization method for assigning initial low-dimensional coordinates, replacing the randomized initialization used in t-SNE. This initialization plays a critical role in data structure preservation [35]. The implementation procedure can be summarized as follows: Firstly, the graph matrix $\boldsymbol{A}$ is determined, which represents the weighted adjacency matrix of the 1-skeleton of the topological representation. Then, the matrix $\boldsymbol{D}$ is calculated as the degree matrix for the graph $\boldsymbol{A}$. Next, the Laplacian matrix $\boldsymbol{L}$ is determined using the formula

$$\boldsymbol{L} = \boldsymbol{D}^{1/2}(\boldsymbol{D} - \boldsymbol{A})\boldsymbol{D}^{1/2}. \tag{4}$$

With the Laplacian matrix $\boldsymbol{L}$ determined, the eigenvalues and eigenvectors are computed, and the eigenvectors are sorted to serve as the output for the initialization.

Compared to randomized initialization methods, the utilization of spectral embedding in UMAP possesses two distinct advantages. Firstly, it reduces the variation of results across different trials by eliminating the randomness associated with random initialization. This increased consistency contributes to a more stable representation in the low-dimensional space. Secondly, spectral embedding provides a structured and consistent starting point for the dimension reduction process, potentially enhancing the stability and interpretability of the final low-dimensional representation.

3.5. Loss Function

The t-SNE uses the Kullback–Leibler (KL) loss function to project the high-dimensional probability onto the low-dimensional probability as [7]

$$\mathcal{C}_{KL} = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}} \tag{5}$$

where $p_{j|i}$ and $q_{j|i}$ denote the conditional probability defined in Equation (A.1).

In contrast, the construction of a cost function in UMAP aims to determine a suitable fuzzy topological structure for a low-dimensional representation. Theoretically, the fuzzy topological structure in low dimensions can be derived using a similar method as in high dimensions. However, a key distinction arises: in low dimensions, the data do not lie on a generic manifold but rather on a specific Euclidean manifold to which they need to be embedded. Consequently, the previous efforts to introduce variation in the notion of distance across the manifold become irrelevant. Instead, the distance on the manifold is sought to be the standard Euclidean distance with respect to the global coordinate system, which eliminates the need for a varying metric [36]. Furthermore, the performance of the representation is mathematically quantified by how "close" a match is found in terms of fuzzy topological structures, which can be turned into an optimization problem.

When merging the conflicting weights associated with simplices, it is conventionally interpreted as the weights representing the probability of the simplex being present. Consequently, since both compared topological structures share the same 0-simplices, the comparison can be viewed as a comparison between two vectors of probabilities indexed by the 1-simplices. Based on this, as these probabilities correspond to Bernoulli variables (where the simplex either exists or does not exist, and the probability serves as a parameter of a Bernoulli distribution), cross entropy (CE) is a suitable choice.

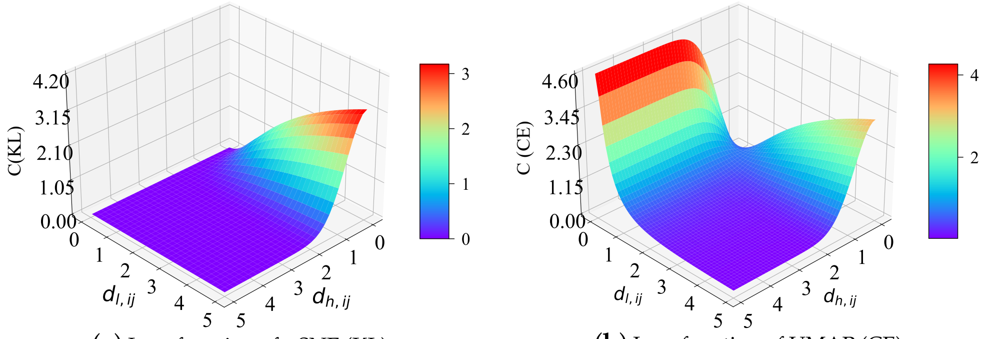
Let $\mathcal{S}$ denote the set of all possible 1-simplices, and consider weight functions $w_h(s)$ and $w_l(s)$ corresponding to the weights of the 1-simplices in the high-dimensional and low-dimensional cases, respectively. Thus, the CE loss function $\mathcal{C}_{CE}$ can be constructed as

$$\mathcal{C}_{CE} = \sum_{s \in \mathcal{S}} \left[\underbrace{w_h(s) log(\frac{w_h(s)}{w_l(s)})}_{Attractive\ force} + \underbrace{(1 - w_h(s)) log(\frac{1 - w_h(s)}{1 - w_l(s)})}_{Repulsive\ force} \right] \tag{6}$$

where $w_h(s)$ represents the weight of the 1-simplex s from high-dimensional manifold approximation and $w_l(s)$ represents the weight of the 1-simplex s to be discovered for low-dimensional representation.

The minimization of the $\mathcal{C}_{CE}$ can be interpreted as a force-directed graph layout algorithm. The first term in the $\mathcal{C}_{CE}$ equation exerts an attractive force between the points s that the 1-simplex spans whenever there is a large weight associated with the high-dimensional manifold approximation. This term is minimized when $w_h(s)$ is maximized, which occurs when the distance between the points is minimized. Conversely, the second term in the $\mathcal{C}_{CE}$ equation generates a repulsive force between the endpoints of s when $w_h(s)$ is small. Minimizing this term involves reducing $w_l(s)$ as much as possible. Through this interplay of attraction and repulsion, mediated by the weights on the edges of the topological representation of the high-dimensional data, the low-dimensional representation settles into a state that accurately reflects the overall topology of the source data.

A comparison of the KL loss function adopted in t-SNE and the CE loss function adopted in UMAP is visualized in Figure 6.



(**a**) Loss function of t-SNE (KL) (**b**) Loss function of UMAP (CE)

Figure 6. Comparison of the loss functions of t-SNE and UMAP.

3.6. Stochastic Gradient Descent (SGD)

Gradient descent (GD) is an optimization algorithm used to minimize a differentiable function by iteratively adjusting the parameters in the direction of the negative gradient [37].

Given a function $f(w)$ that we seek to minimize, the GD algorithm (as shown in Algorithm 1) updates the values of parameter w according to

$$w_{t+1} = w_t - \eta_t \nabla F_s(w_t) \tag{7}$$

where w_t represents the current parameter values, learning rate η_t is a hyperparameter that determines the step size in each iteration, and ∇ denotes the gradient vector of the function evaluated at the current parameter values. The gradient points in the direction of the steepest ascent, so taking the negative of the gradient ensures that the move is in the direction of the steepest descent. By iteratively updating the parameter values utilizing Equation (7), the algorithm gradually converges towards the optimal solution, where the gradient becomes close to zero, indicating a local minimum of the function.

Algorithm 1 Gradient Descent (GD)

```
for $t = 1$ to $T$ do
    $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t - \eta_t \bigtriangledown \boldsymbol{F}_s(\boldsymbol{w}_t)$
    $\eta_t = \frac{1}{\xi(t_0+t)}$
end for
return $\boldsymbol{w}_{T+1}$ or an average of $\boldsymbol{w}_1,\ldots,\boldsymbol{w}_{T+1}$
```

GD, while simple and widely used for optimization, has certain drawbacks. Firstly, the time for computing $\nabla F_s(w_t)$ is $\mathcal{O}(n)$, which is computationally expensive if n is large, as it requires a full pass over the entire training set to calculate the gradient as

$$\nabla F_s(w_t) = \frac{1}{n} \sum_{i=1}^{n} \nabla f(w_t). \tag{8}$$

Secondly, it may converge slowly when the loss function is non-convex or has narrow, elongated valleys. However, this is the case with the UMAP loss function. Additionally, it can get stuck in local minima, failing to find the global minimum. To address these drawbacks, stochastic gradient descent (SGD) is introduced [38]. SGD randomly selects a subset of training samples, called a mini-batch, to estimate the gradient, resulting in faster computation. This introduces noise into the gradient estimation, but it also allows the algorithm to escape shallow local minima and explore the parameter space more effectively. By repeatedly sampling mini-batches and updating the parameters accordingly, SGD can overcome the limitations of standard GD and converge to a satisfying solution.

The main difference between GD and SGD lies in Equation (8). As illustrated in Algorithm 2, instead of going through examples for a gradient computation, SGD adopts the form of

$$\nabla F_s(w_t) = \nabla f(w_t, z_{i_t}) \tag{9}$$

where $i_t \in \mathcal{N} = \{1, 2, ..., n\}$ represents a random index selected from $\mathcal{N}$ with equal probability. The learning rate η_t in (7) can be either constant or gradually decaying. For classification, the default learning rate schedule is given by

$$\eta_t = \frac{1}{\xi(t_0 + t)} \tag{10}$$

where t is the time step, t_0 is determined based on a heuristic proposed by Léon Bottou such that the expected initial updates are comparable with the expected size of the weights, and $\xi > 0$ is a non-negative hyperparameter that controls the regularization strength.

Algorithm 2 Stochastic Gradient Descent (SGD)

```
for $t = 1$ to $T$ do
    $i_t \leftarrow$ random index from $N = \{1, 2, \ldots, n\}$
    $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t - \eta_t \bigtriangledown \boldsymbol{F}_s(\boldsymbol{w}_t, z_{i_t})$
    $\eta_t = \frac{1}{\xi(t_0+t)}$
end for
return $\boldsymbol{w}_{T+1}$ or an average of $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{T+1}$
```

4. Mechanism Analysis of Global Data Structure Preservation and Computational Efficiency

4.1. Mechanism of Global Data Structure Preservation

The reasons for only the local structure of the data being preserved by t-SNE are discussed here from two perspectives: 1. the parameter σ in Equation (A.1); 2. the loss function, as shown in Equation (5).

- Firstly, the parameter σ in Equation (A.1) determines the degree of local interaction between data points. As shown in Figure 7, the pairwise Euclidean distances' probabilities decay at different speeds. Smaller values of σ such as the blue curve (σ = 0.1) and orange curve (σ = 1) result in near-zero probabilities for distant points (large pairwise Euclidean distances d), while rapidly increasing probabilities are observed only for the nearest neighbors (small pairwise Euclidean distance d). Conversely, larger σ such as the green curve (σ = 10) and red curve (σ = 20) values lead to comparable probabilities for distant and close points, and as σ approaches infinity (relatively larger values such as σ = 100 and σ = 200), the probabilities become equal to one for all distances between any pair of points, resulting in equidistant data points.
- Secondly, the "locality" of t-SNE can also be interpreted through the examination of its KL loss function (shown in Equation (5)). Assume that $d_{h,ij}$ is a high-dimensional distance between data points i and j, and $d_{l,ij}$ is a low-dimensional distance between data points i and j. Approximate $P(d_{h,ij})$ and $Q(d_{l,ij})$ as

$$P(d_{h,ij}) \approx e^{-d_{h,ij}^2}, \quad Q(d_{l,ij}) \approx \frac{1}{1 + d_{l,ij}^2} \tag{11}$$

The KL loss function $\mathcal{C}_{KL}(d_{h,ij}, d_{l,ij})$ can be approximated as

$$\mathcal{C}_{KL}(d_{h,ij}, d_{l,ij}) = \sum_i^n \sum_j^n \left[\underbrace{P(d_{h,ij}) \times log(P(d_{h,ij}))}_{\approx 0,\ \forall d_{h,ij}} - P(d_{h,ij}) \times log(Q(d_{l,ij})) \right]. \tag{12}$$

The first term in Equation (12) approaches zero for all $d_{h,ij}$. On the one hand, when $d_{h,ij}$ is small, it approaches zero as the exponent becomes close to one and $log(1) = 0$. On the other hand, when $d_{h,ij}$ is large, it also tends to zero as the exponential pre-factor decreases faster than the logarithm approaches $-\infty$. Thus, Equation (12) can be further approximated as

$$\mathcal{C}_{KL}(d_{h,ij}, d_{l,ij}) \approx \sum_i^n \sum_j^n \left[-P(d_{h,ij}) \times log(Q(d_{l,ij})) \right]. \tag{13}$$

Substituting the approximation Equation (11) into Equation (13), we have

$$\mathcal{C}_{KL}(d_{h,ij}, d_{l,ij}) \approx \sum_i^n \sum_j^n \left[e^{-d_{h,ij}^2} \times \left(1 + d_{l,ij}^2\right) \right]. \tag{14}$$

A visualization of $\mathcal{C}_{KL}(d_{h,ij}, d_{l,ij})$ with different values of $d_{h,ij}$ and $d_{l,ij}$ is shown in Figure 6a. The distribution of the loss function values (z-axis) in Figure 6a exhibits an asymmetric shape. Small distances between points in high dimensions $d_{h,ij}$ lead to an exponential pre-factor approaching one, while the logarithmic term behaves as $log(1 + d_{l,ij}^2)$. This enforces a significant penalty for large distances in low dimensions $d_{l,ij}$, motivating t-SNE to minimize $d_{l,ij}$ when $d_{h,ij}$ is small. Conversely, for large distances $d_{h,ij}$ in high dimensions, the exponential term dominates, allowing $d_{l,ij}$ to span from zero to ∞. Consequently, points that are distant in high dimensions may be projected closer in low dimensions. Hence, t-SNE solely guarantees the preservation of close points in high dimensions to remain close in low dimensions.

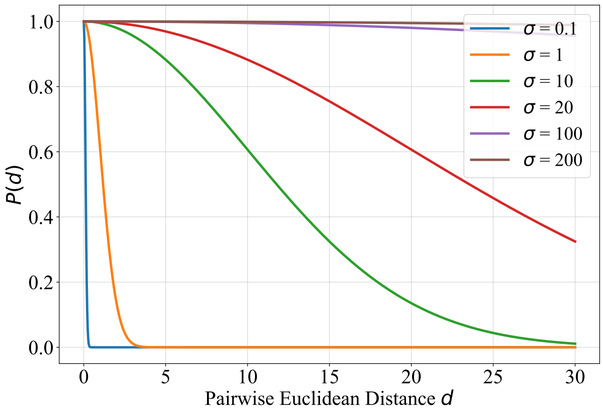


Figure 7. The impact of different σ on the high-dimensional probability.

Building upon the aforementioned explanations regarding the limited preservation of global structure by t-SNE, we now delve into the mathematical principles underlying UMAP to elucidate its ability to preserve global structure. Unlike t-SNE, UMAP employs CE as its loss function, which is expressed as

$$\mathcal{C}_{CE}(d_{h,ij}, d_{l,ij}) = \sum_i \sum_j \left[P(d_{h,ij}) \times log\left(\frac{P(d_{h,ij})}{Q(d_{l,ij})}\right) + (1 - P(d_{h,ij})) \times log(\frac{1 - P(d_{h,ij})}{1 - Q(d_{l,ij})}) \right] \tag{15}$$

where $P(d_{h,ij})$ and $Q(d_{l,ij})$ denote the joint probability defined in Equation (B.2).

By comparing Equation (15) and Equation (12), it can be observed that the first term in Equation (15) is equivalent to Equation (12). Therefore, UMAP intuitively preserves global structure by introducing the second term. Additionally, referring to the physical force interpretation in Equation (6), the second term corresponds to the repulsive force. In essence, UMAP's ability to preserve global structure is largely attributed to the inclusion of the repulsive force term. The impacts of this introduction on the final classification results are discussed and validated in a subsequent case study section. Furthermore, similar to the analysis conducted for t-SNE, a boundary limitation analysis is conducted here for the approximation interpretation.

Substituting the approximation of $P(d_{h,ij})$ and $Q(d_{l,ij})$, as shown in Equation (11), the loss function of UMAP $\mathcal{C}_{CE}(d_{h,ij}, d_{l,ij})$ can be further approximated as

$$\mathcal{C}_{CE}(d_{h,ij}, d_{l,ij}) \approx \sum_i \sum_j \left[e^{-d^2_{h,ij}} \times log\Big((1+d^2_{l,ij})\Big) + (1 - e^{-d^2_{h,ij}}) \times log\left(\frac{(1+d^2_{l,ij})}{d^2_{l,ij}}\right)\right]. \tag{16}$$

This results in the balance between local and global structure preservation. When $d_{h,ij} \to 0, \forall i,j \in \mathcal{V}$, the limit converges to that of t-SNE, as the second term vanishes due to the pre-factor and the slower growth of the logarithmic function compared to the polynomial function, as shown in the first case in Equation (17):

$$\mathcal{C}_{CE}(d_{h,ij}, d_{l,ij}) \approx \begin{cases} \sum_i^n \sum_j^n log(1+d^2_{l,ij}) & \text{if } d_{h,ij} \to 0, \forall i,j \in \mathcal{V} \\ \sum_i^n \sum_j^n log\left(\frac{1+d^2_{l,ij}}{d^2_{l,ij}}\right) & \text{if } d_{h,ij} \to \infty, \forall i,j \in \mathcal{V}. \end{cases} \tag{17}$$

The behavior is analogous to that of t-SNE. However, in the limit of large $d_{h,ij}$ (i.e., $d_{h,ij} \to \infty$), the first term vanishes, the pre-factor of the second term becomes one, and the expression of the second case in Equation (17) can be obtained. In this case, a high penalty is incurred when $d_{l,ij}$ is small due to its presence in the denominator of the logarithm. Consequently, $d_{l,ij}$ is encouraged to increase, causing the ratio under the logarithm to approach one and resulting in zero penalty. As $d_{h,ij}$ approaches infinity, $d_{l,ij}$ also tends towards infinity, ensuring the preservation of global distances during the transition from high-dimensional to low-dimensional space, which aligns with the desired objective.

A visualization of the CE loss function $\mathcal{C}_{CE}$ is presented in Figure 6b. Comparing Figure 6a and Figure 6b , the right part of the CE loss function $\mathcal{C}_{CE}$ (Figure 6b) demonstrates a noticeable resemblance to the KL loss function $\mathcal{C}_{KL}$ (Figure 6a). This resemblance indicates a preference for low $d_{l,ij}$ values at low $d_{h,ij}$ to minimize penalties. Conversely, at large $d_{h,ij}$, it becomes crucial for the $d_{l,ij}$ distance to be large. When $d_{l,ij}$ is small, the penalty incurred by the $\mathcal{C}_{CE}$ $(d_{h,ij}, d_{l,ij})$ term becomes exceedingly large. Notably, unlike the $\mathcal{C}_{KL}$ $(d_{h,ij}, d_{l,ij})$ surface, the $\mathcal{C}_{CE}(d_{h,ij}, d_{l,ij})$ cost function introduces a distinction in penalties between low and high $d_{l,ij}$ values at large $d_{h,ij}$. This distinction enables the $\mathcal{C}_{CE}$ $(d_{h,ij}, d_{l,ij})$ cost function to effectively preserve both global and local distances.

4.2. Mechanism of Computational Efficiency

UMAP's improved computational efficiency can be attributed to the following factors:

- Firstly, in the stages of probability modeling in high- and low-dimensional distance representation, the computational efficiency is enhanced by eliminating the normalization process in both high-dimensional and low-dimensional probability modeling, as summation or integration is computationally expensive. Additionally, the use of tree-based algorithms for nearest neighbor search in standard t-SNE results in slow performance for more than two embedding dimensions, as these algorithms scale exponentially with the number of dimensions.

- Secondly, the initialization, as demonstrated in the forthcoming case study section, has minimal impact on the final computational time and accuracy. While some effect may be present, it is not of an order of magnitude significance.
- Thirdly, UMAP adopts SGD for the optimization process, in contrast to the regular GD used in t-SNE. This choice enhances speed by calculating gradients from a random subset of samples rather than utilizing all samples as in regular GD. Additionally, SGD reduces memory consumption by storing gradients for only a subset of samples in memory rather than all samples.
- Fourthly, increasing the number of dimensions in the original dataset introduces sparsity, resulting in a fragmented manifold with dense regions and isolated points. UMAP resolves this issue by introducing the local connectivity parameter σ, which partially connects sparse regions through an adaptive exponential kernel that incorporates local data connectivity. This characteristic enables UMAP to theoretically operate with any number of dimensions, eliminating the necessity for a pre-dimensionality reduction step before integrating it into the primary dimensionality reduction procedure.

5. Case Study

In this section, a blade study is conducted to verify the theoretical analysis. A blade study, also known as a sensitivity analysis or parameter study, is a commonly adopted testing approach in the machine learning field. It is used to quantitatively examine how variations in individual parameters affect the overall system or model outcomes. By systematically varying one parameter at a time while keeping others constant, the resulting changes and understanding of the system's sensitivity, robustness, or vulnerability can be observed. The objective of the blade study in this study is to investigate the impacts of the previously mentioned factors on global and local structure preservation as well as computational efficiency. Specifically, the standard control variate method is used, where each scenario involves changing only one factor while keeping the other parameters at baseline levels.

Two groups of datasets are utilized:

1. Vibration data collected in our lab tests for motors with varying vibration levels.
2. Vibration data obtained from an open-source dataset [38] for motors with different bearing defect conditions.

For both datasets, six scenarios are defined based on factors related to modeling, initialization, loss function, and optimization, as discussed in Sections 3 and 4. The t-SNE modeling technique is used as the baseline for comparison. Computation time and bearing fault classification accuracy are statistically compared for both datasets.

5.1. Lab Experiment for Data Collection

An offline test was conducted in our laboratory to collect data under different motor-bearing lubrication conditions. The setup is shown in Figure 8. This experimental setup was designed to replicate the on-site motor, with identical specifications. The motor load was simulated using a tailor-made metal wheel, serving as a mechanical analogy for the fan blade and airflow. Safety measures, including emergency stops, circuit breakers, and protection shields, were implemented. The data collection schedule for each bearing lubrication level scenario is presented in Table 4. Five motors (M1 to M5) were tested, with "Original" indicating a new motor and "Reinstalled" denoting a motor that has been reinstalled. Bearing lubrication percentages were calculated based on weight. Each sample motor was scheduled to run for a minimum of three hours to obtain data. The classification task involved six classes representing lubricant levels: 100%, 75%, 50%, 25%, 10%, and 5%.

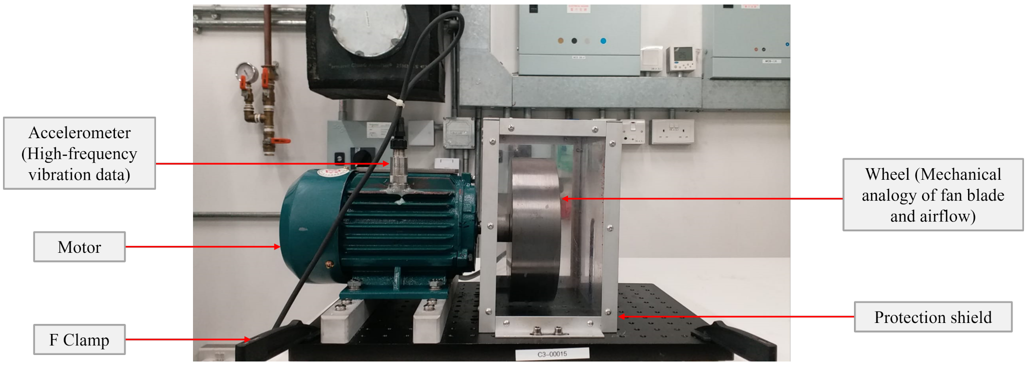

Accelerometer (High-frequency vibration data)
Motor
F Clamp
Wheel (Mechanical analogy of fan blade and airflow)
Protection shield

Figure 8. Laboratory experimental test for data collection.

Table 4. Lab test schedule of motor under different bearing lubrication conditions.

Test Class	Motor Name	Motor Condition	Fan-End Bearing Lubrication	Drive-End Bearing Lubrication	Duration (h)
A	M1	Original	100%	100%	6
	M2	Original	100%	100%	3
	M3	Original	100%	100%	3
	M4	Original	100%	100%	3
	M5	Original	100%	100%	8
B	M5	Reinstalled	75%	75%	8
C	M5	Reinstalled	50%	50%	8
D	M5	Reinstalled	25%	25%	8
E	M2	Reinstalled	10%	10%	10
F	M4	Reinstalled	5%	5%	141

5.2. Datasets

Samples were extracted for different lubricant levels at a rate of one second per sample. The sample sizes utilized in this study for each lubricant level are presented in Table 5.

Table 5. Balanced and unbalanced sample sizes for different lubrication levels.

	A	B	C	D	E	F	Total
Lubrication level	100%	75%	50%	25%	10%	5%	
Sample size 1 (balanced)	1000	1000	1000	1000	1000	1000	6000
Sample size 2 (unbalanced)	800	1000	900	1200	1000	1100	6000

In addition to the data collected from our laboratory test, the open-source data for different bearing faults are further tested for comparison and verification. The sample sizes for these open-source data under the category of *12k Fan-End Bearing Fault Data* used in this study are presented in Table 6.

Table 6. Balanced and unbalanced sample sizes for different bearing faults.

	A	B	C	D	E	F	Total
Fault diameter (cm)	0.018	0.018	0.036	0.036	0.053	0.053	
Motor load (HP)	0	1	0	1	0	1	
Sample size 3 (balanced)	2000	2000	2000	2000	2000	2000	12,000
Sample size 4 (unbalanced)	1800	2100	2000	1800	2100	2200	12,000

5.3. Results and Analysis

The results for computational time and fault detection accuracy, using the data obtained from our laboratory test, are presented in Table 7. Dataset 1 corresponds to the balanced data described in the second row of Table 4, while Dataset 2 represents the unbalanced data from the third row of Table 5. Each dataset comprises six categories and a total of 6000 samples. The train–test split is set at 0.8 to 0.2, and a tenfold cross-evaluation is adopted. The mean value (mean) represents the statistical mean, while the standard deviation (std) indicates the variability.

For comparison, the model structure of t-SNE is treated as the baseline. Seven different modeling structures are considered:

Scenario 1—Modeling (HD): refers to the scenario where UMAP high-dimensional modeling (Equation (B.1–3)) replaces t-SNE high-dimensional modeling (Equation (A.1–3)).

Scenario 2—Modeling (LD): denotes the scenario where UMAP low-dimensional modeling (Equation (2)) replaces t-SNE low-dimensional modeling (Equation (1)).

Scenario 3—Modeling (both HLD): represents the scenario where both UMAP high- and low-dimensional modeling (Equation (B.1–3), (2)) replace t-SNE high- and low-dimensional modeling (Equation (A.1–3), (1)).

Scenario 4—Spectral embedding: signifies the scenario where UMAP spectral embedding initialization (Section 3.4) replaces t-SNE random initialization.

Scenario 5—Loss function: denotes the scenario where the CE loss function (Equation (15)) replaces the KL loss function (Equation (12)).

Scenario 6—SGD: indicates the scenario where SGD is used instead of GD (Algorithms 1 and 2).

Scenario 7—UMAP: corresponds to the scenario where all t-SNE functions are replaced by UMAP functions.

Table 7. Results for the impact of UMAP components on time and accuracy using laboratory data.

Scenarios	**Dataset 1 (Balanced)**				**Dataset 2 (Unbalanced)**			
	Time (s)		**Accuracy**		**Time (s)**		**Accuracy**	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
t-SNE (baseline)	18.567	1.706	0.917	0.025	20.070	3.558	0.908	0.028
Modeling (HD)	13.972 ↓	1.562 ↓	0.937 ↑	0.021 ↓	13.985 ↓	2.663 ↓	0.932 ↑	0.020 ↓
Modeling (LD)	15.425 ↓	1.692 ≈	0.930 ↑	0.024 ≈	17.866 ↓	2.515 ↓	0.913 ↑	0.023 ≈
Modeling (both HLD)	10.730 ↓	1.223 ↓	0.975 ↑↑	0.019 ↓ ↓	10.592 ↓	2.332 ↓	0.970 ↑↑	0.020 ↓
Spectral embedding	18.782 ≈	1.509 ↓	0.920 ≈	0.024 ≈	21.717 ≈	3.320 ↓	0.910 ≈	0.028 ≈
Loss function	19.002 ≈	1.511 ↓	0.986 ↑↑	0.021 ≈	21.566 ≈	3.478 ≈	0.982 ↑↑	0.026 ≈
SGD	6.728 ↓↓	1.365 ↓	0.919 ≈	0.038 ↑	7.055 ↓	3.627 ↑	0.909 ≈	0.040 ↑
UMAP	**1.766** ↓↓	**0.786** ↓↓	**0.993** ↑↑	**0.020** ↓	**1.994** ↓↓	**2.314** ↓	**0.990**↑↑	**0.020** ↓

Note: ↓: decrease, ↓↓: large decrease, ≈: approximately equal, ↑: increase, ↑↑: large increase.

From the analysis of Table 7, the following conclusions can be drawn:

1. Both for balanced and unbalanced data, the enhanced fault detection accuracy, quantified by the formula $Accuracy = \frac{Correct\ predictions}{All\ predictions}$, is primarily achieved through the introduction of probability modeling formulas and the reformulation of the loss function. These results validate the theoretical discussion presented in Section 4.1 regarding the data structure-preserving property.
2. Both for balanced and unbalanced data, the increased computational efficiency, as measured by time, is primarily attributed to the exclusion of normalization in the high- and low-dimensional probability modeling, as well as the utilization of SGD instead of SG. This finding aligns with the theoretical discussion presented in Section 4.2.
3. The impact of low-dimensional initialization alone on computational efficiency and accuracy is negligible. However, when combined with modeling, loss function, and optimization, the overall performance can be significantly improved. This can be attributed to the stable transition of the structure-preserving pattern from high dimensions to the low dimension.
4. Both high- and low-dimensional modeling not only preserve the data structure but also reduce the standard deviation, indicating more stable results.

In addition to the laboratory test data, a practical industrial bearing fault dataset is utilized to further validate and compare the results. The complete dataset is publicly available in [39], and for this study, a subset of the data is used. The extracted subsets are referred to as Dataset 3 (balanced) and Dataset 4 (unbalanced). The sample sizes for these two datasets are provided in Table 6.

From the analysis of Table 8, the following conclusions can be drawn:

1. Similar to the laboratory test data results, enhanced fault detection accuracy for both balanced and unbalanced data is primarily achieved through the introduction of probability modeling formulas and the reformulation of the loss function. These findings further validate the theoretical discussion on the structure-preserving property presented in Section 4.1.
2. Similar to the laboratory test data results, the increased computational efficiency, as measured by time, for both balanced and unbalanced data is primarily attributed to the exclusion of normalization in high- and low-dimensional probability modeling as well as the utilization of SGD instead of SG. This observation further confirms the theoretical discussion presented in Section 4.2.
3. As for the initialization, similar to the results shown in Table 7, the impact of low-dimensional initialization alone on computational efficiency and accuracy is negligible, and both high- and low-dimensional modeling not only preserve the structure but also reduce the standard deviation.

Table 8. Results for the impact of UMAP components on time and accuracy using open-source data.

Scenarios	**Dataset 3 (Balanced)**				**Dataset 4 (Unbalanced)**			
	Time (s)		**Accuracy**		**Time (s)**		**Accuracy**	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
t-SNE (baseline)	31.766	3.545	0.935	0.022	32.472	3.778	0.923	0.025
Modeling (HD)	23.373 ↓	3.620 ↑	0.951 ↑	0.021 ≈	25.466 ↓	3.703 ↓	0.937 ↑	0.021 ↓
Modeling (LD)	29.367 ↓	3.468 ↓	0.947 ↑	0.020 ≈	29.588 ↓	3.715 ↓	0.929 ↑	0.023 ≈
Modeling (both HLD)	19.635 ↓	2.711 ↓	0.982 ↑↑	0.020 ≈	22.337 ↓	3.693 ↓	0.972 ↑↑	0.021 ↓
Spectral embedding	33.249 ≈	3.413 ↓	0.933 ≈	0.019 ↓	33.549 ≈	3.022 ↓	0.925 ≈	0.025 ≈
Loss function	32.472 ≈	2.591 ↓	0.982 ↑↑	0.016 ↓	33.632 ↑	3.445 ↓	0.971 ↑↑	0.028 ↑
SGD	9.762 ↓↓	3.402 ↓	0.937≈	0.022 ≈	11.567 ↓	3.402 ↓	0.924 ≈	0.024 ↓
UMAP	**2.076** ↓↓	**1.672** ↓↓	**0.989** ↑↑	**0.015** ↓	**2.768** ↓↓	**2.821** ↓	**0.987**↑↑	**0.020** ↓

Note: ↓: decrease, ↓↓: large decrease, ≈: approximately equal, ↑: increase, ↑↑: large increase.

Comparing Tables 7 and 8 yields the following observations:

1. UMAP demonstrates robust tolerance towards unbalanced data, as evident from both laboratory data and open-source data applications.
2. Increasing the data size leads to a proportional increase in computational time. While the impact on individual components is significant, the overall effect is relatively modest. Consequently, UMAP can handle larger data sizes without experiencing a substantial computational burden.
3. The laboratory data exhibit higher accuracy compared to the open-source data, suggesting sensitivity to the datasets, defect types, and levels. Nonetheless, all results demonstrate acceptable high-accuracy levels exceeding 99%.
4. A larger dataset presents higher standard deviations. This can be caused by the degree of variation between samples, the computational errors, and the randomized selection of sample batches for each evaluation.
5. Both case studies validate the aforementioned theoretical discussion concerning global and local structure preservation as well as computational efficiency.

Comparing the performance of t-SNE and UMAP on our lab-collected datasets (Dataset 1 and Dataset 2) and open-source datasets (Dataset 3 and Dataset 4), the impact of different datasets can be summarized as follows:

1. Impact of original data structure on accuracy: The original structure of high-dimensional data affects the accuracy of both t-SNE and UMAP. In classification tasks, as shown in this study, the clearer the relationships within clusters and between different clusters, the higher the accuracy of both algorithms. This is evidenced by the results showing that our lab experiment datasets present higher accuracy compared to the open-source data. This is because our experiment was designed to separate different lubricant levels with distinct patterns between clusters, while the differences between clusters (bearing fault levels) in the open-source data were not as distinct. Thus, it can be

inferred that the clearer the differences between data points in different clusters, the better the accuracy of t-SNE and UMAP in classification tasks.

2. Impact of data volume on computational efficiency: Computational efficiency is related to the volume of the data rather than the type of dataset. The larger the data volume, the more computation time is needed. However, as the data volume increases, UMAP shows a greater improvement in computational efficiency compared to t-SNE.

6. Discussions

Based on the results obtained in this study, the UMAP method is generally preferred over T-SNE, provided the UMAP algorithm is available, as it demonstrates higher accuracy and efficiency in most cases. However, it is noted that there may be certain scenarios where the advantages of UMAP are not as significant, in which either method can be adopted. Nevertheless, there are two specific situations where prioritizing the UMAP method over T-SNE is recommended:

- Situation 1: In data structures where high-dimensional data are close while low-dimensional data are far away, the T-SNE loss function penalizes discrepancies between low-dimensional and high-dimensional distances less severely. As a result, the resulting 2D embedding may position clusters with relatively small overall disparities farther apart than clusters with larger disparities. In such cases, the UMAP method should be adopted, as it is less prone to producing misleading results. This is because the UMAP loss function, which is based on binary cross-entropy, heavily penalizes low-dimensional distances deviating from their corresponding high-dimensional counterparts, regardless of the proximity of the high-dimensional distances. Consequently, UMAP demonstrates superior ability to preserve the intrinsic data structure in these scenarios.
- Situation 2: When multiple repeated trials are required and result consistency is paramount, the UMAP method should be prioritized over T-SNE. This is due to the different approaches used for initializing the low-dimensional data representation: T-SNE utilizes a random distribution, while UMAP assigns initial coordinates via a graph Laplacian transformation that leverages high-dimensional data characteristics. This distinction leads to UMAP exhibiting greater result stability across repeated experiments.

7. Conclusions

This paper presents a comparative analysis of t-SNE and UMAP, two manifold learning-based dimension reduction methods, from a mathematical perspective. The investigation focuses on two key aspects: global data structure preservation and computational efficiency. The mathematical principles underlying these aspects are explored, revealing that the global structure preservation property arises from high- and low-dimensional probability modeling and the design of the loss function. Computational efficiency is achieved by eliminating the normalization process during modeling and adopting stochastic descent instead of gradient descent. The impact of spectral embedding on the final results is minimal; however, when combined with high- and low-dimensional modeling and the loss function, it presents significant influences. Two datasets, comprising both balanced and unbalanced sample sizes, are employed for validation, including a laboratory test dataset and an open-source dataset, both related to bearing condition-related operation states.

The results confirm the superiority of UMAP over t-SNE in terms of global data structure preservation and computational efficiency. For the laboratory data, UMAP reduces computational time by 90.49% (balanced data) and 90.06% (unbalanced data), while increasing operation state identification accuracy by 8.29% (balanced data) and 9.03% (unbalanced data). For the open-source data, UMAP reduces computational time by 93.46% (balanced data) and 91.48% (unbalanced data), and increases operation state identification accuracy by 5.78% (balanced data) and 6.93% (unbalanced data).

Author Contributions: Conceptualization, W.Y. and S.B.; methodology, W.Y., H.-H.L., C.-H.C. and S.B.; validation, W.Y.; formal analysis, W.Y.; experiment: W.Y., H.-H.L. and C.-H.C.; data curation, W.Y. and C.-H.C.; writing—original draft preparation, W.Y.; writing—review and editing, S.B.; supervision, S.B.; project administration, H.-H.L.; funding acquisition, H.-H.L. and S.B. All authors have read and agreed to the published version of the manuscript.

Funding: The work presented in this article is supported by Centre for Advances in Reliability and Safety (CAiRS) admitted under AIR@InnoHK Research Cluster.

Data Availability Statement: Due to confidentiality reasons, access to our laboratory data is available upon request. The open-source data can be accessed at https://engineering.case.edu/bearingdatacenter (accessed on 1 June 2023).

Acknowledgments: The authors thank the editor and the anonymous reviewers for their time and valuable comments, which greatly contributed to the improvement of the manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

PCA	Principal Component Analysis
t-SNE	t-Distributed Stochastic Neighbor Embedding
UMAP	Uniform Manifold Approximation and Projection
GD	Gradient Descent
SGD	Stochastic Gradient Descent
KL	Kullback–Leibler Divergence
CE	Cross Entropy
HD	High Dimension
LD	Low Dimension

References

1. Fodor, I.K. *A Survey of Dimension Reduction Techniques*; Lawrence Livermore National Laboratory: Livermore, CA, USA, 2002.
2. Garzon, M.; Yang, C.-C.; Venugopal, D.; Kumar, N.; Jana, K.; Deng, L.-Y. *Dimensionality Reduction in Data Science*; Springer: Cham, Switzerland, 2022.
3. Espadoto, M.; Rafael, M.; Andreas, K.; Hirata, N.; Telea, A. Toward a quantitative survey of dimension reduction techniques. *IEEE Trans. Vis. Comput. Graph.* **2021**, *27*, 2153–2173. [CrossRef] [PubMed]
4. Abdi, H.; Williams, J. Principal Component Analysis. *Wiley Interdiscip. Rev. Comput. Stat.* **2010**, *2*, 433–459. [CrossRef]
5. Wold, S.; Esbensen, K.; Geladi, P. Principal component analysis. *Chemom. Intell. Lab. Syst.* **1987**, *2*, 37–52. [CrossRef]
6. Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
7. Maaten, L. Accelerating t-SNE using tree-based algorithms. *J. Mach. Learn. Res.* **2014**, *15*, 3221–3245.
8. Linderman, G.; Steinerberger, S. Clustering with t-SNE, provably. *SIAM J. Math. Data Sci.* **2019**, *1*, 313–332. [CrossRef] [PubMed]
9. McInnes, L.; Healy, J.; Melville, J. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv* **2019**, arXiv:1802.03426.
10. Sainburg, T.; McInnes, L.; Gentner, T.Q. Parametric UMAP embeddings for representation and semisupervised learning. *Neural Comput.* **2021**, *33*, 2881–2907. [CrossRef] [PubMed]
11. Ghojogh, B.; Ghodsi, A.; Karray, F.; Crowley, M. Uniform Manifold Approximation and Projection (UMAP) and its variants: Tutorial and survey. *arXiv* **2021**, arXiv:2109.02508.
12. Ma, Y.; Fu, Y. *Manifold Learning Theory and Applications*; CRC Press: Boca Raton, FL, USA, 2012.
13. Tong, L.; Zha, H. Riemannian manifold learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 796–809. [CrossRef] [PubMed]
14. Hamid, Y.; Sugumaran, M. A t-SNE based non linear dimension reduction for network intrusion detection. *Int. J. Inf. Technol.* **2020**, *12*, 125–134. [CrossRef]
15. Devassy, B.; George, S. Dimensionality reduction and visualisation of hyperspectral ink data using t-SNE. *Forensic Sci. Int.* **2020**, *311*, 110194. [CrossRef]
16. Devassy, B.; George, S.; Nussbaum, P. Unsupervised clustering of hyperspectral paper data using t-SNE. *J. Imaging* **2020**, *6*, 29. [CrossRef] [PubMed]
17. Huang, C.; Bu, S.; Lee, H.; Chan, C.; Yung, W. Prognostics and health management for induction machines: A comprehensive review. *J. Intell. Manuf.* **2024**, *35*, 937–962. [CrossRef]
18. Huang, C.; Bu, S.; Lee, H.; Chan, C.; Kong, S.; Yung, W. Prognostics and health management for predictive maintenance: A review. *J. Manuf. Syst.* **2024**, *75*, 78–101.

19. Xu, X.; Xie, Z.; Yang, Z.; Li, D.; Xu, X. A t-SNE based classification approach to compositional microbiome data. *Front. Genet.* **2020**, *11*, 620143. [CrossRef] [PubMed]
20. Kobak, D.; Berens, P. The art of using t-SNE for single-cell transcriptomics. *Nat. Commun.* **2019**, *10*, 5416. [CrossRef]
21. Li, W.; Cerise, J.; Yang, Y.; Han, H. Application of t-SNE to human genetic data. *J. Bioinform. Comput. Biol.* **2017**, *15*, 1750017. [CrossRef]
22. Linderman, G.; Rachh, M.; Hoskins, G.; Steinerberger, S.; Kluger, Y. Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data. *Nat. Methods* **2019**, *16*, 243–245. [CrossRef]
23. Wang, Y.; Huang, H.; Rudin, C.; Shaposhnik, Y. Understanding how dimension reduction tools work: An empirical approach to deciphering t-SNE, UMAP, TriMAP, and PaCMAP for data visualization. *J. Mach. Learn. Technol.* **2021**, *22*, 1–73.
24. Wu, D.; Poh Sheng, J.Y.; Su-En, G.T.; Chevrier, M.; Jie Hua, J.L.; Kiat Hon, T.L.; Chen, J. Comparison between UMAP and t-SNE for multiplex-immunofluorescence derived single-cell data from tissue sections. *bioRxiv* **2019**. [CrossRef]
25. Hozumi, Y.; Wang, R.; Yin, C.; Wei, G. UMAP-assisted K-means clustering of large-scale SARS-CoV-2 mutation datasets. *Comput. Biol. Med.* **2021**, *131*, 104264. [CrossRef]
26. Heiser, C.; Lau, K. A quantitative framework for evaluating single-cell data structure preservation by dimensionality reduction techniques. *Cell Rep.* **2020**, *31*, 107576. [CrossRef] [PubMed]
27. Rather, A.; Chachoo, M. Robust correlation estimation and UMAP assisted topological analysis of omics data for disease subtyping. *Comput. Biol. Med.* **2023**, *155*, 106640. [CrossRef]
28. Roman, V. *High-Dimensional Probability: An Introduction with Applications in Data Science*; Cambridge University Press: Cambridge, MA, USA, 2018; Volume 47.
29. Cho, H.; Venturi, D.; Karniadakis, G.E. Numerical methods for high-dimensional probability density function equations. *J. Comput. Phys.* **2016**, *305*, 817–837. [CrossRef]
30. Baraniuk, R.G.; Cevher, V.; Wakin, M.B. Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective. *Proc. IEEE* **2010**, *98*, 959–971. [CrossRef]
31. Xia, T.; Tao, D.; Mei, T.; Zhang, Y. Multiview spectral embedding. *IEEE Trans. Syst. Man. Cybern.* **2010**, *40*, 1438–1446.
32. Barron, J.T. A general and adaptive robust loss function. *arXiv* **2019**, arXiv:1701.03077.
33. Sun, S.; Cao, Z.; Zhu, H.; Zhao, J. A survey of optimization methods from a machine learning perspective. *IEEE Trans. Cybern.* **2019**, *50*, 3668–3681. [CrossRef] [PubMed]
34. Venna, J.; Peltonen, J.; Nybo, K.; Aidos, H.; Kaski, S. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Mach. Learn. Technol.* **2010**, *11*, 451–490
35. Kobak, D.; Linderman, G. Initialization is critical for preserving global data structure in both t-SNE and UMAP. *Nat. Biotechnol.* **2021**, *39*, 156–157. [CrossRef]
36. Chern, S.; Kuiper, N. Some theorems on the isometric imbedding of compact Riemann manifolds in Euclidean space. *Ann. Math.* **1952**, *56*, 422–430. [CrossRef]
37. Andrychowicz, M.; Denil, M.; Gomez, S.; Hoffman, M.W.; Pfau, D.; Schaul, T.; Shillingford, B.; De Freitas, N. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems 29*; Neural Information Processing Systems Foundation, Inc.: La Jolla, CA, USA, 2016; Volume 29.
38. Amari, S. Backpropagation and stochastic gradient descent method. *Neurocomputing* **1993**, *5*, 185–196. [CrossRef]
39. Ball Bearing Test Data for Normal and Faulty Bearings. Available online: https://engineering.case.edu/bearingdatacenter (accessed on 1 June 2023) .

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.