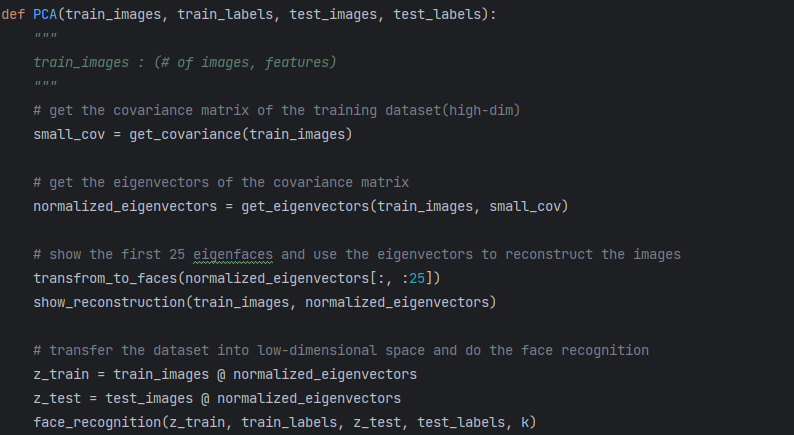
1. **Code with detailed explanations (40%)**

Kernel Eigenfaces (25%)

Part1 (10%)

PCA:

In PCA, derive the projection matrix W by computing the eigenvectors of the covariance matrix in the high-dimensional space, selecting the q eigenvectors corresponding to the q largest eigenvalues. Display the first 25 eigenfaces based on these eigenvectors and reconstruct 10 randomly chosen images.

get\_covariance():

When the covariance matrix is too large, computing its eigenvectors directly becomes time- and memory-intensive. Instead of calculating the eigenvectors for a (features,features)-sized matrix, we compute the eigenvectors of a (# of datapoints, # of datapoints)-sized matrix. This approach, often referred to as the dual space trick in PCA, is more efficient for datasets with many features but fewer data points.一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

get\_eigenvectors():

To compute the eigenvalues and eigenvectors efficiently, we use the smaller matrix. Once the eigenvectors of the small matrix are obtained, they are mapped back to the original feature space using the formula:



where ui is the eigenvector of the origin covariance. XT is the transpose of the data matrix, vi is the eigenvector of the small matrix. Finally, sort the eigenvalues in descending order to rearrange the corresponding eigenvectors.一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

transform\_to\_faces():

In this function we reshaped the first 25 eigenvectors to the size of images and showed all of them.

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

自動產生的描述

show\_reconstruction():

First random pick 10 images and then for each image it transfers to low dimension first and then transfer back into high dimension to get the reconstruction of it by using this formula:



Finally, show the original images and corresponding reconstruction images.

一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

自動產生的描述

LDA:

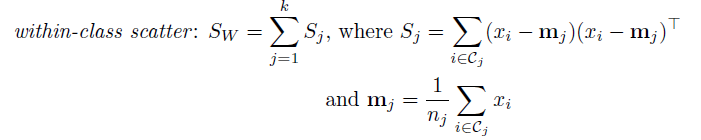
First, apply PCA to reduce the dimensionality, as the goal of PCA is to lower dimensions and reduce computational complexity. Next, use the dimensionality-reduced data from PCA to compute the LDA eigenvectors. Once the LDA eigenvectors are obtained, display the first 25 eigenfaces and use these eigenvectors to reconstruct the images.

一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

自動產生的描述

get\_LDA\_eigenvectors():

To get LDA eigenvectors, we need to calculate the SW and SB , the formula of SW and SB are as follows,



一張含有 文字, 字型, 圖表, 螢幕擷取畫面 的圖片

自動產生的描述

After obtaining these two matrix, the way to get the eigenvectors of projection matrix W is let



Finally, sort the eigenvalues in descending order to rearrange the corresponding eigenvectors.

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述

Part 2 (5%)

First, construct the low-dimensional representations of the training and testing datasets using

一張含有 字型, 印刷術, 圖形, 符號 的圖片

自動產生的描述

Then, perform classification on the testing dataset using the k-Nearest Neighbors (k-NN) algorithm based on the low-dimensional features. In LDA, the dimensionality reduction must result in a number of dimensions fewer than the number of classes; otherwise, LDA may not effectively capture all the discriminative directions between classes.

In PCA:

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

In LDA:

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述

Part 3 (10%)

kernelPCA:

First, compute the kernel matrix for the training dataset based on the specified kernel type. Next, derive the eigenvectors of the kernel matrix to form the projection matrix. Finally, use the projection matrix to obtain the low-dimensional representations of both the training and testing datasets in the kernel space, and perform classification on the testing dataset using the kernel-based features.一張含有 文字, 螢幕擷取畫面, 字型 的圖片

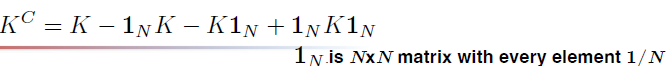
自動產生的描述

get\_kernel\_function():

We use two kernel matrix for the kernelPCA and kernelLDA, one is RBF matrix, another is polynomial kernel matrix.

RBF: exp(-γ||x-y||2)

Polynomial: (xTy)d

For easy computation we need to center the kernel matrix by using this formula:  


However, for the testing kernel, X should be testing data Y should be training data, this cause the kernel matrix is not symmetry. Therefore, 1N need some modification according to their size.

一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

kernelLDA:

First, compute the kernel matrix for the training dataset based on the specified kernel type. Next, derive the eigenvectors of the kernel matrix to form the projection matrix. The function get\_LDA\_eigenvectors is as same as in LDA. Finally, use the projection matrix to obtain the low-dimensional representations of both the training and testing datasets in the kernel space, and perform classification on the testing dataset using the kernel-based features.

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

t-SNE (15%)

Part 1 (10%)

The parts that need to be modified are the calculation of qij and the gradient of y

qij :

tSNE: 一張含有 字型, 文字, 行, 筆跡 的圖片

自動產生的描述

SSNE: 一張含有 文字, 字型, 行, 白色 的圖片

自動產生的描述

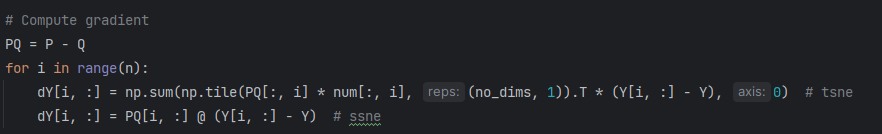
  
  
y:

tSNE: 一張含有 字型, 文字, 行, 印刷術 的圖片

自動產生的描述

SSNE: 一張含有 字型, 文字, 印刷術, 筆跡 的圖片

自動產生的描述



Part 2 (2%)

For every 50 or 100 iterations, save the images of the scatter plot of 2D space data according to their labels.

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

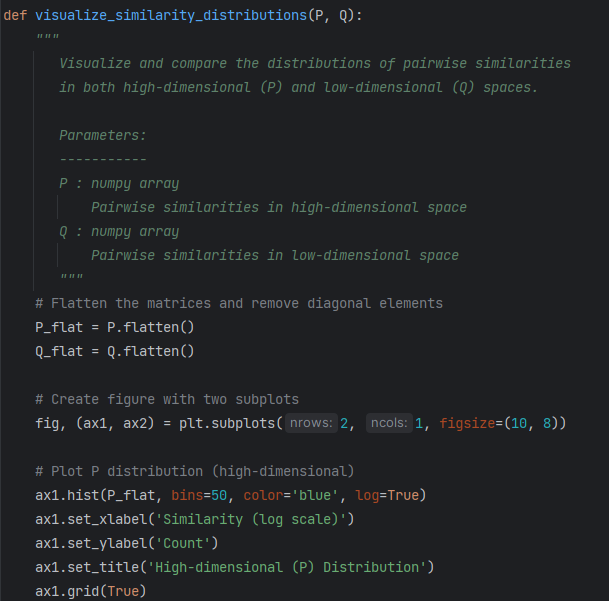
自動產生的描述

After that, use these images to construct the gif of the whole process

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Part 3 (2%)



一張含有 文字, 螢幕擷取畫面 的圖片

自動產生的描述

Part 4 (1%)

just change the variable of perplexity





1. **Experiments and Discussion (50%)**

Kernel Eigenfaces (20%)

Part 1 (5%)

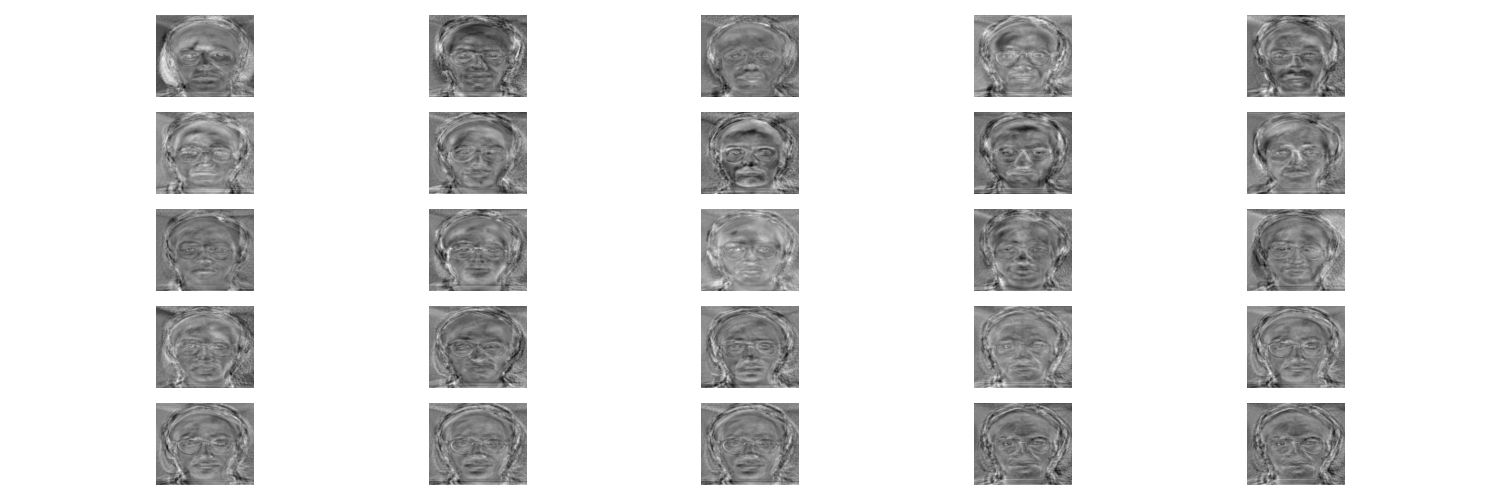
First 25 eigenfaces



Reconstructed faces of PCA

The left column represents the original images, while the right column shows the corresponding reconstructed faces. These reconstructed faces were obtained by reducing the dimensionality of the features to 100.

First 25 fisher faces:



Reconstructed faces of LDA

The left column represents the original images, while the right column shows the corresponding reconstructed faces. These reconstructed faces were obtained by reducing the dimensionality of the features to number of classes -1 (=14).



Part 2 (5%)

parameter for k nearest neighbors: k=3

PCA:

reducing the dimensionality of the features to 100.



LDA:

reducing the dimensionality of the features to 14(# of classes -1).



Part 3 (5%)

parameter for k nearest neighbors: k=3

kernelPCA:

reducing the dimensionality of the features to 100.

RBF : gamma = 1/number of features = 0.0000222



Polynomial: degree = 5



kernelLDA:

reducing the dimensionality of the features to 14(number of classes -1).

RBF: gamma = 1/number of features = 0.0000222



Polynomial: degree = 3



Discussion:

The performance of the kernel method is significantly worse than that of simple PCA and LDA. I experimented with different kernel parameters, but the results did not improve. I see three possible explanations for this:

1. Implementation Error: There may be an error in my kernel method implementation.
2. Inappropriate Kernel Choice: The chosen kernel might not be suitable for this dataset, as it may fail to accurately represent the similarities between data points.
3. Linear Nature of the Data: Kernels are generally more effective for non-linear data distributions. If the dataset has a predominantly linear distribution, methods like PCA and LDA might perform better.

This is supported by my test results. For example, simple LDA achieved an accuracy of 1.0, indicating that standard LDA was already sufficient to preserve the critical features of the dataset. This suggests that the dataset is likely closer to a linear distribution.

Additionally, there was little difference in performance between the RBF and polynomial kernels. At best, the polynomial kernel performed slightly better in kernel PCA, but it still fell significantly short of the performance of standard PCA.

t-SNE (30%)

Part 1 (5%) & (5%) Please discuss the observation in this part.

|  |  |
| --- | --- |
| Symmetry SNE | t-SNE |
|  |  |

The key difference between SSNE and t-SNE lies in the method of calculating Q. SSNE uses a normal distribution to convert the distances between data points into probabilities, while t-SNE employs a t-distribution. The t-distribution has longer tails, meaning that in the low-dimensional space, data points need to be further apart to achieve a low probability. This characteristic allows t-SNE to address the crowding problem encountered in symmetric SNE.

Part 2 (5%)

Perplexity = 20

Target dimension = 2

initial dimension(reduce by PCA) = 50

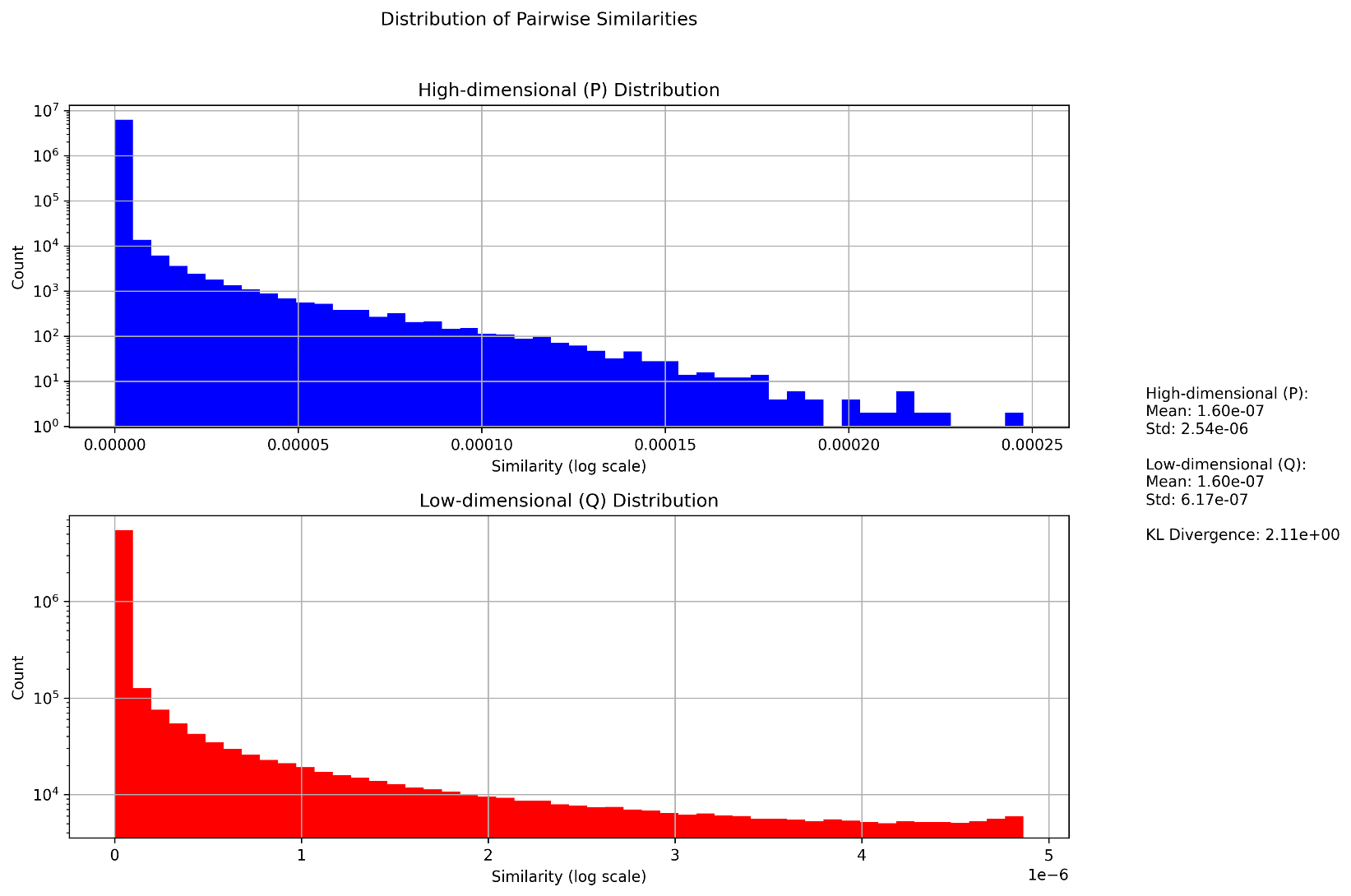
iterations: SSNE:500, t-SNE:1000

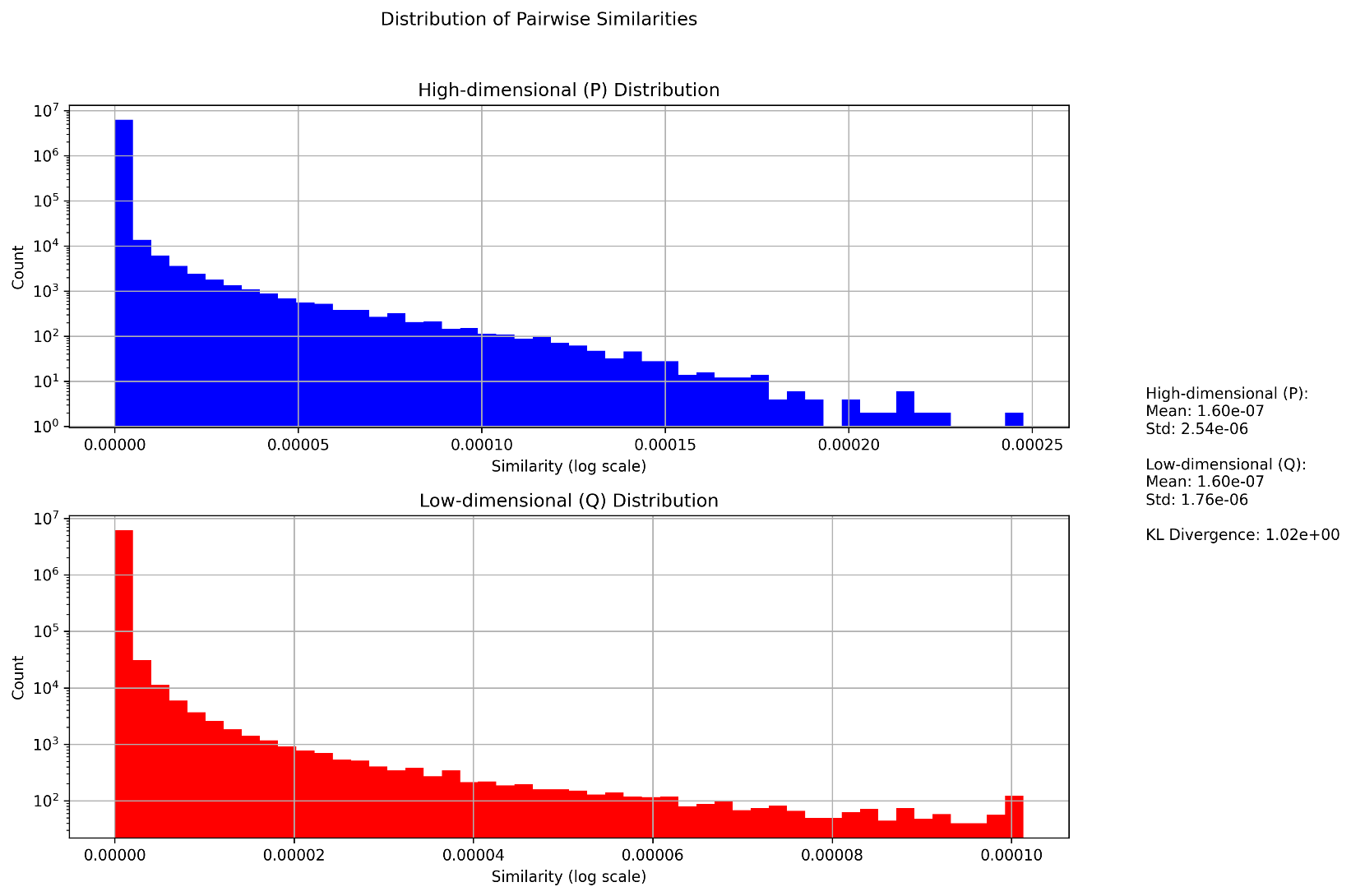
The reason that SSNE use smaller iteration is because it has higher convergence speed. **( Observations and Discussion)**

|  |  |
| --- | --- |
| Symmetry SNE | t-SNE |
| 一張含有 地圖 的圖片  自動產生的描述 | 一張含有 地圖 的圖片  自動產生的描述 |

Part 3 (5%)

Symmetry SNE:



t-SNE:

It is clear that t-SNE effectively separates similarities corresponding to distances, whereas the similarity distribution in SSNE appears more scattered. **( Observations and Discussion)**

Part 4 (5%) & (5%) Please discuss the observation in this part.

|  |  |  |
| --- | --- | --- |
| Perplexity | Symmetry SNE | t-SNE |
| 5 |  |  |
| 20 |  |  |
| 50 |  |  |

It can be observed that when the perplexity is higher, points within the same group tend to distribute more evenly. Conversely, when the perplexity is lower, the same group forms smaller, tighter local clusters. This happens because perplexity measures the effective neighborhood size, representing the approximate number of neighbors for each data point. A lower perplexity means fewer effective neighbors, causing closer points to be grouped together, while points farther away, even with the same label, may not be considered part of the same local cluster. Additionally, due to SSNE's crowding problem, the effects of different perplexity values are less noticeable in its visualizations.

**Observations and Discussion (10%)**

The reconstructed faces in PCA are better than those in LDA. I believe this is because the maximum dimensionality reduction in LDA is constrained by the number of classes. As a result, when reducing the dimensions and mapping back to the original feature space, much of the finer detail is lost.

|  |  |
| --- | --- |
| PCA reconstruction | LDA reconstruction |
|  |  |

Each eigenface represents a basis vector in a lower-dimensional subspace that captures the most significant variations in the dataset. These variations often correspond to features such as the general shape of the face, shadows, or other patterns in facial structure.

Summary of SSNE and t-SNE

|  |  |  |
| --- | --- | --- |
| Features | SSNE | t-SNE |
| |  | | --- | | **Low-Dimensional Similarity** | | |  |  |  | | --- | --- | --- | |  | Gaussian distribution |  | | |  | | --- | | Student's t-distribution (heavier tails) | |
| |  | | --- | | **Crowding Problem** | | |  |  | | --- | --- | |  | Poor handling | | |  | | --- | | Better handling | |
| |  | | --- | | **Preservation** | | |  |  |  | | --- | --- | --- | |  | Emphasizes local structure |  | | |  | | --- | | Balances local and global structure | |
| |  | | --- | | **Visual Quality** | | |  |  |  | | --- | --- | --- | |  | Less interpretable |  | | |  | | --- | | Highly interpretable | |
| |  | | --- | | **Parameter Sensitivity** | | |  | | --- | | **Parameter Sensitivity** | | |  | | --- | | **Parameter Sensitivity** | |