Ordering face dataset using mainfold learning

Chen Zhang
Department of CEE
HKUST
czhangdn@connect.ust.hk

Saheli Bhattacharya
Department of CEE
HKUST
sbaj@connect.ust.hk

Han Peng
Department of CEE
HKUST
hpengak@connect.ust.hk

Abstract

This project investigates the feature extraction ability of six dimension-reduction methods on face images whose main discriminative information is the rotation angle. We explore and discuss the ranking results of Diffusion Map, MDS, ISOMAP, LLE, LTSA, and t-SNE under different hyper-parameter settings, and find out that LTSA performs best on ordering these images by angle. The source code for implementing the six dimension-reduction methods can be found via the github link: `https://github.com/mandgrapes/CISC5011-Project-Code/tree/main`.

1 Introduction

Dimensionality reduction techniques are designed to deal with high-dimensional data. One challenge with such datasets is that not all features are relevant to a specific task. While certain methods with high computational complexity exhibit strong performance in prediction or classification problems, it's still valuable to decrease data dimensionality. This can lead to models that are easier to interpret and explain.

Dimensionality reduction primarily serves two purposes. The first is feature extraction. As data dimensions increase, an effective technique becomes essential for all data mining and machine learning tasks. Dimensionality reduction, often used as a preprocessing step, can enhance machine learning methods' performance and cut down training time. The second application is visualization, which is crucial for producing meaningful outputs. Good visualizations explain data distribution and the operational principles of machine learning methods.

Face pose estimation is a fascinating area of research. It aids in determining a person's focus and identifying associated behaviors. In this report, we aim to sequence the face orientation of 33 images from the same individual using various manifold learning methods, including Diffusion map, MDS, ISOMAP, LLE, LSTA, and TSNE. We derived the embedding results from various methods and arranged the faces based on the size of the first eigenvector. We also created a scatter plot of two eigenvectors, visualized the face images, and investigated the relationship between face orientation and data distribution. We then compared the outcomes of different manifold learning methods. Finally, we endeavored to identify the most suitable nearest neighbor number for ISOMAP, LLE, and LSTA.

The report is organized as follows: Section 2 introduces the dataset and presents the ground truth rank of face images. Section 3 provides a brief description of the six manifold learning methods. Section 4 analyzes the rank results of the first eigenvector and the 2D embedding graph. Conclusions are drawn in Section 5.

2 Dataset and problem statement

We utilized a classical dataset comprising 33 images of the same individual captured from various angles. The dataset, denoted as Y and represented by a three-dimensional matrix of size 112x92x33 ($Y \in \mathbb{R}^{112\times92\times33}$), can be accessed at the following URL: `https://github.com/yao-lab/yao-lab.github.io/blob/master/data/face.mat`.

Figure 1 displays the images in the ground truth order, illustrating the progression of the person's facial orientation. Initially, the individual's face is oriented towards the left, gradually transitioning towards the left direction, and finally returning to the right direction. Although the overall order exhibits a smooth transition, it is important to note that the angle changes do not consistently follow a single direction. This variability in angle changes can pose challenges for manifold learning algorithms in determining the correct image order when considering monotonic changes in coordinates.

Figure 1: Face images ordered by ground truth labels

The primary aim of this report is to accurately predict the order of face datasets through manifold learning methods, including Diffusion map, MDS, ISOMAP, LLE, LSTA, and t-SNE. We achieve this by extracting embedding results from these methods and organizing the faces based on the value of the first eigenvector. Furthermore, additional results are presented and analyzed in the subsequent sections.

3 Manifold learning methods

This section presents the equations and algorithms employed in this report for manifold learning methods used to determine the correct order of images.

3.1 Diffusion Map

Diffusion map [1] is a technique used to discern the low-dimensional manifold structure inherent in a high-dimensional dataset by leveraging diffusion distances as a metric. The process of dimensionality reduction involves discarding dimensions in which diffusion distances indicate a negligible likelihood of connection between points.

The initial step in applying diffusion maps is the creation of a kernel matrix that encapsulates the connectivity between any two data points x_i and x_j, representing the one-step transition probability in a random walk. Commonly, the Gaussian kernel is utilized for calculating the kernel matrix entries:

$$W_{ij} = k(x_i, y_i) = \exp\left(-\frac{(x_i - x_j)^2}{t}\right) \tag{1}$$

where t serves as the scale parameter of the kernel that requires careful tuning.

Subsequently, the Markov matrix is formed by row-wise normalization of the kernel matrix:

$$P = D^{-1}W \tag{2}$$

where $D = diag(\sum_j W_{ij})$ is the degree matrix.

The adjacency matrix is then defined as:

$$Q = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \tag{3}$$

It is noteworthy that matrices Q and P share identical eigenvectors and eigenvalues. The symmetry of Q simplifies the computation.

Let $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$ be the ordered eigenvalues of Q, with corresponding and eigenvectors $v_1, ..., v_n$. The eigenvalue $\lambda_1 = 1$ is typically disregarded as it does not provide useful information.

The diffusion map Y of Markov matrix P at time τ is defined as:

$$Y = [\lambda_2^\tau v_2, \lambda_3^\tau v_3, ..., \lambda_n^\tau v_n] \tag{4}$$

resulting in a $n \times (n-1)$ matrix.

It can be proved that the Euclidean distance amongst the diffusion coordinates is an effective approximation of the diffusion distance. Dimensionality reduction is realized by retaining the initial k columns of Y corresponding to the largest eigenvalues. The embedding of the i-th point in the diffusion map is represented by the i-th row of the truncated Y matrix, denoted as $y_i = Y(i, :)$

3.2 MDS

Multidimensional Scaling (MDS) [2] is a method designed to reconstruct Euclidean coordinates from pairwise distances or dissimilarities. Let us consider a forward distance computation problem. Say we have a collection of points $x_1, x_2, ..., x_n \in \mathbb{R}^p$ which can be write in the matrix form:

$$X = [x_1, x_2, ..., x_n]^{p \times n}$$

The Euclidean distance between points x_i and x_j is then calculated by

$$d_{ij}^2 = ||x_i - x_j||^2 = (x_i - x_j)^T (x_i - x_j) = x_i^T x_i + x_j^T x_j - 2 x_i^T x^j \tag{5}$$

MDS aims to solve the inverse problem: given only the distances d_{ij}, find a specific $x_i \in \mathbb{R}^p : i = 1, ..., n$ for some p that meet the condition $d_{ij} = ||x_i - x_j||$.

MDS has following two key steps:

(a) Convert the squared distance matrix $D = [d_{ij}^2]$ into a matrix that represents inner products;

(b) Carry out eigen-decomposition on the new matrix of inner products.

An central operation of MDS is to apply a double-centering transformation to the squared distance matrix to obtain the kernel matrix:

$$-\frac{1}{2} H D H^T = (XH)^T (XH) =: \hat{K} \tag{6}$$

Here $H := I - \frac{1}{n} \mathbf{1} \cdot \mathbf{1}^T = H^T$ is the Householder centering matrix, where $\mathbf{1} = (1, 1, ..., 1)^T \in \mathbb{R}^n$

Thus, $Y = XH = X - \frac{1}{n} X \cdot \mathbf{1} \cdot \mathbf{1}^T$ is the matrix of centered data, and $\hat{K} = Y^T Y$is the inner product matrix of this centered data. We can prove $\hat{K}$ is positive semi-definite and can be decomposed into orthogonal eigenvectors.

To achieve a dimensionality reduction, we often keep top k nonzero eigenvectors of $\hat{K}$ such that the collection of n data points is projected into a lower k-dimensional space.

3.3 ISOMAP

ISOMAP is one example of an isometric mapping method, it is an extension of metric multidimensional scaling or MDS [3]. It incorporates the geodesic distances imposed by a weighted graph. It defines the geodesic distance to be the sum of edge weights along the shortest path between two

nodes. ISOMAP differs from other scaling methods because it incorporates the geodesic distance generated by a neighborhood graph. This is done so that the resulting embedding will include a manifold structure. According to isomap, the geodesic distance is the total of the edge weights along the shortest path between any two nodes, e.g., by Dijkstra's algorithm.

Given a set of data $x_1, x_2, \ldots, x_n \in \mathbb{R}$ we firstly determine the neighbors of each data point and construct a neighborhood graph $G = (V, E)$, where

$$V = \{x_i : i = 1, \ldots, n\}, E = (i, j)$$

if j is a neighbor of i. Then we compute the shortest path $d_(i, j) = dist(x_i, x_j)$ between nodes $x_i \quad and \quad x_j$ by Dijkstra's algorithm. Finally, we repeat the MDS steps and use $dist(x_i, x_j)$ as the input of MDS to get the embedding results.

3.4 LLE/MLLE

In comparison to Isomap, locally-linear embedding (LLE) [4] offers a number of benefits, such as faster optimization when using sparse matrix algorithms and superior outcomes for a variety of issues. Locally linear Embedding (LLE) focuses on preserving the local linearity of the sample when reducing data dimensionality. Given graph $G = (V, E)$, the first step is to compute the distance from $x_(i)$ to $x_(j)$ and find the k nearest neighbors of $x_(i)$ by KNN.

$$N_i = KNN(x_i, k), N_i = [x_{1i}, \ldots, x_{ki}]$$

For each x_i and its neighbours $x_j \in N_i$, solve the reconstruction weights matrix w

$$\min \epsilon(w) = \sum_{i=1}^{n} \left\| x_i - \sum_{j=1}^{k} w_{ij} x_j \right\|^2 \quad subject \quad to \quad \sum_{j=1}^{k} w_{ij} = 1 \tag{7}$$

We create sparse matrix $M = (I - W)^T (I - W)$ where

$$W_{ij} = \begin{cases} w_{ij}, & \text{if } j \in N_i \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

We find bottom $d + 1$ nonzero eigenvalues from $\lambda_{n-1} \quad to \quad \lambda_{n-d}$ and their corresponding eigenvectors of M because we discard the bottom eigenvector with eigenvalue zero.

$$\min \epsilon(Y) = \sum_{i=1}^{n} \left\| y_i - \sum_{j=1}^{k} w_{ij} x_j \right\|^2 = \quad tr(YMY^T) \tag{9}$$

The final embedding coordinate is defined as $Y_d = U_d \Lambda_d^{1/2}$ where

$$U_d = [u_{n-d}, \ldots, u_{n-1}] \Lambda_d = diag(\lambda_{n-d}, \ldots, \lambda_{n-1}) \quad \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{n-1} > \lambda_n = 0$$

3.5 LSTA

Local tangent space alignment (LTSA) [5] applies PCA locally to obtain the basis of local tangent space and align these spaces after global optimization to reduce the **n**-dimension data to the **d**-dimension. The algorithm mainly contains three steps and is similar to the idea of LLE that preserves local correlations in high-dimensional data (Except LLE focuses on the neighborhood distance). 1) Compute Local SVD of neighborhood Matrix **X** for **k** Nearest neighbors. 2) Obtain Alignment Matrix **K** which is calculated from the top **d** dimension right singular vectors from local SVD and selection Matrix S. 3) Choose the smallest **d+1** eigenvectors of **K** and drop the smallest eigenvector to realize the **d**-embedding.

3.6 2D t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE) [6] replaces the normal distribution in SNE [7] with t-distribution and turns conditional probabilities into pairwise probabilities (Symmetric probabilities). The algorithm can be summarized into three steps: 1) Compute the joint probability of being a neighbor between two points (for all point-pairs) based on their Euclidean distance. 2) Initialize the position of data points in low-dimensional space and calculate the corresponding joint probabilities distribution. 3) Compare the two probability distributions according to KL divergence and update the point position in low-dimension space by gradient descent. t-distribution brings a larger gap among clusters in low dimensional embedding w.r.t normal distribution, meanwhile using joint probabilities makes the symmetrical nature of the cost function easy to optimize.

4 Results and discussion

4.1 Diffusion Map

The diffusion map results, depicted in Figure 2, exhibit a smooth transition in angles, effectively ordering the images. However, certain discrepancies are evident when comparing the predicted order to the ground truth order. To begin with, some highly similar images are not correctly placed in the order. Additionally, the diffusion map algorithm arranges the images with faces turning to the left at the end, whereas these images should ideally be positioned in the middle.

These discrepancies can be attributed to two reasons. First, the presence of images with very similar angles poses a challenge even for human identification, making it difficult for the algorithm to accurately determine their order. This task is inherently complex and prone to errors. Second, as previously mentioned, the algorithm solely relies on ordering the images based on monotonic changes in angles. Consequently, it fails to capture the sequence where faces turn both to the left and right, as the algorithm's internal limitations prevent it from recognizing such patterns.

Figure 2: Face images ordered by Diffusion map

4.2 MDS

When utilizing the MDS algorithm in scikit-learn, we observed that the results are influenced by the random seed. This is because the algorithm employs an iterative optimization procedure to determine the positions of points in a low-dimensional space. Consequently, it becomes necessary to set the random initialization for the point positions. To ensure consistent results across multiple runs of MDS using scikit-learn, the random state parameter must be specified when creating the MDS object.

In our implementation, we employed random states 42 and 100, and the corresponding results are depicted in Figure 3 and Figure 4, respectively. We can see that MDS results only partially capture the correct ordering of images. To mitigate the impact of random states, we conducted 100 runs of MDS with different random states and computed the average result, as shown in Figure 5. Unfortunately, the average result still did not show significant improvement.

The suboptimal performance of MDS can be explained by its fundamental objective of finding an affine space to represent the underlying data structure. This method is not well-suited for capturing data points that reside in manifolds exhibiting high nonlinearity in their geometric features. In such cases, it is advisable to consider nonlinear dimensionality reduction algorithms such as ISOMAP, which extends the capabilities of MDS specifically for manifolds.

Figure 3: Face images ordered by MDS (random state=42)

Figure 4: Face images ordered by MDS (random state=100)

Figure 5: Face images ordered by averaging 100 MDS runs

4.3 ISOMAP

Our study demonstrates the successful use of the Isomap algorithm with 5 nearest neighbors for face ordering, as visualized in Figure 6.The python script flattens each 2D facial image, applies the Isomap algorithm for dimensionality reduction, and then sorts the resulting embeddings to order the faces.

The 2D embedding graph from Isomap, also displayed in Figure 7, shows distinct advantages over the Multidimensional Scaling (MDS) method. Isomap uses geodesic distances instead of Euclidean ones, allowing for a more accurate representation of the data's intrinsic geometry in the 2D space.

The distribution of data points in the Isomap results reveals a clear, gradual transition of faces, reflecting the algorithm's ability to maintain the inherent structure of the data. This is an improvement over the results obtained from MDS, which can lose some structure due to its reliance on Euclidean distances.

In comparing the Ground Truth Labels and the Isomap Labels, we can see that the first 23 labels are identical in both cases, meaning the original order was maintained for these data points. However, starting from labels 24 to 33, the Isomap algorithm reordered the data points.This reordering suggests that the Isomap algorithm has identified a more significant geometric or intrinsic structure within these data points. The algorithm has effectively reinterpreted the distances between these latter points, indicating a different relationship or structure not initially apparent in the original ordering.

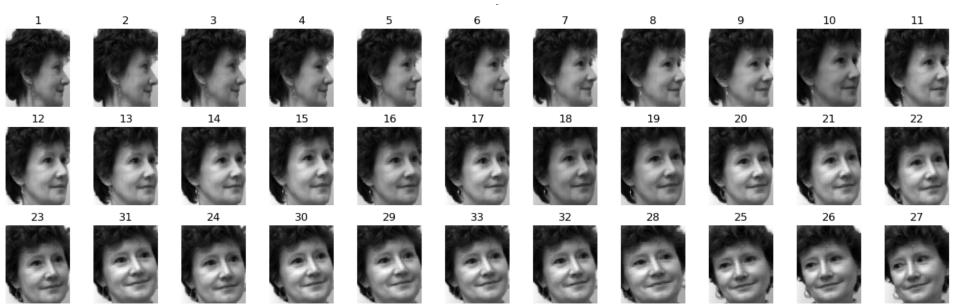

Figure 6: Face images ordered by ISOMAP

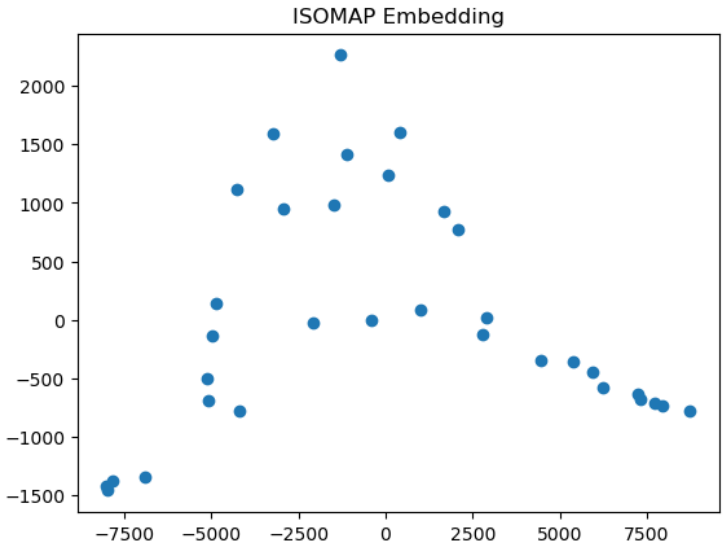

Figure 7: ISOMAP Embedding

4.4 LLE/MLLE

Our study effectively employed the standard and modified Locally Linear Embedding (LLE/MLLE) method with five nearest neighbors for face ordering, as visualized in Figure 8. The LLE embedding graph shown in Figure 9 illustrated a clear, gradual transition of faces, reflecting the algorithm's ability to maintain the inherent structure of the data. This gradual transition of faces provides visual evidence for the effectiveness of LLE in preserving data's local structure. Compared with ISOMAP , the LLE curve has a more uniform distribution because of the data approximation by local linear interpolations.

Comparing the Ground Truth Labels and the LLE Labels, it is evident that the first 24 labels (expect label 10 and 11) are identical in both cases, meaning the original order was maintained for these data points. However, starting from labels 24 to 33, the LLE algorithm reordered the data points. This reordering suggests that the LLE algorithm has identified a more significant local structure within these data points. The algorithm has effectively reinterpreted the distances between these points, indicating a different relationship or structure not initially apparent in the original ordering.

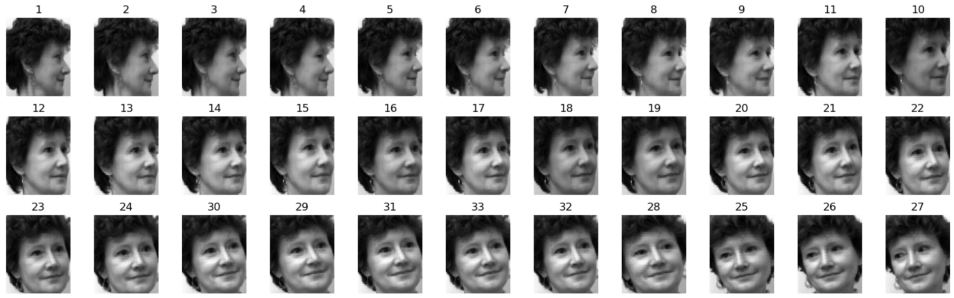

Figure 8: Face images ordered by LLE

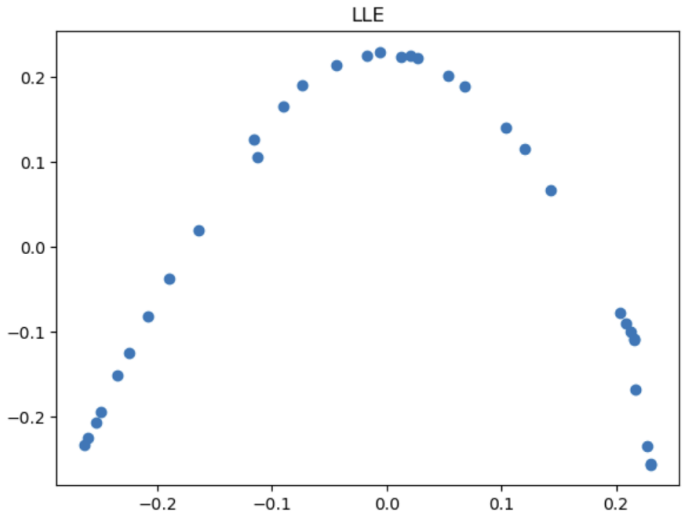

Figure 9: LLE Embedding

4.5 LSTA

We apply both 1D and 2D cases for LSTA. For LSTA 1D-embedding as Figure 10, the ordered images accurately match the default order before image 23 (Only a few image-pair switch the order like 18 and 19). For images 25 to 27 marked in the red box, the actual rotation angle is more than 90 degrees while the head in the rest of the images is rotated within 90 degrees. We consider these images as noisy data in default order. Their corresponding data points also have a large gap from the rest of the data points on the axis, which implies that LSTA may learn the angle of rotation from the images. Image 28 is another unusual data point. If we carefully inspect its detail, its rotation angle is the closest to 90 degrees with more pixels of the right side of the nose and the bigger face project. For LSTA 2D-embedding Figure 11, except the abnormal images mentioned above, the rest of the points can be approximately fitted with a linear function. The linearity indicates the rotation feature can be embedded well in one dimension.

Compared to ISOMAP, local tangent spaces alignment enables LSTA to capture more local variations, which is critical for our objective (ordering) that the rotations are visually trivial among neighbor

images. The whole pairwise geodesic distances, which ISOMAP tries to preserve, make the embedding focus on the global manifold. As a result, ISOMAP shows a more sparse scatter plot in 2D embedding that can not properly fit with any continuous or linear function. The order of projection on the y-axis also makes no sense. Besides, noisy and outlier images can affect the embedding of ISOMAP because of the potential erroneous shortest paths, while LSTA is robust to these images with k reference neighbor for each data point.

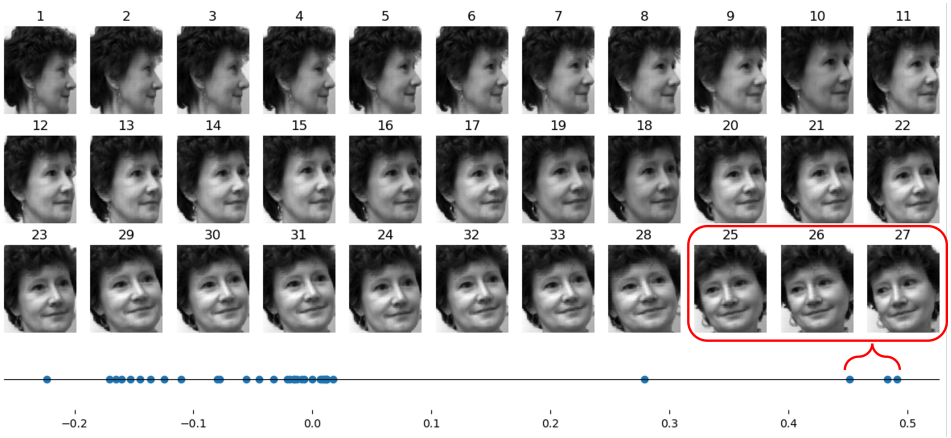

Figure 10: Face images ordered by 1D LSTA with corresponding 1D embedding axis. The number above each image represents the default order in the original dataset. The red box circles the noisy data in the default order.

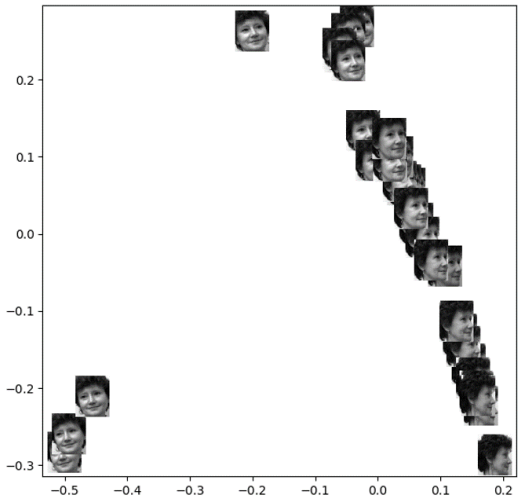

Figure 11: Face images embedding by 2D LSTA

4.6 2D t-SNE

We explore 2D t-SNE embedding around perplexity, as it is the key hyperparameter besides the embedding dimension. The choice of perplexity [8] can significantly affect the embedding results as shown in Figure 12. Perplexity measures the probability distribution used to model the neighborhood of a point, and high perplexity indicates a wider Gaussian (higher tail). To ensure the result is reproducible, we apply PCA initialization and fix the randomness for the ablation study. The data points are more dispersed and tend to form more clusters with higher perplexity. This can be attributed to low perplexity paying more attention to the local neighbors, while high perplexity considers more distant points. Therefore, low perplexity embedding is preferred for ordering the face images. We

pick the best ordering result between the project on the x-axis and y-axis for further comparison in section 4.7.

Compared to the above Manifold methods, which can usually perform good low-dimension embedding on the single continuous manifold like the well-known Swiss roll, t-SNE is powerful in finding local data structures and extracting clusters for similar data points. It is effective for visualizing datasets containing multi-manifolds like MNIST in low dimensions, but may not be good at dealing with sequential data points like our dataset.

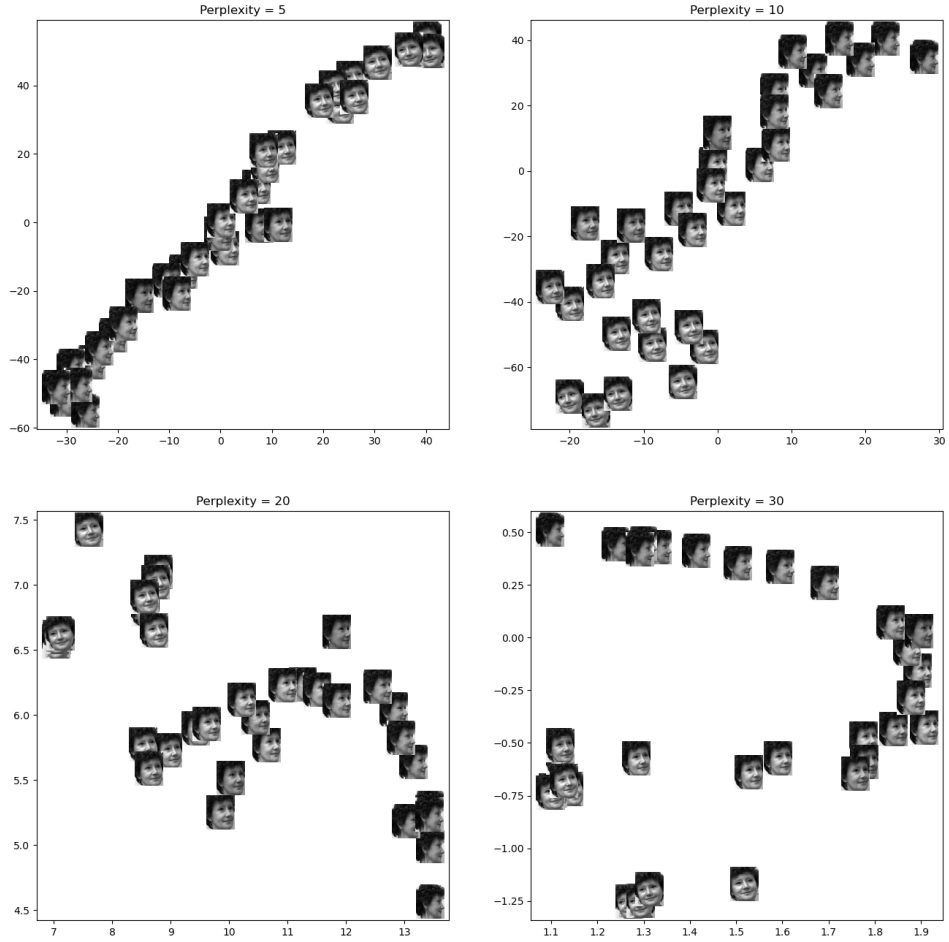

Figure 12: Face images embedding by 2D t-SNE under different Perplexity

4.7 Comparison of different manifold learning methods

4.7.1 Comparison metric

We assess the precision of different methods by selecting an appropriate metric. In this study, we utilize Kendall's Tau coefficient (τ) as our metric of choice. This coefficient quantifies the similarity between two orderings. A τ close to 1 indicates a high level of agreement, while a τ close to -1 signifies significant disagreement, and a τ around 0 indicates a lack of significant relationship. To calculate Kendall's Tau, we utilize the implementation available in the scipy.stats module of Python, which also provides a corresponding p-value.

A p-value less than or equal to 0.05 indicates strong evidence against the null hypothesis, suggesting a statistically significant relationship between the algorithm's ordering and the ground truth ordering.

Table 1: Metrics for different dimension-reduction methods

Methods	Diffusion map	MDS	ISOMAP	LLE/MLLE	LSTA	2D t-SNE
τ	0.856	0.477	0.697	0.674	0.898	0.875
p-value	0.000	0.000	0.000	0.000	0.000	0.000

Conversely, a p-value greater than 0.05 suggests insufficient evidence to reject the null hypothesis, indicating a lack of significant correlation between the algorithm's ordering and the ground truth.

Table 1 presents the resulting metrics for different dimension-reduction methods. Note that the result of MDS is taken from the average of 100 runs.

4.7.2 Interpretation of the result

Diffusion Map: With a τ of 0.856, this method does quite well. It indicates a strong positive correlation between the original high-dimensional data and the reduced dimensional data. This suggests that the ordering of faces is largely preserved.

MDS (Multidimensional Scaling): The τ of 0.477 suggests MDS is less effective in preserving the order of faces compared to the other methods. The order is only moderately preserved.

ISOMAP: With a τ of 0.697, ISOMAP is doing a decent job. The ordering of faces in the reduced dimensionality is fairly consistent with the original high-dimensional data.

LLE (Locally Linear Embedding): The LLE method, with a τ of 0.674, is also doing a fairly good job in preserving the order of faces.

LSTA: The LSTA method has the highest τ of 0.898, suggesting that it's the most effective at preserving the original order of faces in the reduced dimensional space.

2D-TSNE: The 2D-TSNE method also does quite well, with a τ of 0.875, indicating that the ordering of faces is largely preserved.

In conclusion, if your goal is to preserve the ordering of faces when reducing dimensionality, the LSTA method appears to be the most effective, followed by 2D-TSNE and Diffusion Map. However, the best method for your specific problem could also depend on other factors not reflected in these τ. It's often a good idea to consider other aspects like computational cost and the specific nature of your data.

5 Conclusions

In this project, the ordering performance given by Kendall's Tau coefficient indicates that Diffusion Map, LSTA, and 2D-TSNE with proper perplexity are all good methods for finding the feature of sequential data points. Meanwhile, we find out that the six methods can nicely deal with noisy face images whose rotation angle is much larger than the others. In general, LSTA is the most recommended and suitable manifold learning method for the face dataset or similar data in performance and theoretical perspectives.

6 Contributions

ZHANG, Chen: LSTA, t-SNE, Abstract, Conclusion and References.
BHATTACHARYA, Saheli: ISOMAP, LLE, Introduction, Interpretation of the result.
PENG, Han: Diffusion Map, MDS, Dataset and problem statement, Comparison metric, Figure labels, Formatting.

References

[1] De Porte, J., Herbst, B.M., Hereman, W. and van Der Walt, S.J. (2008) An Introduction to Diffusion Maps. In Proceedings of the 19th Symposium of the Pattern Recognition Association of South Africa (15-25).

[2] Borg, I., Groenen, P. (1997) The Four Purposes of Multidimensional Scaling. In Modern Multidimensional Scaling: Theory and Applications (3-14).

[3] Tenenbaum, J.B., De Silva, V. et al. (2000) A global geometric framework for nonlinear dimensionality reduction. *Science* **290**(5500):2319-2323.

[4] Roweis, S., Saul, L. (2000) Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**(5500):2323-2326.

[5] Zhang, Z.Y. & Zha, H.Y. (2004) Principal Manifolds and Nonlinear Dimensionality Reduction via Tangent Space Alignment. *SIAM Journal on Scientific Computing* **26**(1): 313-338.

[6] Hinton, G. E. & Roweis, S. (2002) Stochastic neighbor embedding. *Advances in neural information processing systems* **15**.

[7] Van der Maaten, L. & Hinton, G. (2008) Visualizing data using t-SNE. *Journal of machine learning research* **9**(11).

[8] Wattenberg, M., Viégas, F. & Johnson, I. (2016) How to use t-SNE effectively. *Distill* **1**(10): e2.