HIROSHIMA UNIVERSITY

MASTER THESIS

Nonlinear Dimensionality Reduction with q-Gaussian Distribution

Author:
Motoshi ABE (M206466)

Supervisor:
Professor Takio KURITA
Sub Supervisor:
Junichi MIYAO
Hiroaki MUKAIDANI

A thesis submitted in fulfillment of the requirements for the degree of Master of Engineering

in the

Informatics and Data Science Program

January 5, 2022

Declaration of Authorship

I, Motoshi ABE (M206466), declare that this thesis titled, "Nonlinear Dimensionality Reduction with q-Gaussian Distribution" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Motoshi Abe

Date: 2021

HIROSHIMA UNIVERSITY

Abstract

Graduate School of Advanced Science and Engineering
Informatics and Data Science Program

Master of Engineering

Nonlinear Dimensionality Reduction with q-Gaussian Distribution

by Motoshi ABE (M206466)

In recent years, dimensionality reduction has become more important as the number of dimensions of data used in various tasks such as regression and classification has increased. As popular nonlinear dimensionality reduction methods, t-distributed stochastic neighbor embedding (t-SNE) and uniform manifold approximation and projection (UMAP) have been proposed. However, the former outputs only one low-dimensional space determined by the t-distribution and the latter is difficult to control the distribution of the distance between each pair of samples in low-dimensional space. To tackle these issues, we propose novel t-SNE and UMAP extended by q-Gaussian distribution, called "q-Gaussian distributed stochastic neighbor embedding" (q-SNE) and "q-Gaussian distributed uniform manifold approximation and projection" (q-UMAP). The q-Gaussian distribution is a probability distribution derived by maximizing the tsallis entropy by escort distribution with mean and variance, and a generalized version of Gaussian distribution with a hyperparameter q. Since the shape of the q-Gaussian distribution can be tuned smoothly by the hyperparameter q, q-SNE and q-UMAP can intuitively derive different embedding spaces. However, they are applicable for a given data set and it is not possible to map new samples into the embedded space. To address this issue for the t-SNE, parametric t-SNE has been proposed. The parametric t-SNE constructs the nonlinear mapping by using a feed-forward neural network. We proposed a novel technique called parametric q-SNE as an extension of parametric t-SNE by using a convolutional neural network (CNN). To show the quality of the proposed method, we compared the visualization of the low-dimensional embedding space and the classification accuracy by k-NN in the low-dimensional space. Empirical results on MNIST, COIL-20, OliverttiFaces, and FashionMNIST demonstrate that the q-SNE and q-UMAP can derive better embedding spaces than t-SNE and UMAP. Empirical results on MNIST, COIL-20, and FashionMNIST demonstrate that the parametric q-SNE can derive better embedding spaces than parametric t-SNE and PCA.

Acknowledgements

I would like to express my gratitude to Associate Professor Takio Kuirta, Junichi Miyao, and Hiroaki Aizawa. They provided the good environment for my research, supported my student life and helped my research with many ideas. I am also grateful to Professor Masaru Tanaka at Fukuoka University, who supported my research. He provides many knowledge and supported my research. Our proposals were inspired from his text book. My research was completed thanks to their knowledge and advice.

I am also grateful to my lab members and my family.

Contents

List of Figures

List of Tables

List of Abbreviations

SNE	**S**tochastic **N**eighbor **E**mbedding
t-SNE	**t**-distributed **S**tochastic **N**eighbor **E**mbedding
UMAP	**U**niform **M**anifold **A**pproximation and **P**rojection
q-SNE	**q**-Gaussian distributed **S**tochastic **N**eighbor **E**mbedding
q-UMAP	**q**-Gaussian distributed **U**niform **M**anifold **A**pproximation and **P**rojection
PCA	**P**rincipal **C**omponent **A**nalysis

Chapter 1

Introduction

Recently, the number of dimensions of data has been increasing in the world. The dimensionality reduction has been widely used to the high-dimensional data for regression, classification, feature analysis, and visualization [23] [36]. Dimensionality reduction produces a low-dimensional mapping of the high dimensional data that preserves some features of interest in the data [11]. The principal component analysis (PCA) [31], multidimensional scaling (MDS) [10], Fisher's linear discriminant analysis (LDA) [29], canonical correlations analysis (CCA) [45], linear regression, and locally linear embedding (LLE) [38] have been proposed as linear dimensionality reduction techniques. The kernel PCA [40], neural network (NN), stochastic neighbor embedding (SNE) [18], Sammon mapping [39], Isomap [44], Maximum Variance Unfolding (MVU) [49], Laplacian Eigenmaps [6], curvilinear components analysis (CUCA) [12], Visualizing Large-scale and High-dimensional Data (LargeVis) [43], t-distributed SNE (t-SNE) [26], and uniform manifold approximation and projection (UMAP) [28] have been proposed as nonlinear dimensionality reduction techniques. Dimensionality reduction aims to preserve the structure and features of high-dimensional data as much as possible and project them to lower dimensional space.

For the visualization of high-dimensional data by using linear dimensionality, the PCA [31] is the most popular technique. For example, it is used for visualization of activation mapping of deep neural network [1] [19], experimental comparisons of dimensionality reduction [18] [44] [26] [28], or using compressed data for input [26]. It uses a matrix to map to low-dimensional space. This matrix is derived from eigenvectors of covariance matrix of high-dimensional data. It is called mapping function or mapping matrix. Since the PCA makes mapping matrix, it can map the new coming sample to same low dimensional space which is fitted by training samples.

For the visualization of high-dimensional data by using non-linear dimensionality reduction, the SNE[18], t-SNE [26], and UMAP [28] are often used. They give better embedding in low-dimensional space than PCA, Sammon mapping, Isomap, and LLE, because they consider the proximity between high-dimensional space and low-dimensional space using a probability distribution. The Gaussian distribution is used in high-dimensional space as a probability distribution by SNE, t-SNE, and

UMAP. They can consider the probability of similarity between samples in high-dimensional space by using the Gaussian distribution. In low-dimensional space, the SNE uses the Gaussian distribution. The SNE embeds the similarity between samples in the high-dimensional space which are defined by using local Gaussian distribution into the low-dimensional space which are also defined by using local Gaussian distribution. The SNE uses the Kullback-Leibler divergence [20]. It measures the similarity between two probability distributions. The Kullback-Leibler divergence between the local Gaussian distributions of the original high-dimensional space and the embedded low-dimensional space is used to measure the goodness of the embedded space. It is known that the SNE can visualize the distributions of the high-dimensional data in 2 or 3-dimensional embedded space. However, the SNE has some problems in which the Gaussian distribution in the embedded low-dimensional space does not give enough weights for the distant samples from the center point and the separation between the clusters or the samples is not enough in the embedded space. On the other hand, the t-SNE uses the t-distribution in low-dimensional space instead of the normal Gaussian distribution of SNE. In the paper of t-SNE, it can make better visualization than SNE because the t-distribution allows samples with closer distances to be embedded closer together and samples with large distances to be embedded farther apart. However, the t-SNE has some problems. It takes long computational time when the total number of samples is large [28], can not control the low-dimensional space, because distribution in low-dimensional space can not be changed. To improve computational time, several algorithms are proposed [46, 7, 47]. The UMAP has been proposed as an extension of t-SNE. The UMAP can reduces the computational time than t-SNE because it does not use all samples when computing similarity in high-dimensional space. It also can control the low-dimensional space, because it uses a changeable curve which is similar to the probability distribution in low-dimensional space. These methods consider the Gaussian distribution, the t-distribution, and the curve which is similar to probability distribution to embed the proximity of high-dimensional data into low-dimensional space.

However, these method can not embed the new coming samples for the same embedding space, which is fitted by training samples, like a PCA [31], because these methods do not construct the mapping function from the high dimensional space to the low dimensional embedding space. To address this problem, the parametric t-SNE [48] has been proposed as an the variants of t-SNE. The parametric t-SNE uses a feed-forward deep neural network [5] as a mapping function. In the parametric t-SNE, the mapping function from the high dimensional space to the embedding space are modeled by using a feed-forward deep neural network. In that paper, the parametric t-SNE uses deep multi-layered perceptron (MLP) [27] with only fully connected layers with sigmoid activation function as a feed-forward deep neural network. It can embed the new coming samples to the same embedding space which is fitted by training samples.

One of the feed-forward neural network, convolutional neural network (CNN) [3] has been proposed. The CNN has been widely used for image classification [15] [16] [22] [41], image recognition [51] [4], object detection [14] [25] [37], and so on. For each task of image data, the CNN can achieve great success, because the structure of CNN specializes to extract good features of images. The CNN is constructed by some convolutional layers, pooling layers, and fully connected layers.

The Gaussian distribution and t-distribution are very famous. In statistics, they are often appeared. There are not only them, but also many probability distributions. The q-Gaussian distribution [42] is also proposed as one of them. The q-Gaussian distribution is a probability distribution obtained when the Tsallis entropy is maximized by escort distribution with mean and variance. The q-Gaussian distribution is the generalized version of the Gaussian distribution with a hyperparameter q. It has Gaussian distribution when $q \to 1$, moreover it has t-distribution with degrees of freedom ν when $q = 1 + \frac{2}{\nu+1}$ and Cauchy distribution when $q = 2$. The advantage of the q-Gaussian distribution is that several distributions can be selected smoothly by tuning the hyperparameter q.

In this paper, we proposed a new non-linear dimensionality reduction method using the q-Gaussian distribution to improve t-SNE and UMAP. Since the q-Gaussian distribution can change the distribution smoothly, using the q-Gaussian distribution provides intuitive operation by choosing hyperparameter q. We experimentally verified the advantages of the proposed method by changing the conventional distributions to the q-Gaussian distribution. These are called q-Gaussian distributed stochastic neighbor embedding (q-SNE) [2] and q-Gaussian distributed uniform manifold approximation and projection (q-UMAP). For q-SNE, it can control visualization and can be same as t-SNE by tuning the hyperparameter q. For q-UMAP, it can also control visualization and provide intuitive selectivity of distribution for low-dimensional space by choosing hyperparameter q than UMAP.

We also proposed a novel parametric non-linear dimensionality reduction technique called parametric q-SNE. The proposal q-SNE has same problem as t-SNE which can not embed the new coming samples. Since we focused on image data as input data, the parametric q-SNE uses a convolutional neural network instead of the MLP of parametric t-SNE. The CNN is modeled to minimize the loss function of q-SNE. The parametric q-SNE can embed the new coming samples to same embedding space.

Next section, we describe related works to introduce our proposal. First, we describe about probability distribution. We show the formulation of Gaussian distribution, t-distribution, and q-Gaussian distribution. Since the q-Gaussian distribution is related to our proposal techniques, we describe detail and show the graphs in Fig.2.1 Second, we describe about linear dimensionality reduction. Third, we describe about non-linear dimensionality reduction. We show the formulation of SNE, t-SNE, and UMAP. Finally, we describe about parametric non-linear dimensionality reduction and CNN. We show the formulation of parametric t-SNE and CNN.

Third section, we describe our proposal techniques about q-SNE and q-UMAP. The q-SNE improves t-SNE by using q-Gaussian distribution. The q-UMAP improves UMAP by using q-Gaussian distribution. In experiments, we show the the effectiveness of q-SNE and q-UMAP by comparison of embedding visualization and classification accuracy by using k nearest neighbor (k-NN) [34]. First, we show the preliminary experiments by using swissroll data in Fig.3.2. This experiment shows the change of embedding when the hyper parameter q of q-Gaussian distribution is moved. For comparison experiments we used MNIST [13], COIL-20 [35], OliverttiFaces [24], and FashionMNIST [50] dataset. We show the change of embedding when the hyper parameters are moved in Fig.3.4, Fig.3.5, Fig.3.6, and Fig.3.7. We also show the classification accuracy scores in Table 3.1. The q-SNE and q-UMAP can control visualization intuitively and they can achieve better classification accuracy than t-SNE and UMAP.

Forth section, we descrbe our proposal techniques about parametric q-SNE. In experiments, we show the the effectiveness of parametric q-SNE by comparison of embedding visualization and classification accuracy by using k nearest neighbor (k-NN). We show the change of embedding when hyper parameter q of q-Gaussian distribution is moved in Fig.4.2. It also shows embedding by using test samples. We also show the classification accuracy scores in Table 4.2. The parameteric q-SNE can derive better embedding spaces than parametric t-SNE and PCA, and it can map the new coming samples for the same embedding space which is fitted training samples.

Finally, we describe about our conclusion. We summarized our proposal, experiments, and future works.

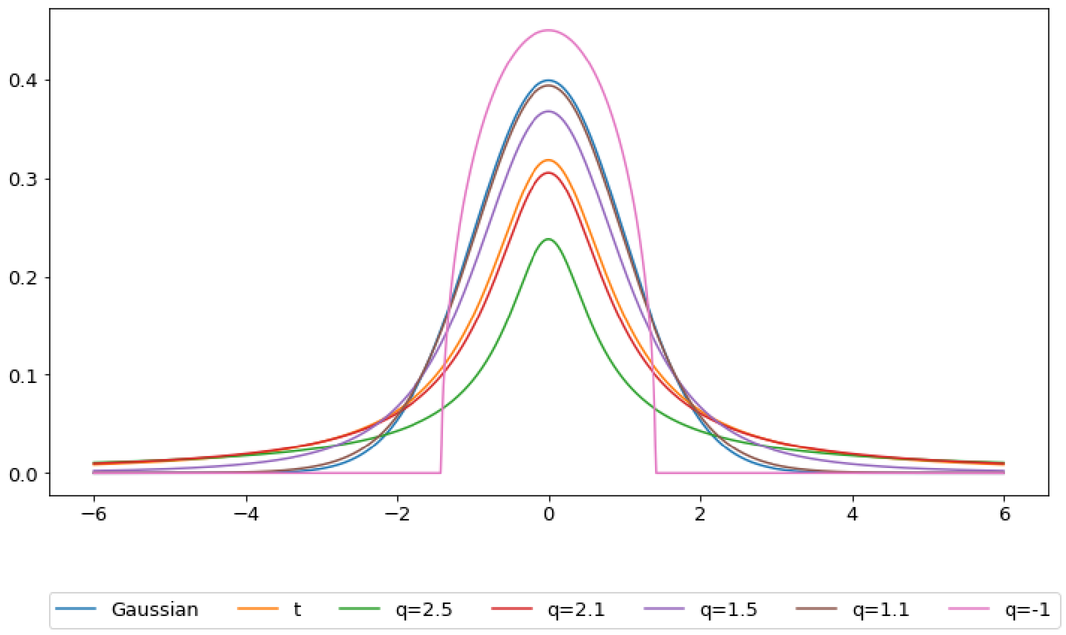


FIGURE 2.1: This figure shows the graphs about a Gaussian distribution, a t-distribution, and some q-Gaussian distributions. In legend, Gaussian denotes the Gaussian distribution as blue line, t denotes the t-distribution of degrees of freedom 1 as orange line, and q=2.5, q=2.1, q=1.5, q=1.1 and q=-1.0 denote the q-Gaussian distributions as green, red, purple, brown and pink line, respectively. When q=2.0, the q-Gaussian distribution becomes t-distribution.

Chapter 2

Related Works

2.1 Probability Distribution

2.1.1 Probability Distribution

The probability distribution is used for most dimensionality reduction techniques because they can take into account the proximity of high-dimensional data. The Gaussian distribution is used for SNE [18], t-SNE [26] and UMAP [28] in high-dimensional space. For SNE [18], it also use the Gaussian distribution in low-dimensional space. The Gaussian distribution with a 1-dimensional observation s is defined as

$$P(s;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(s-\mu)^2}{2\sigma^2}\right), \tag{2.1}$$

where μ and σ are mean and variance.

The t-SNE [26] uses the t-distribution in low-dimensional space. The t-distribution is defined as follows.

$$P_t(s;\nu) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)} \left(1+\frac{s^2}{\nu}\right)^{-\left(\frac{\nu+1}{2}\right)}, \tag{2.2}$$

where ν is degree of freedom.

2.1.2 q-Gaussian distribution

The q-Gaussian distribution is derived by the maximization of the Tsallis entropy of escort distribution with mean and variance and is a generalization of the Gaussian distribution.

Let s be a 1-dimensional observation. The q-Gaussian distribution for the observation s is defined as

$$P_q(s;\mu,\sigma^2) = \frac{1}{Z_q}\left(1+\frac{q-1}{3-q}\frac{(s-\mu)^2}{\sigma^2}\right)^{-\frac{1}{q-1}} \tag{2.3}$$

where μ and σ are the mean and the variance, respectively. The normalization factor Z_q is given by

$$Z_q = \begin{cases} \sqrt{\frac{3-q}{q-1}} Beta\left(\frac{3-q}{2(q-1)}, \frac{1}{2}\right)\sigma, & 1 \le q < 3 \\ \sqrt{\frac{3-q}{1-q}} Beta\left(\frac{2-q}{1-q}, \frac{1}{2}\right)\sigma, & q < 1 \end{cases} \tag{2.4}$$

where $Beta()$ is the beta function. It is known that the q-Gaussian distribution defined by Eq.(2.3) always satisfies the inequality

$$1+\frac{q-1}{3-q}\frac{(s-\mu)^2}{\sigma^2} \ge 0. \tag{2.5}$$

The q-Gaussian distribution has the parameter q as shown in Eq.(2.3) and we can recover the Gaussian distribution and the t-distribution by setting the parameter q in the q-Gaussian distribution. Fig.2.1 shows the graph of the Gaussian distribution, the t-distribution, and the q-Gaussian distributions with a few different parameters q. In this graph, we set μ and σ to 0 and 1. If $q \to 1$, then the q-Gaussian distribution becomes the Gaussian distribution. If $q = 1 + \frac{2}{n+1}$, then the q-Gaussian distribution becomes the t-distribution of degrees of freedom n. If $q < 1$, we call compact support to this case and s has the region of cutoff as

$$-\sqrt{\frac{3-q}{1-q}}\sigma + \mu < s < \sqrt{\frac{3-q}{1-q}}\sigma + \mu. \tag{2.6}$$

We show the example of compact support as pink line in Fig.2.1. From Fig.2.1, it is noticed that the q-Gaussian distribution has a more sharp peak at 0 than the t-distribution.

2.2 Linear Dimensionality Reduction

The linear dimensionality reduction produces linear low-dimensional mapping of high-dimensional data. The John P. et al. [11] defined linear dimensionality reduction as a matrix optimization problem. Let $X = [x_1, x_2, \cdots, x_N] \in R^{D\times N}$, where D is a high dimension and N is a number of samples, be high-dimensional samples. The low-dimensional projection $Y = [y_1, y_2, \cdots, y_N] \in R^{d\times N}$, where d is a low dimension, is obtained by linear projection function $f_M(X)$ as follows,

$$Y = f_M(X) = M^T X, \tag{2.7}$$

where $M \in R^{D\times d}$ is a linear projection matrix with d orthonormal columns. The optimization framework is written by

$$\begin{aligned} minimize_M \|X - MM^T X\|^2 \\ subjecttoM \in \mathcal{M} \end{aligned} \tag{2.8}$$

where $\mathcal{M} = M \in R^{D\times d} : MM^T = I$ is a orthogonality constraint for linear matrix.

2.2.1 Principle Component Analysis

The PCA is one technique of the linear-dimensionality reduction. In [11], PCA is derived from 2.8. Let $\mathcal{O}$ be orthogonal matrices, the PCA is written by

$$\begin{aligned} minimize_M \|X - MM^T X\|^2 \\ subjecttoM \in \mathcal{O}^{D\times d}. \end{aligned} \tag{2.9}$$

Let $\frac{1}{N}XX^T$ be covariance matrix, the decomposition is written as $XX^T = Q\Lambda Q^T$ which produces an optimal point $M = Q_r$, where Q_r denotes the columns of Q assosiated with the largest r eigenvalues of XX^T and Λ denotes a matrix of eigenvalues.

2.3 Non Linear Dimensionality Reduction

2.3.1 SNE

The SNE embeds the pairwise similarities between samples in the high-dimensional space into the low-dimensional space. The goodness of the low-dimensional space is evaluated as the Kullback-Leibler divergence between the conditional probabilities of the samples in the high-dimensional space and the low-dimensional space. Both the conditional probabilities in the high-dimensional space and the low-dimensional space are defined by using local Gaussian distribution.

Let $\{\boldsymbol{x}_i | i = 1 \dots N\}$ be a set of the samples in the high-dimensional space. We assume that the vectors of each samples are represented as $\boldsymbol{x}_i = \begin{bmatrix} x_{i1} & x_{i2} & \cdots & x_{iD} \end{bmatrix}^T$ and the dimension of the vector is $D > 2$.

To define the pairwise similarities between samples in the high and low-dimensional space, we define the conditional probability by using the local Gaussian distribution. The conditional probability in the high-dimensional space is defined as

$$p_{j|i} = \frac{\exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i}^{N} \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_k\|^2 / 2\sigma_i^2\right)}, \tag{2.10}$$

where σ_i is the variance of the local Gaussian distribution around sample $\boldsymbol{x}_i$ which is determined by binary search by using the entropy defined as

$$\log k = -\sum_{j \neq i}^{N} p_{j|i} \log p_{j|i}, \tag{2.11}$$

where k is called perplexity. Eq.(2.10) defines the local Gaussian distribution in high-dimensional space for all samples around the sample $\boldsymbol{x}_i$, and $p_{i|i}$ is set to be 0 because we are interested in only the pairwise similarities.

Let $\{\boldsymbol{y}_i | i = 1 \dots N\}$ be a set of the embedded vectors in the low-dimensional space of the samples $\{\boldsymbol{x}_i | i = 1, \dots N\}$. The vectors in the embedded low-dimensional space are represented as $\boldsymbol{y}_i = \begin{bmatrix} y_{i1} & \cdots & y_{id} \end{bmatrix}^T$ and the dimension of the low-dimensional space is much smaller than the original space as $d < D$.

Similarly, the conditional probability in the embedded low-dimensional space is defined as

$$r_{j|i} = \frac{\exp\left(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)}{\sum_{k \neq i}^{N} \exp\left(-\|\boldsymbol{y}_i - \boldsymbol{y}_k\|^2\right)}, \tag{2.12}$$

where $r_{i|i}$ is also set to be 0. Eq.(2.12) defines the local Gaussian distribution in the embedded low-dimensional space for all samples around the sample $\boldsymbol{y}_i$.

The Kullback-Leibler divergence between these conditional probabilities in the original high-dimensional space and the embedded low-dimensional space is used to measure the goodness of the embedded space and is maximized to obtain the vectors in the embedded space. The Kullback-Leibler divergence is defined as

$$C = \sum_{i}^{N} \sum_{j \neq i}^{N} p_{j|i} \log \frac{p_{j|i}}{r_{j|i}}. \tag{2.13}$$

The SNE finds the embedded vectors $\{\boldsymbol{y}_i\}$ in the low-dimensional space of the samples $\{\boldsymbol{x}_i\}$ in the high-dimensional space by minimizing the Kullback-Leibler divergence C. The update rule of $\boldsymbol{y}_i$ by the gradient decent is given as

$$\boldsymbol{y}_i^{t+1} = \boldsymbol{y}_i^{t} - \eta \frac{\partial C}{\partial \boldsymbol{y}_i} + \alpha(t)(\boldsymbol{y}_i^{t} - \boldsymbol{y}_i^{t-1}), \tag{2.14}$$

where t, η, $\alpha(t)$, and $\frac{\partial C}{\partial \boldsymbol{y}_i}$ are respectively the iteration, the learning rate, the momentum of iteration t, and the gradient defined as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = 2 \sum_{j}^{N} (p_{j|i} - r_{j|i} + p_{i|j} - r_{i|j})(\boldsymbol{y}_i - \boldsymbol{y}_j). \tag{2.15}$$

The details of derivation of Eq.(2.15) is shown in Appendix.

Hinton et al. proposed the symmetric SNE in [26]. The symmetric SNE uses joint probability instead of the conditional probability in the original SNE. The joint probability in the high-dimensional space is defined as

$$p_{ij} = \frac{1}{2}(p_i p_{j|i} + p_j p_{i|j}) = \frac{p_{j|i} + p_{i|j}}{2N}, \tag{2.16}$$

where $p_i = p_j = \frac{1}{N}$, p_{ii} is 0, and $p_{ij} = p_{ji}$ for $\forall i, j$. Similarly, the joint probability in the low-dimensional space is define as

$$r_{ij} = \frac{\exp\left(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)}{\sum_{l}^{N} \sum_{k \neq l}^{N} \exp\left(-\|\boldsymbol{y}_l - \boldsymbol{y}_k\|^2\right)}, \tag{2.17}$$

where r_{ii} is 0, and $r_{ij} = r_{ji}$ for $\forall i, j$.

Then, the Kullback-Leibler divergence is defined as

$$C = \sum_{i}^{N} \sum_{j \neq i}^{N} p_{ij} \log \frac{p_{ij}}{r_{ij}}. \tag{2.18}$$

The optimization is performed by using the same equation with Eq.(2.14) and the gradient for this case becomes more simple and is defined as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = 4 \sum_{j}^{N} (p_{ij} - r_{ij})(\boldsymbol{y}_i - \boldsymbol{y}_j). \tag{2.19}$$

The details of derivation of Eq.(2.19) is shown in Appendix.

2.3.2 t-SNE

The conditional probability or joint probability in the low-dimensional space is defined by using the local Gaussian distribution in the SNE or the symmetric SNE. However, the separation between the clusters or the samples is not enough in the embedded space because the Gaussian distribution can not give enough weights for the distant samples from the center point.

To improve this problem, the t-SNE has been proposed as an extension of the SNE. The t-SNE uses the local t-distribution in low-dimensional space instead of the local Gaussian distribution in the SNE. Since the kurtosis of the t-distribution is larger than the Gaussian distribution, it is expected that the separation between the clusters of the samples in the embedded low-dimensional space by the t-SNE can be improved than the SNE for the visualization.

The joint probability in the high-dimensional space is defined by using the local Gaussian distribution similar to the symmetric SNE. The joint probability in the embedded low-dimensional space is defined by using t-distribution as

$$r_{ij} = \frac{(1 + \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-1}}{\sum_l^N \sum_{k\neq l}^N (1 + \|\boldsymbol{y}_k - \boldsymbol{y}_l\|^2)^{-1}}, \tag{2.20}$$

where r_{ii} is 0, and $r_{ij} = r_{ji}$ for $\forall i, j$. The Kullback-Leibler divergence and the update rule for optimization are almost the same as Eq.(2.18) and Eq.(2.14). The gradient for the t-SNE is given as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = 4 \sum_j^N (p_{ij} - r_{ij})(\boldsymbol{y}_i - \boldsymbol{y}_j)(1 + \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-1}. \tag{2.21}$$

It is known that the t-SNE can produce a more understandable plot of the samples in the low-dimensional embedded space.

2.3.3 UMAP

The t-SNE is great for visualization, but is computationally time consuming when the number of samples is large. The Uniform manifold approximation and projection (UMAP) has been proposed as a non linear dimensionality reduction technique. The UMAP requires less computation time than t-SNE since it uses only a subset of samples when calculating joint probabilities in high-dimensional space. The conditional probability in high-dimensional space is defined as

$$p_{j|i} = \exp\left(- \frac{max(0, \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 - \rho_i)}{\sigma_i} \right), \tag{2.22}$$

where ρ_i is the nearest neighbor distance between one sample and i-th sample and σ_i is the variance which is determined by binary search by using the entropy defined as

$$u = 2^{\sum_{j \in \beta} p_{j|i}}, \tag{2.23}$$

where u is the number of nearest neighbor and β is a set of u nearest samples. This factor makes the UMAP faster than t-SNE in terms of computational time because it does not use all samples. To consider the symmetry, the joint probability is defined as

$$p_{ij} = p_{j|i} + p_{i|j} - pj|ipi|j \tag{2.24}$$

where $p_{ii} = 0$ and $p_{ij} = p_{ji}$ for $\forall i, j$. The joint probability in low-dimensional space is defined as

$$r_{ij} = (1 + a\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b})^{-1}, \tag{2.25}$$

where a and b are fitted by

$$\begin{aligned} &(1 + a\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b})^{-1} \approx \\ &\begin{cases} 1 & if \|\boldsymbol{y}_i - \boldsymbol{y}_j\| \leq min\_dist \\ \exp\left(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\| - min\_dist\right) & otherwise, \end{cases} \end{aligned} \tag{2.26}$$

where min_dist is a hyperparameter to control joint probability function in low dimensional space. Then the loss function is binary cross-entropy(CE) instead of Kullback-Leibler divergence defined as:

$$CE = -\sum_{i}^{N}\sum_{j \neq i}^{N} p_{ij} \log r_{ij} + (1 - p_{ij}) \log\left(1 - r_{ij}\right). \tag{2.27}$$

The gradient is derived as follows:

$$\frac{\partial CE}{\partial \boldsymbol{y}_i} = \sum_{j}^{N} \left(\frac{p_{ij}}{r_{ij}} + \frac{1 - p_{ij}}{1 - r_{ij}}\right)(\boldsymbol{y}_i - \boldsymbol{y}_j) \frac{2ab\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b-1}}{(1 + a\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b})^2}. \tag{2.28}$$

The details of derivation of Eq.(2.28) is shown in Appendix. Since the joint probability in low-dimensional space is changed by tuning min_dist, the UMAP can make various embedding and better visualization than t-SNE.

2.4 Parametric Non Linear Dimensionality Reduction

2.4.1 Parametric t-SNE

The t-SNE and UMAP can not embed the new coming samples for the embedding space like a PCA[31], because they do not construct the mapping from the high dimensional space to the low dimensional embedding space. To address this problem, the parametric t-SNE[48] has been proposed as an extension of t-SNE. The parametric t-SNE uses a feed-forward deep neural network to map the input vector in the high-dimensional space into the low-dimensional embedding space.

Let f be the deep neural network. $f(\boldsymbol{x}_i|\boldsymbol{W})$ is the output vector of deep neural network by high-dimensional sample $\boldsymbol{x}_i$, where $\boldsymbol{W}$ is weights of deep neural network. The network architecture of the original parametric t-SNE is the deep multi-layered perceptron (MLP) with only fully connected layers with sigmoid activation function. We assume that the dimension of the output vector is d, where $D > d$. The conditional probability in the high-dimensional space is defined as Eq.(2.16). The joint probability in the low-dimensional space is defined as

$$r_{ij} = \frac{(1 + \|f(\boldsymbol{x}_i|\boldsymbol{W}) - f(\boldsymbol{x}_j|\boldsymbol{W})\|^2)^{-1}}{\sum_l^N \sum_{k \neq l}^N (1 + \|f(\boldsymbol{x}_k|\boldsymbol{W}) - f(\boldsymbol{x}_l|\boldsymbol{W})\|^2)^{-1}}, \tag{2.29}$$

where r_{ii} is 0, and $r_{ij} = r_{ji}$ for $\forall i, j$.

To train the deep neural network f of the parametric t-SNE [48], a stack of restricted Boltzmann machines (RBMs) [17] is used for pre-training. After the pre-training, the parameters of the deep neural network are trained by using the training samples.

Similar with the standard t-SNE, the Kullback-Leibler divergence defined as Eq.(2.18) is used for the objective function for the optimization. The gradient for the parametric t-SNE is given as

$$\frac{\partial C}{\partial \boldsymbol{W}} = \frac{\partial C}{\partial f(\boldsymbol{x}_i|\boldsymbol{W})} \frac{\partial f(\boldsymbol{x}_i|\boldsymbol{W})}{\partial \boldsymbol{W}}, \tag{2.30}$$

where

$$\begin{aligned} \frac{\partial C}{\partial f(\boldsymbol{x}_i|\boldsymbol{W})} = {} & 4 \sum_j^N (p_{ij} - r_{ij})(f(\boldsymbol{x}_i|\boldsymbol{W}) - f(\boldsymbol{x}_j|\boldsymbol{W})) \\ & (1 + \|f(\boldsymbol{x}_i|\boldsymbol{W}) - f(\boldsymbol{x}_j|\boldsymbol{W})\|^2)^{-1}, \end{aligned} \tag{2.31}$$

and $\frac{\partial f(\boldsymbol{x}_i|\boldsymbol{W})}{\partial \boldsymbol{W}}$ is computed using the standard backpropagation learning algorithm. In the training, the parametric t-SNE uses fixed 5,000 data points for batches to prepare the conditional probability in the high-dimensional space.

2.4.2 CNN

The Convolutional Neural Network (CNN)[3] has achieved great success for image classification, image recognition, object detection, and so on. The CNN consists of several convolutional layers, pooling layers, and fully connected layers. We show the overview of CNN in Fig.2.2. The convolutional layers have the filtering with the trainable weights. Let $I \in [0, 1]^{IH \times IW}$ be an feature mapping, where IH and IW denote height and width of image respectively. A trainable weights of a convolutional

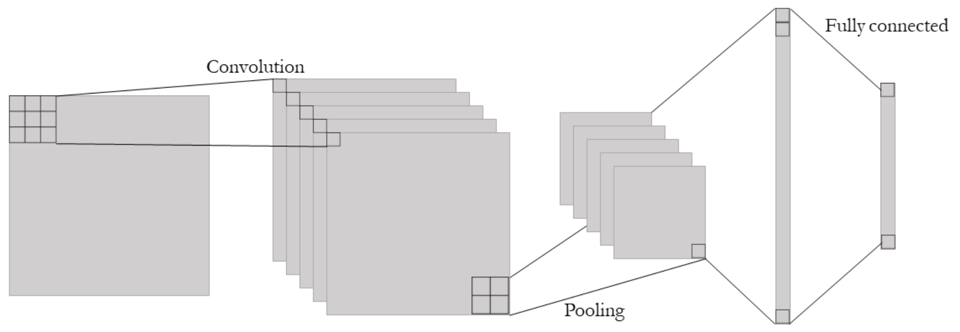


FIGURE 2.2: This figure shows overview structure of CNN. Convolution denoted convolutional layer. Pooling denotes max pooling layer. Fully connected denotes fully connected layer.

filter is defined as $w \in R^{fH \times fW}$, where fH and fW denote height and width of filter respectively. The computation of a convolutional layer is written by

$$I_{p,q}^{(L+1)} = \sum_{a=0}^{fH-1} \sum_{b=0}^{fW-1} w_{a,b} I_{p+a,q+b}^{(L)} + w_0 \tag{2.32}$$

where w_0 is bias term, $p, q, a,$ and b denote position of pixel, and L is a number of layer.

The pooling layer is called sub sampling layer. It narrows down the size of the input feature map by half. To narrow down, some techniques have been proposed. The max pooling is often used as one of them. The computation of a max pooling layer 2×2 is written by

$$\begin{aligned} I_{p,q}^{(L+1)} &= \max_{g_i \in \beta} g_i \\ \beta &= \{I_{p,q}^{(L)}, I_{p+1,q}^{(L)}, I_{p,q+1}^{(L)}, I_{p+1,q+1}^{(L)}\} \end{aligned} \tag{2.33}$$

The fully connected layer is just linear regression. A trainable weights of a fully connected layer is defined as $W \in R^{iH \times iW}$, where iH and iW denote the number of input neurons and the number of output neurons respectively. The computation of a fully connected layer is written by

$$A^{(L+1)} = W^T A^{(L)} + W_0 \tag{2.34}$$

where $A \in R^{iH}$ is the input neurons, W_0 is biases. In the deep neural network like a CNN, the activation function is necessary after every layer. The ReLU function[30] is often used for CNN. The formulation of ReLU defined as follows

$$h = \max h, 0 \tag{2.35}$$

The ReLU function prevents gradient vanish and it allows to construct deep neural networks. By using them, the CNN can extract good features from image data.

Chapter 3

Non Linear Dimensionality Reduction with q-Gaussian Distribution

3.1 q-Gaussian Stochastic Neighbor Embedding

Since the t-SNE makes better visualization than the SNE, we can understand that the embedding depends on a low-dimensional probability distribution.

However, t-SNE cannot change the low-dimensional probability distribution, because we can not control it. Therefore, t-SNE cannot change the proximity between each sample in the embedding space.

To solve this problem, we proposed to use q-Gaussian distribution[42] in low-dimensional space instead of t-distribution as an extension and improvement of t-SNE. This novel method is called q-Gaussian stochastic neighbor embedding (q-SNE) [2]. The symmetric joint probability in high-dimensional space is the same as in Eq.(2.16). The joint probability in low-dimensional space is defined as follows by using the local q-Gaussian distribution:

$$r_{ij} = \frac{(1 + \frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-\frac{1}{q-1}}}{\sum_l^N \sum_{k \neq l}^N (1 + \frac{q-1}{3-q}\|\boldsymbol{y}_l - \boldsymbol{y}_k\|^2)^{-\frac{1}{q-1}}}, \tag{3.1}$$

where q is the hyperparameter of q-Gaussian distribution, $r_{ii} = 0$, and $r_{ij} = r_{ji}$ for $\forall i, j$. The Kullback-Leibler divergence is same as Eq.(2.13) The gradient is derived as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = \frac{4}{3-q} \sum_j^N (p_{ij} - r_{ij})(\boldsymbol{y}_i - \boldsymbol{y}_j)(1 + \frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-1}. \tag{3.2}$$

The details of derivation of Eq.(3.2) is shown in Appendix. Since the q-SNE uses the q-Gaussian distribution for low-dimensional probability distribution instead of t-distribution of t-SNE, the q-Gaussian distribution can express various probability distributions by tuning the hyperparameter q. Therefore, q-SNE can provide more various embedding and visualizations than t-SNE. We can find the best visualization by choosing hyperparameter q. We show the effectiveness of q-SNE in experiments.

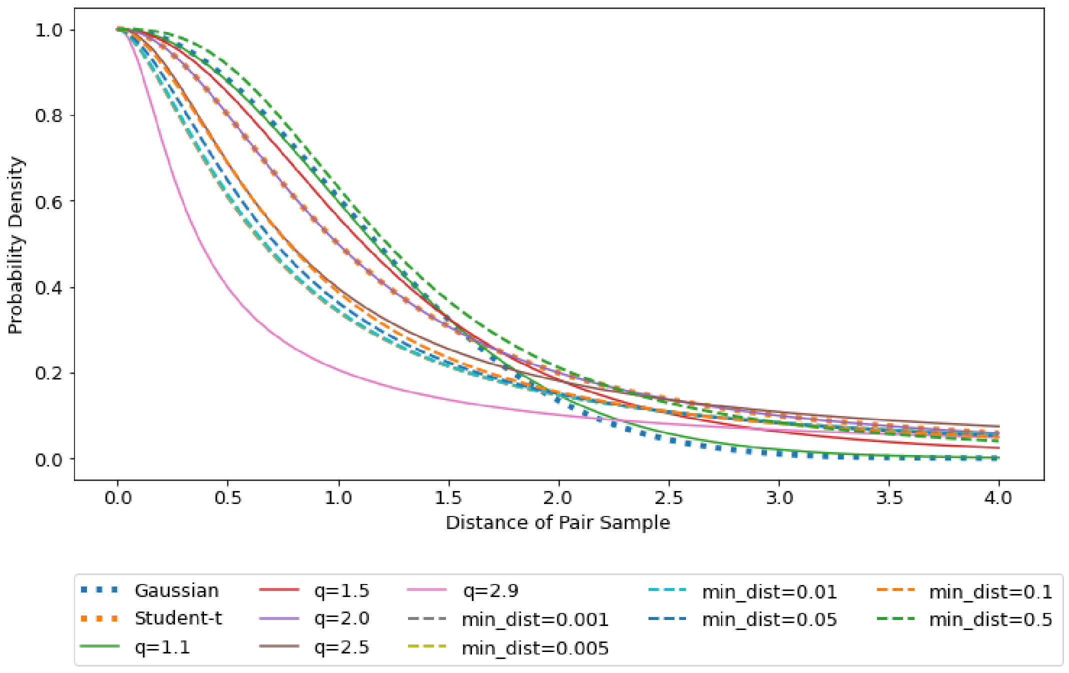


FIGURE 3.1: This figure shows the graphs about Gaussian distribution, t-distribution (dotted line), q-Gaussian distribution (solid line) and the joint probability function of Eq.(2.25) of UMAP (dashed line) by setting various hyperparameter q and min_dist respectively. When $q \to 1$, the graph is close to usual Gaussian distribution (green solid line and blue dashed line). When $q = 2$, the graph is close to t-distribution of degrees of freedom 1 (purple solid line and orange dashed line).

3.2 q-Gaussian Distributed Uniform Manifold Approximation and Projection

The UMAP can perform faster than t-SNE and can control low-dimensional space by changing the shape of the curve in low-dimensional space. However, the joint probability function Eq.(2.25) can not be determined intuitively because parameters a and b are fitted by hyperparameter min_dist, and it is difficult to control the embedding results. In Fig.3.1, we show the set of q-Gaussian distributions by setting various values of the hyperparameter q, and the joint probability function Eq.(2.25) of UMAP with parameter a and b fitted by setting various hyperparameter min_dist. According to this figure, the q-Gaussian distribution is smoothly changed by setting q and we can determine the shape of the probability distribution intuitively. In contrast, the shapes of most curves in the function Eq.(2.25) of UMAP are similar (except green dashed line), it is difficult to determine their shapes intuitively. Also we can not know the shapes of curve before fitting a and b.

To solve this problem, we propose to use q-Gaussian distribution in low-dimensional space instead of the curve of UMAP. This novel technique is called q-Gaussian distributed uniform manifold approximation and projection (q-UMAP).

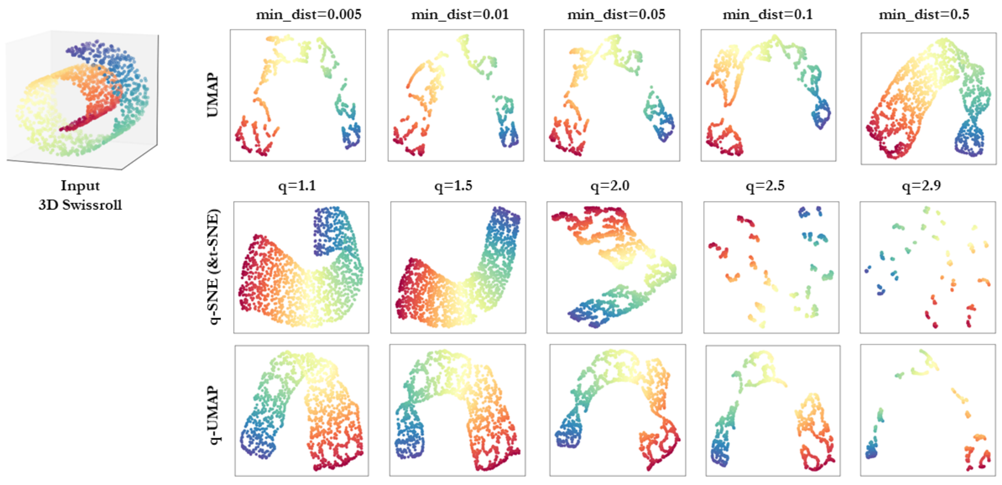


FIGURE 3.2: This figure shows embedding of swissroll dataset by using q-SNE, q-UMAP, t-SNE, and UMAP with each parameter. When $q = 2.0$ of q-SNE, the embedding is same as t-SNE. The most of left shows 3-d mapping of swiss roll dataset. The perplexity for q-SNE and t-SNE is 30. The number of nearest neighbor for q-UMAP and UMAP is 15.

The joint probability in high-dimensional space is the same as Eq.(2.24). The joint probability in low-dimensional space is defined as follows:

$$r_{ij} = \left(1 + \frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)^{-\frac{1}{q-1}}, \tag{3.3}$$

where q is hyperparameter of q-Gaussian distribution. Then, the loss function is the same binary cross-entropy (CE) as in Eq.(2.27). The gradient is derived as

$$\frac{\partial CE}{\partial \boldsymbol{y}_i} = \frac{2}{3-q}\sum_j^N \left(\frac{p_{ij}}{r_{ij}} + \frac{1-p_{ij}}{1-r_{ij}}\right)(\boldsymbol{y}_i - \boldsymbol{y}_j)\left(1 + \frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)^{-\frac{1}{q-1}-1}. \tag{3.4}$$

The details of derivation of Eq.(3.4) is shown in Appendix. Since the q-Gaussian distribution is used in low-dimensional space, the fitting part of UMAP for parameters a and b is not necessary. The q-UMAP can control the embedding result intuitively with less computational time than UMAP by using the q-Gaussian distribution. We show the effectiveness of q-UMAP in experiments.

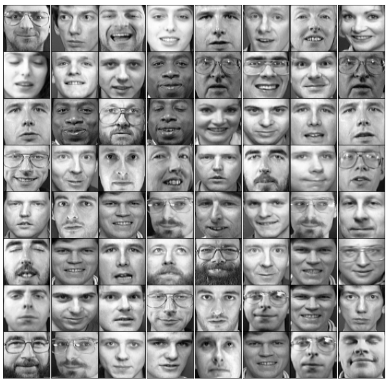
3.3 Experiments

3.3.1 Preliminary Experiment using Swissroll Data

To show the effectiveness of dimensionality reduction using q-Gaussian distribution, we have done experiments using the swissroll dataset. The swissroll dataset is 3D

(A) MNIST

(B) COIL-20


(C) OlivettiFaces

(D) FashionMNIST

FIGURE 3.3: This Figure shows images of each dataset MNIST, COIL-20, OlivettiFaces, and FashionMNIST.

data and includes 1500 data points. We embedded swissroll data on 2-dimensional space by UMAP, q-SNE, t-SNE, and q-UMAP. For UMAP, we set the hyperparameter *min_dist* to 0.005, 0.01, 0.05, 0.1, and 0.5. For q-SNE and q-UMAP, we set the hyperparameter q to 1.1, 1.5, 2.0, 2.5, and 2.9. When $q = 2.0$ of q-SNE, it denotes t-SNE. For q-SNE and t-SNE, we set the perplexity to 30. For q-UMAP and UMAP, we set the number of nearest neighbor to 15. The experimental setting of q-SNE and t-SNE is below. To obtain the embedded low-dimensional vectors of each sample, the update rule is applied 1000 times starting from the random initial vectors in the optimization. The learning rate is set to 200 and the momentum is controlled such that it is set to 0.5 for the first 250 iterations and 0.8 for the remaining iterations. To speed up the optimization in the early stages, the joint probability in the high-dimensional space p_{ij} is multiplied 12 for the first 250 iterations. These settings are almost the same as the implementation of t-SNE in the scikit-learn [32]. The experimental setting of UMAP and q-UMAP is below. To obtain the embedded low-dimensional vectors of

		Test Accuracy (%)												
		UMAP				q-UMAP					q-SNE			
		min_dist				q					q			
	nn	0.01	0.05	0.5	1.0	1.1	1.5	2.0	2.5	pl	1.1	1.5	2.0(t-SNE)	2.5
MNIST	5	94.5	**94.6**	92.8	88.8	88.6	93.3	94.0	**94.3**	5	94.9	95.8	95.6	95.5
	50	93.0	92.8	90.8	88.3	83.0	90.9	92.2	92.7	15	88.9	95.8	95.7	95.6
	100	90.8	91.0	89.1	85.4	81.3	87.1	91.3	91.7	25	89.0	**96.1**	95.9	95.7
	200	91.1	90.5	85.3	80.3	78.3	86.3	89.8	91.1	35	88.2	95.7	95.8	95.6
	300	90.4	90.2	82.7	79.4	76.4	81.9	88.3	90.6	40	87.3	95.6	95.8	95.5
COIL-20	5	**95.9**	95.8	**95.9**	93.9	95.8	95.7	96.1	**96.4**	5	99.8	**99.9**	**99.9**	99.6
	50	90.7	90.7	90.5	89.1	88.3	89.7	90.5	91.5	15	96.6	99.4	**99.9**	**99.9**
	100	90.5	90.5	87.5	85.3	86.3	88.1	89.5	90.9	25	94.7	99.1	**99.9**	99.5
	200	90.6	90.6	86.9	83.6	84.1	86.8	89.0	91.2	35	92.7	98.8	99.7	99.5
	300	89.9	89.2	85.1	82.5	82.3	86.4	88.5	90.3	40	91.7	98.5	99.8	99.5
Olivette	5	90.6	**91.4**	91.2	86.7	86.3	90.9	**92.4**	91.8	5	92.5	94.0	92.0	89.5
	50	80.1	78.9	69.6	61.1	61.3	69.6	76.1	79.1	15	85.0	**94.5**	90.8	90.0
	100	76.1	76.7	64.5	56.2	57.2	64.9	73.2	75.0	25	80.8	90.3	92.3	89.5
	200	73.9	71.8	59.6	51.5	51.9	60.2	68.7	74.3	35	75.3	90.0	91.3	90.5
	300	71.3	69.6	58.3	52.3	51.8	59.3	66.6	74.0	40	71.3	89.3	91.8	87.8
Fashion	5	**84.2**	83.9	81.0	78.4	79.9	81.3	83.0	**83.8**	5	82.9	**86.8**	86.3	86.1
	50	81.4	81.0	78.8	77.7	78.2	79.2	80.3	81.0	15	81.0	**86.8**	86.7	86.2
	100	80.4	80.3	78.1	77.0	77.2	78.5	79.3	80.2	25	80.5	86.1	86.6	86.3
	200	79.7	79.7	77.8	76.6	76.7	77.9	79.0	79.4	35	79.4	85.1	86.6	86.3
	300	79.7	79.5	77.5	76.1	76.2	77.6	78.8	79.4	40	78.9	84.7	86.3	86.3

TABLE 3.1: This table shows classification accuracy of t-SNE, UMAP, q-UMAP, and q-SNE by using k-NN at embedding space on MNIST, COIL-20, OlivettiFaces, and FashionMNIST dataset. The nn and pl means nearest neighbor of UMAP and q-UMAP, and perplexity of t-SNE and q-SNE respectively. When $q = 2.0$ of q-SNE, the results is same as t-SNE. The bold scores denote the best score of each technique.

each sample, the update rule is applied 200 times starting from the random initial vectors in the optimization. The learning rate is set to 1.0. These settings are almost the same as the implementation of UMAP.

The embedding is shown in Fig.3.2. The visualization of 3D swissroll data is shown at most left in Fig.3.2. When $q = 2.0$ for q-SNE, the embedding is the same as t-SNE. When $q = 1.1$ for q-SNE, the embedding is close to SNE. For UMAP, we can see that the results of embedding are almost similar if the value of *min_dist* is between 0.005 and 0.1, because the shape of the curve is almost similar, as shown in Fig 3.1. It is difficult to express an embedding intuitively when *min_dist* is between 0.1 and 0.5. As the value of q increases, the shape of the distribution in Fig.3.1 becomes sharper, and the results of embedding in Fig.3.2 becomes more clustered. According to these results, q-SNE and q-UMAP can obtain various embedding results by using the characteristics of the q-Gaussian distribution for each hyperparameter q. We can

	Computational time (seconds)								
	q-UMAP								q-SNE
	nn								
	5	10	50	100	150	200	250	300	
MNIST	6.11	7.34	13.21	18.24	23.06	26.64	28.20	30.70	2033.86
COIL-20	12.93	12.95	13.30	13.51	13.65	13.90	14.42	14.57	36.84
OLivetti	1.80	1.70	1.83	1.87	1.91	1.93	1.95	1.97	3.60
Fashion	6.19	7.53	13.16	18.32	22.54	24.54	26. 59	28.33	2188.27

TABLE 3.2: This table shows comutational time of q-UMAP and q-SNE. For q-UMAP, it is measured with different hyperparameter nearest neighbor nn. The value of hyperparameter q is not taken into account because it is not involved in the computation time.

see that the embedding when $q = 1.5$ of q-SNE is the best visualization for swiss-roll. By using q-Gaussian distribution, we can intuitively create various embedding spaces.

3.3.2 Comparison Experiments using Image Dataset

In this section, we show the embedding results and classification accuracy by using k nearest neighbor (k-NN) [34] in the embedding space on MNIST [13], COIL-20 [35], OliverttiFaces [24], and FashionMNIST [50] dataset. We show the images of each dataset in Fig.3.3. To compare the embedding visualization and accuracy, we used t-SNE, UMAP, q-SNE, and q-UMAP. The MNIST dataset is the gray scale image dataset of handwritten digits from 0 to 9. It has 60,000 images and the size of each image is 28×28 pixels. For the embedding and classification, we randomly chose $10,000$ images preliminary. The COIL-20 dataset is the gray scale image dataset of 20 objects. It has $1,440$ images and the size of each image is 128×128 pixels. Each object was placed on a motorized turntable against a black background. The turntable was rotated through 360 degrees to vary object pose with respect to a fixed camera. Images of the objects were taken at pose intervals of 5 degrees. The OlivettiFaces dataset is the gray scale image dataset of 40 person. It has 400 images and the size of each image is 92×112 pixels. The FashionMNIST dataset is the gray scale image dataset of 10 fashion items. It has 60,000 images and the size of each image is 28×28 pixels. For the embedding and classification, we randomly chose 10,000 images preliminary.

First, we show the embedding of MNIST dataset by using t-SNE, UMAP, q-SNE, and q-UMAP with different hyperparameters in Fig.3.4. The experimental settings of UMAP, q-SNE, and q-UMAP are same as preliminary experiments. We can know the relationship between hyperparameter n_neighbors and *min_dist*, perplexity and q, or n_neigbors and q. According to Fig.3.4.(a), it is difficult to adjust embedding space because the figures in the top two rows or the bottom two rows make little

TABLE 3.3: Top 29 words around "man" in order from closer words by using q-SNE with each parameter q.

$q = 1.1$	$q = 1.5$	$q = 2.0$	$q = 2.5$
remembered	old	boy	woman
faculty	woman	woman	boy
teaches	boy	old	girl
pride	girl	girl	old
visits	died	childhood	her
retire	herself	life	she
kid	survived	family	herself
theology	her	dying	blind
student	surviving	die	drunk
marines	life	boys	himself
world	family	girls	him
guard	she	men	his
loves	dying	alive	couple
taught	childhood	women	alien
undergraduate	independent	athletes	life
joy	die	lovers	family
likes	drunk	gods	lived
arrives	blind	beloved	lives
her	represents	prophet	living
herself	represent	loving	childhood
territories	lived	jesus	dying
command	representing	lived	die
sacred	represented	divine	child
saturday	lives	lives	families
lived	chose	child	children
ritual	living	living	parents
love	god	christ	elderly
recalled	masses	god	mothers
economics	evil	survived	male

difference, respectively. According to Fig.3.4.(b) and 3.4.(c), we can know the gradually changing the embedding by changing hyperparameter q. We can easily and intuitively control the embedding by using q-Gaussian distribution characteristics.

We also show the other embedding of COIL-20, OlivettiFaces, and FashionMNIST by using t-SNE, UMAP, q-SNE and q-UMAP with different hyperparameters in Fig.3.5, Fig.3.6, and Fig.3.7 respectively. In Fig.3.6, we have shown images of 20 persons in OlivettiFaces.

Next, we show the embedding of MNIST, COIL-20, OlivettiFaces, and Fashion-MNIST by using PCA, Isomap, t-SNE, UMAP, q-SNE, and q-UMAP in Fig.3.8. Also we show the classification accuracy by using k-NN at embedding space by t-SNE, UMAP, q-SNE and q-UMAP in Table.3.1. The k of k-NN is 5. The classification accuracy is averaged 5 trials with different seeds. In Fig.3.8, the hyperparameters for each embedding are used when classification accuracy of the Table3.1 is the best.

TABLE 3.4: Top 29 words around "man" in order from closer words by using q-UMAP with each parameter q.

$q = 1.1$	$q = 1.5$	$q = 2.0$	$q = 2.5$
ambulance	must	event	men
cats	circumstances	milestone	woman
u.n.	fantastic	inaugural	boy
animals	demise	sacrifice	kid
marble	pressed	landmark	kids
animal	thoroughly	advisor	women
erected	impossible	hosted	girl
walls	crush	hosts	children
cow	momentum	hosting	ladies
communist	thought	host	girls
deer	supicious	occasion	child
tall	non-families	adviser	boys
parks	send	historic	female
dog	properly	ritual	male
door	absolutely	aide	females
d.c.	amazing	aides	males
dynasty	rid	advisers	parents
grand	wonderful	mark	adults
insects	exciting	marked	youth
plastic	will	senior	teens
title	differently	breakthrough	young
yard	unless	replacement	youngsters
dogs	play	woman	adult
pet	poison	marking	baby
asia	incredible	replacing	infant
oldest	repeat	replace	teenagers
sheep	should	mourning	herself
competing	bang	forthcoming	teen
glass	chances	occasions	aging

According to Fig.3.8, q-SNE are embedded more tightly together than t-SNE. The embedding of q-UMAP becomes almost same as the embedding of UMAP, however q-UMAP can control intuitively than UMAP according to Fig.3.4. Table3.1 shows that q-SNE scores better than the other methods on all datasets. When considering the pairs of all samples in high-dimensional space like q-SNE, it is better that the sample is embedded so that it spreads by choosing lower q. When considering the pairs of some samples in high-dimensional space like q-UMAP, it is better that the sample is embedded so that it solidifies by choosing higher q. For UMAP and q-UMAP, the hyperparmeter min_dist$=5$ is the best for all cases. Table3.2 shows computational time of q-UMAP and q-SNE for each dataset. The times are also averaged 5 trials with different seeds. According to this table, q-UMAP can give faster computational time.

The q-SNE can give the higher score than q-UMAP, because it uses all samples in high-dimensional space. The q-UMAP can give the faster than q-SNE, because it uses some samples in high-dimensional space. This indicates that the best classification can be selected by setting the hyperparameter q of the q-Gaussian distribution. The advantage of our proposed method is that it can provide an embedding space with high accuracy even if it is small, and the space can be controlled in various ways.

3.3.3 Embedding using Word Dataset

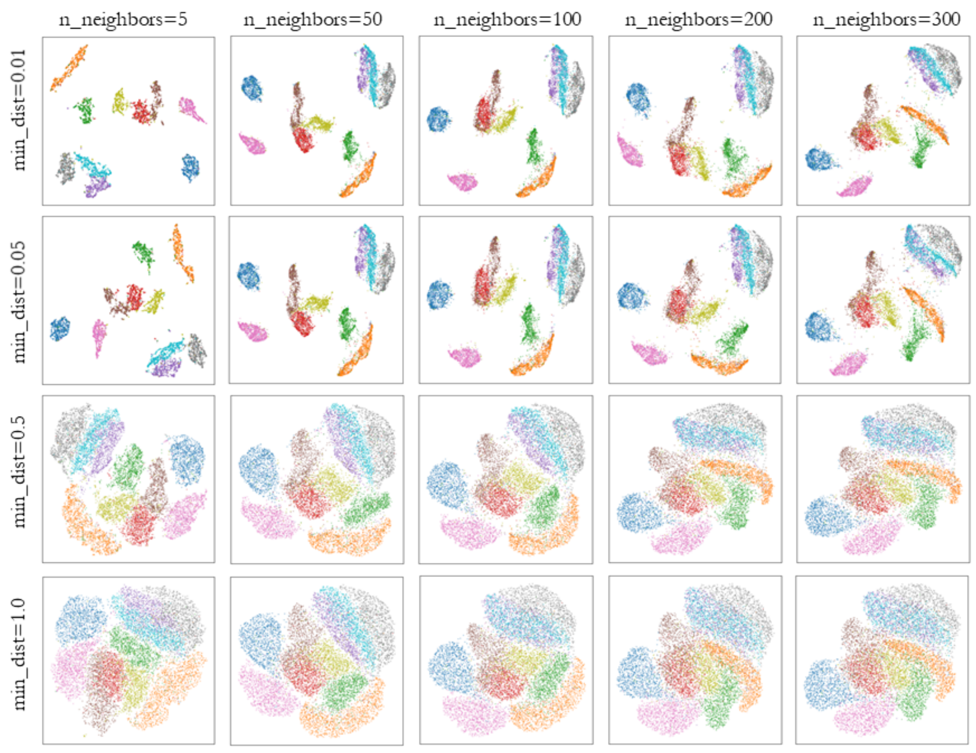
We did the experiments of embedding using word dataset. The word is converted to vectors using a technique called word2vec [9]. It is important to analysis the similarity between each words. The GloVe dataset [33] is one of the word dataset. The GloVe dataset contains 400,000 words which has a 300 dimensions vector. For our experiments, we randomly chose 10,000 words in the dataset. We show the embedding of q-SNE and q-UMAP with several hyperparameter q. The hyperparameter perplexity of q-SNE is 30. The hyperparameter n_neighbor of q-UMAP is 5. Other experimental settings of q-SNE, and q-UMAP are same as preliminary experiments.

We show the 30 words close to the word "man" in the 2-dimensional embedded space for the cases with parameters $q = 1.1$, $q = 1.5$, $q = 2.0$, and $q = 2.5$ in Fig.3.9. To confirm the 30words around "man", we show them in Table3.3 for q-SNE and Table3.4 for q-UMAP. According to embedding, when hyperparameter q increases, each words becomes more cluster. The words around "man" are better when hyperparameter q is higher in both thecniques. For q-SNE, the words related to people appeared than q-UMAP.

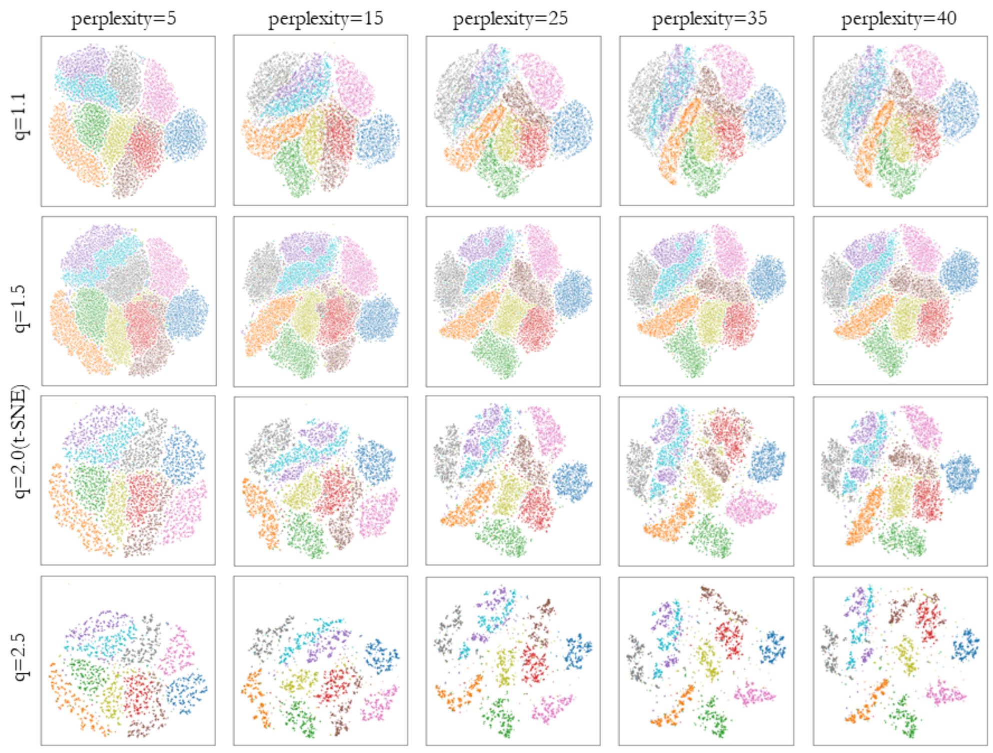
3.3.4 Locus of Embedding by Changing Hyperparameter q

We did experiments of locus of embedding by changing hyperparameter q. To confirm it, we used MNIST, COIL-20, and FashionMNIST. For q-SNE, experimental settings are same as comparison experiments using image dataset. We show the locus of q-SNE embedding when $q = 1.1$, $q = 2.0$, and $q = 2.9$ in Fig.3.10 on MNIST. These embedding are plotted at the same time, and we show the locus by arrow. The arrows are used for random 25 samples. According to this figure, we can know that q-SNE expanded the embedding space when q increases, and made more cluster after expansion. For COIL-20 and FashionMNIST, we show the locus in Fig.3.11 and Fig.3.12 respectively. For these cases, the same phenomena is also observed as MNIST. We open the animation of locus for each dataset in my GitHub (https://github.com/i13abe/qSNE). Also we show the locus of q-UMAP embedding when $q = 1.1$, $q = 2.0$, and $q = 2.9$ in Fig.3.13 on MNIST. These embedding are also plotted at the same time, and we show the locus by arrow. The arrows are used for random 25 samples. According to this figure, we can know that q-UMAP slightly expanded the embedding space than q-SNE when q increases, and made more cluster after expansion. For COIL-20 and FashionMNIST, we show the locus in Fig.3.14

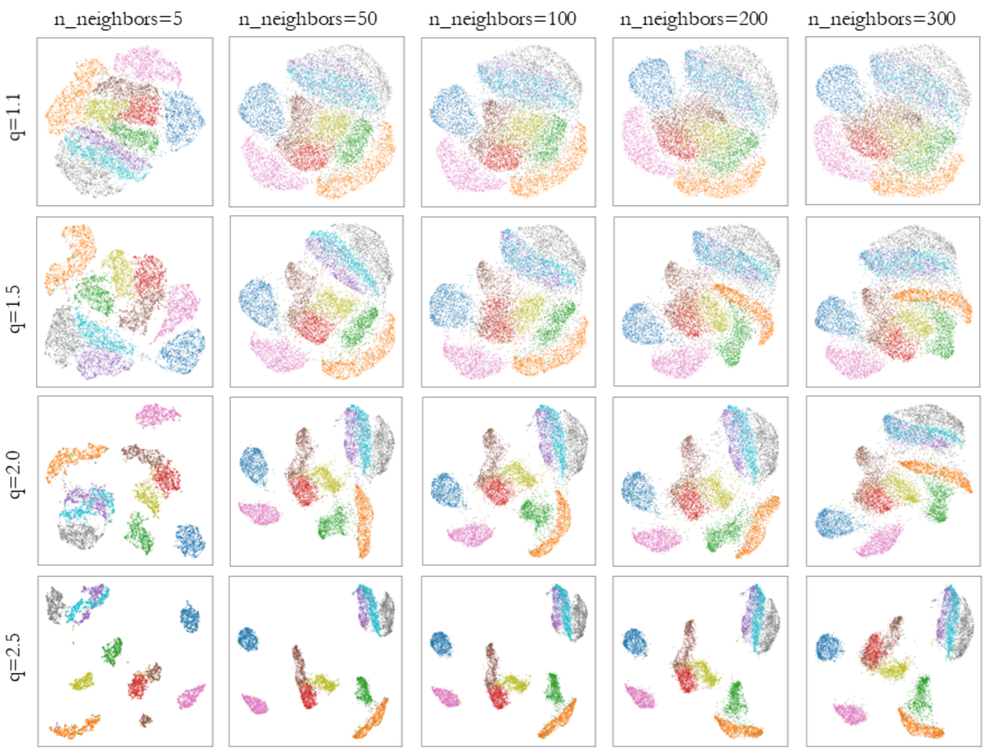
and Fig.3.15 respectively. For these cases, the same phenomena is also observed as MNIST.



(a) UMAP

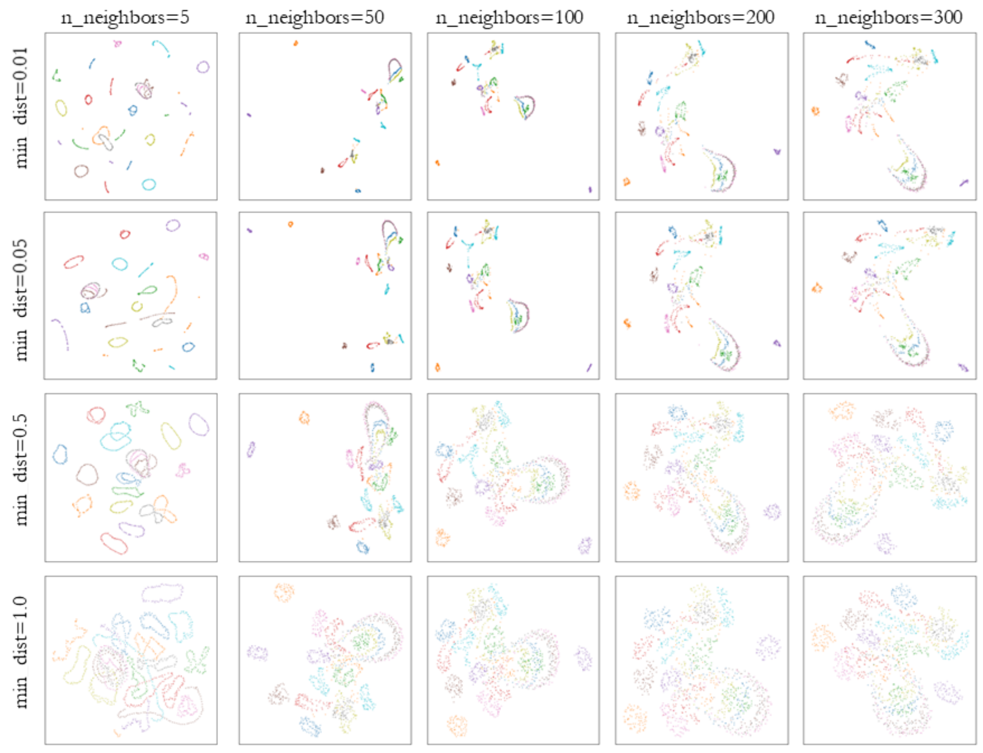


(b) q-SNE (t-SNE when $q = 2.0$)

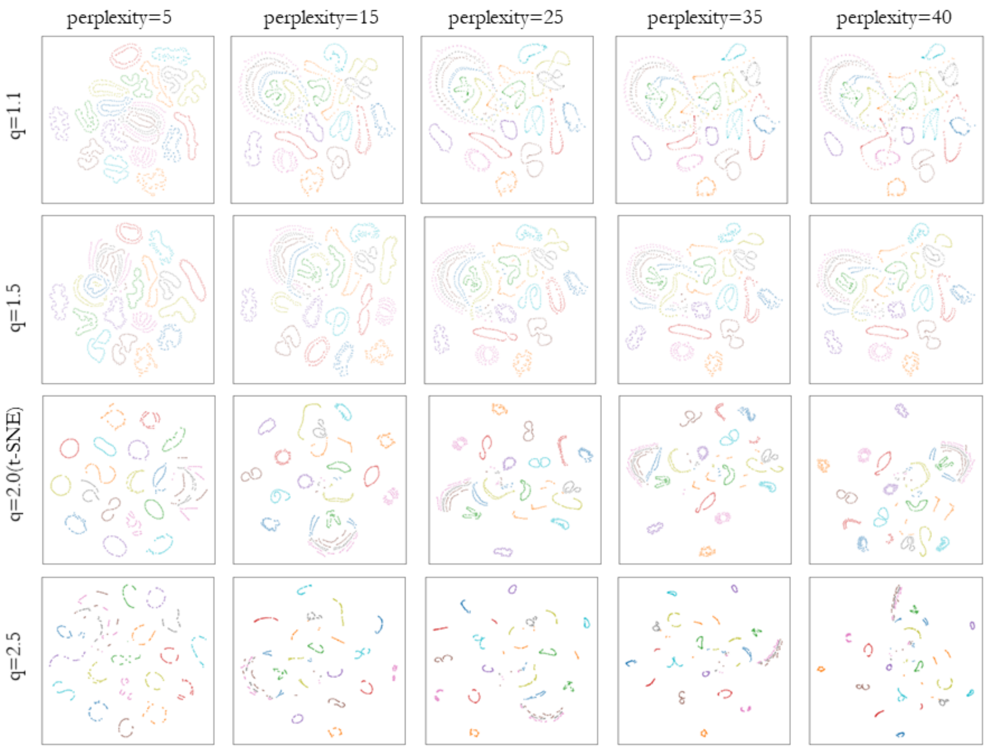


(c) q-UMAP

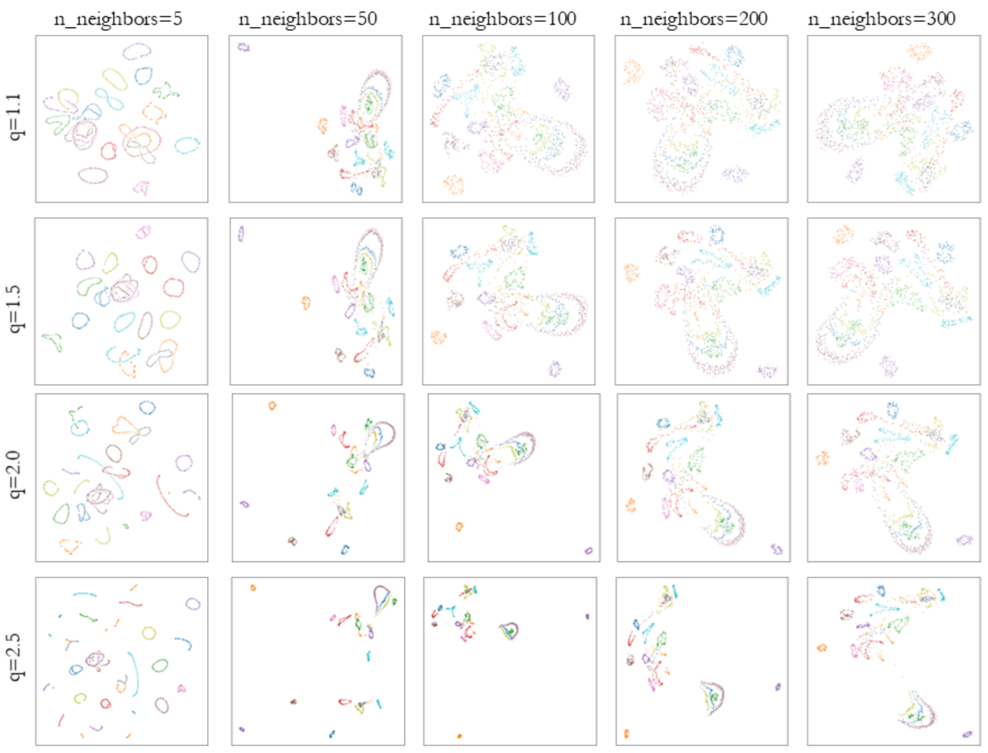
FIGURE 3.4: This figure shows embedding of MNIST dataset by using (a)UMAP, (b)q-SNE(t-SNE), and (c)q-UMAP with different hyperparameter. When $q = 2.0$ of q-SNE, the embedding is same as t-SNE. The n_neighbors, perplexity, min_dist, and q are hyperparameter.



(a) UMAP

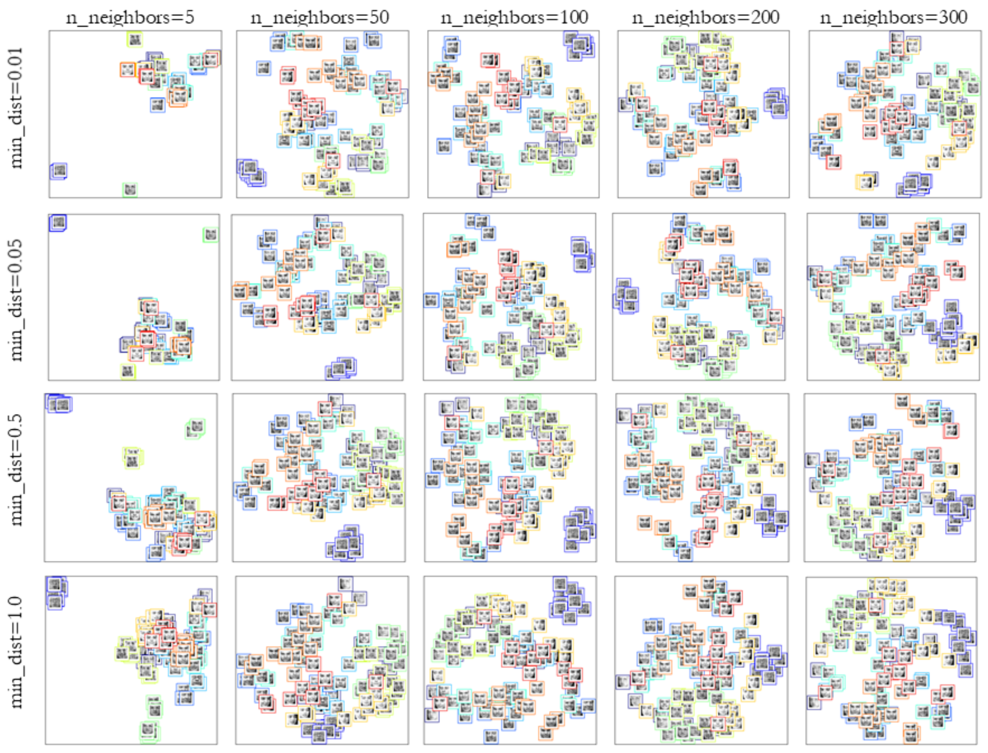


(b) q-SNE (t-SNE when $q = 2.0$)

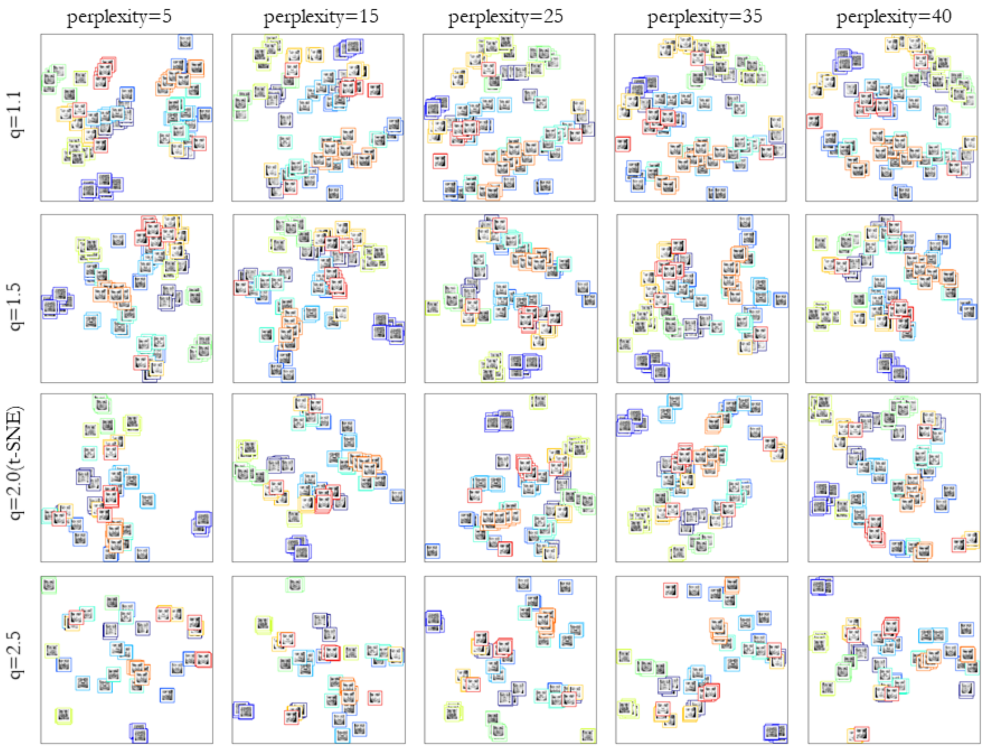


(c) q-UMAP

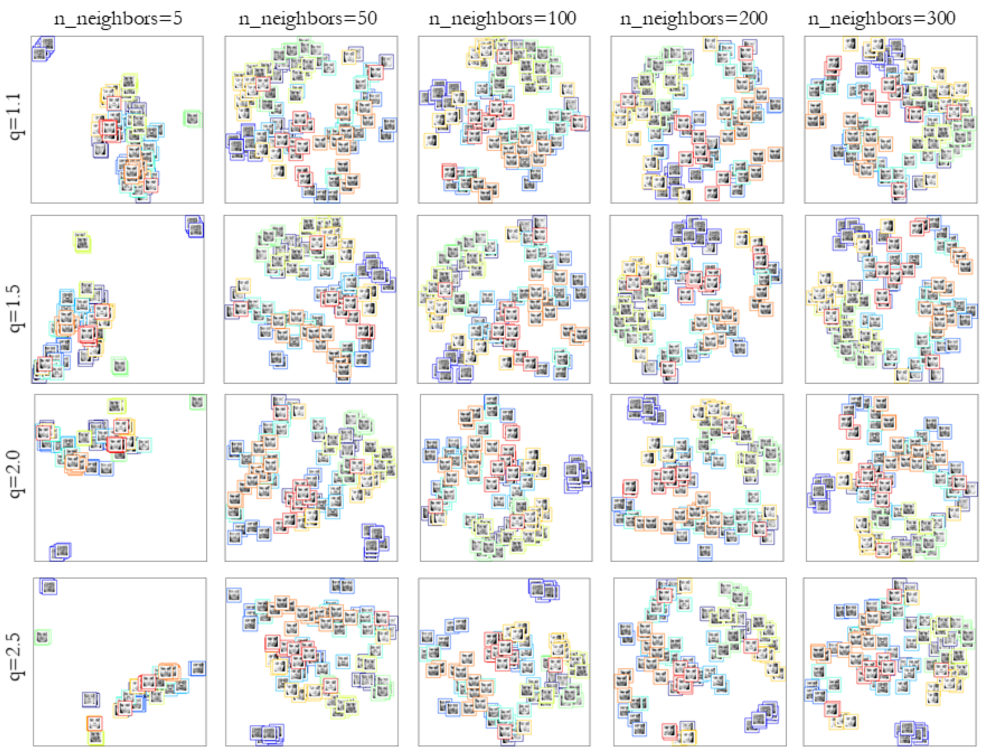
FIGURE 3.5: This figure shows embedding of COIL-20 dataset by using (a)UMAP, (b)q-SNE(t-SNE), and (c)q-UMAP with different hyperparameter. When $q = 2.0$ of q-SNE, the embedding is same as t-SNE. The n_neighbors, perplexity, *min_dist*, and q are hyperparameter.



(a) UMAP

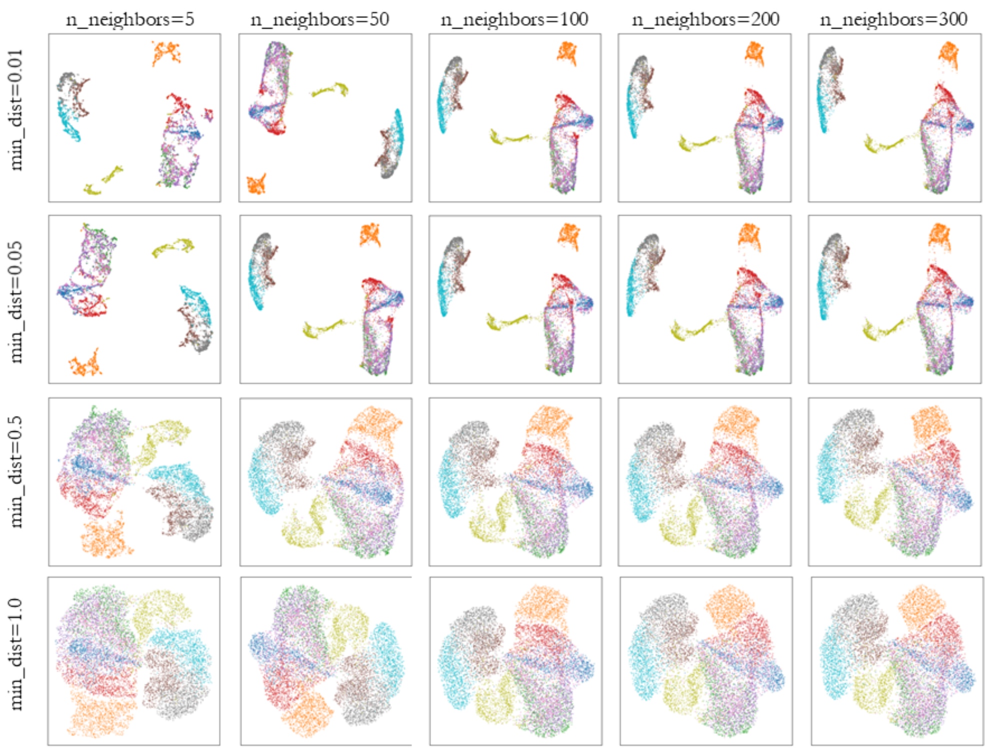


(b) q-SNE (t-SNE when $q = 2.0$)

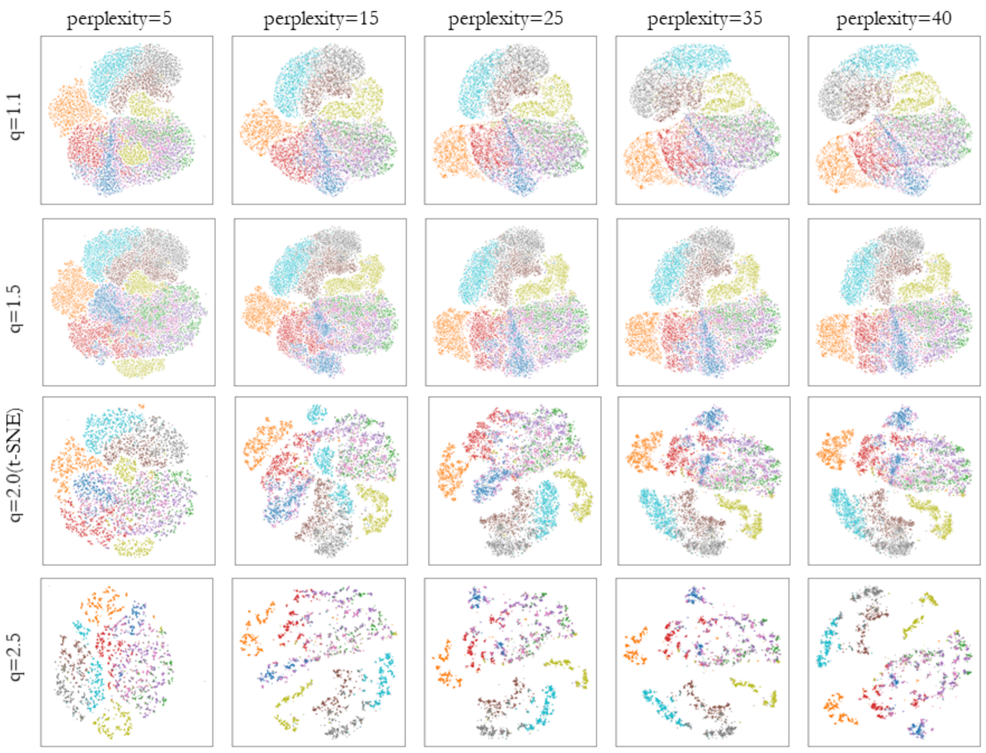


(c) q-UMAP

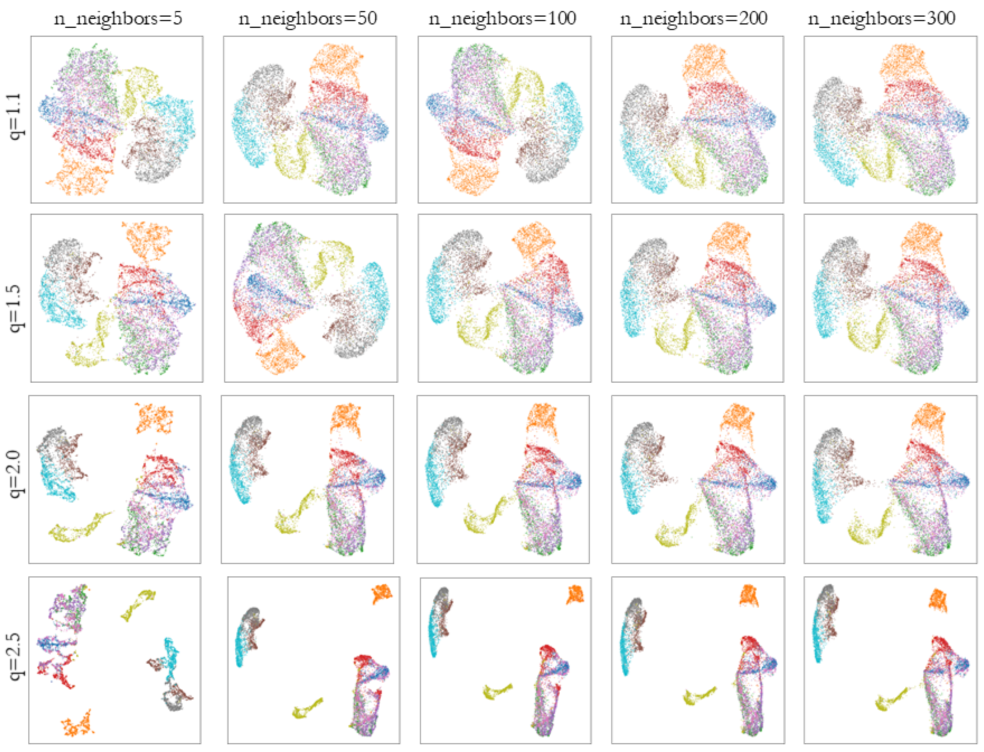
FIGURE 3.6: This figure shows embedding of OlivettiFaces dataset by using (a)UMAP, (b)q-SNE(t-SNE), and (c)q-UMAP with different hyperparameter. When $q = 2.0$ of q-SNE, the embedding is same as t-SNE. The n_neighbors, perplexity, *min_dist*, and q are hyperparameter.



(a) UMAP



(b) q-SNE (t-SNE when $q = 2.0$)



(c) q-UMAP

FIGURE 3.7: This figure shows embedding of FashionMNIST dataset by using (a)UMAP, (b)q-SNE(t-SNE), and (c)q-UMAP with different hyperparameter. When $q = 2.0$ of q-SNE, the embedding is same as t-SNE. The n_neighbors, perplexity, *min_dist*, and q are hyperparameter.

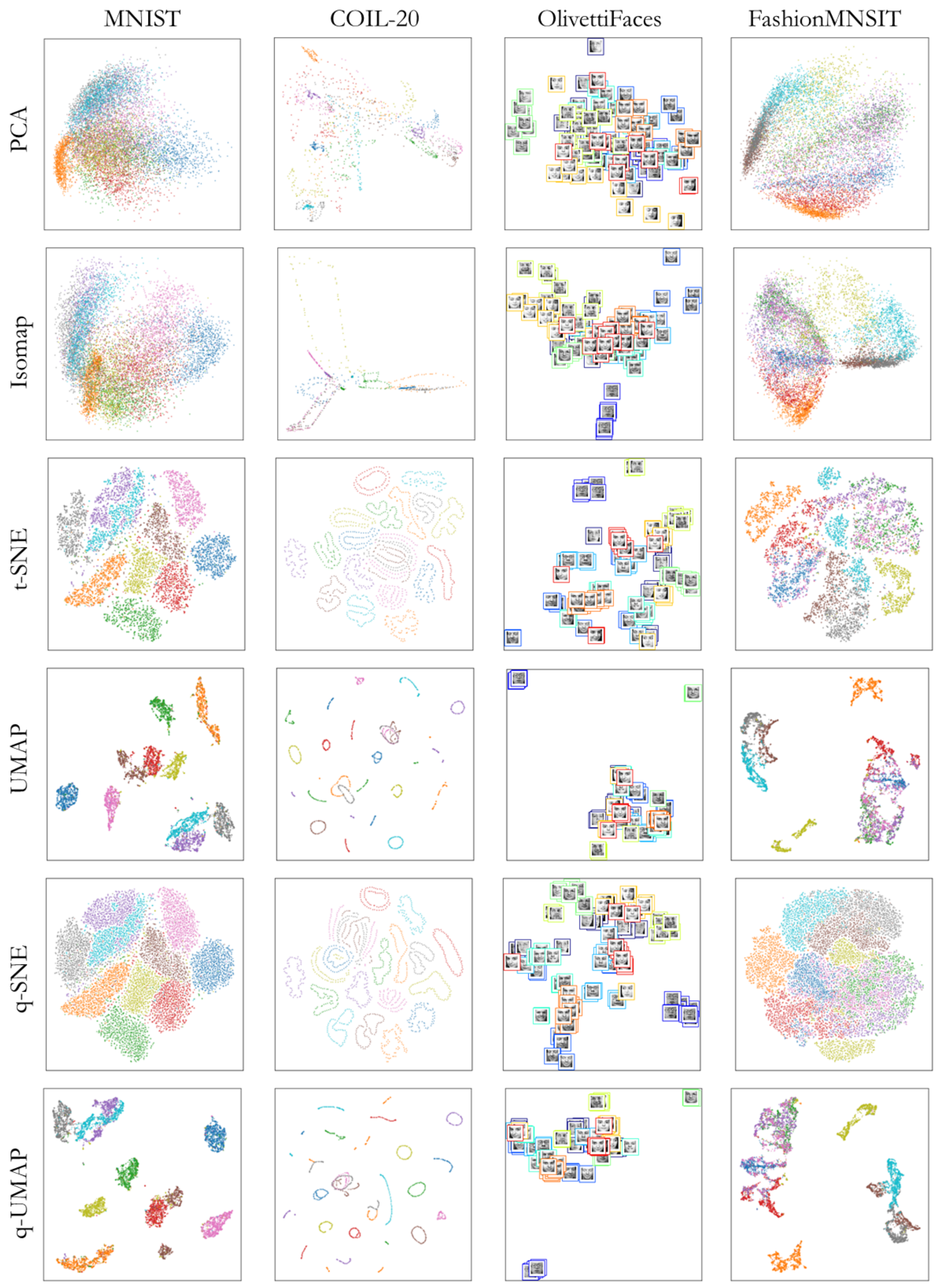
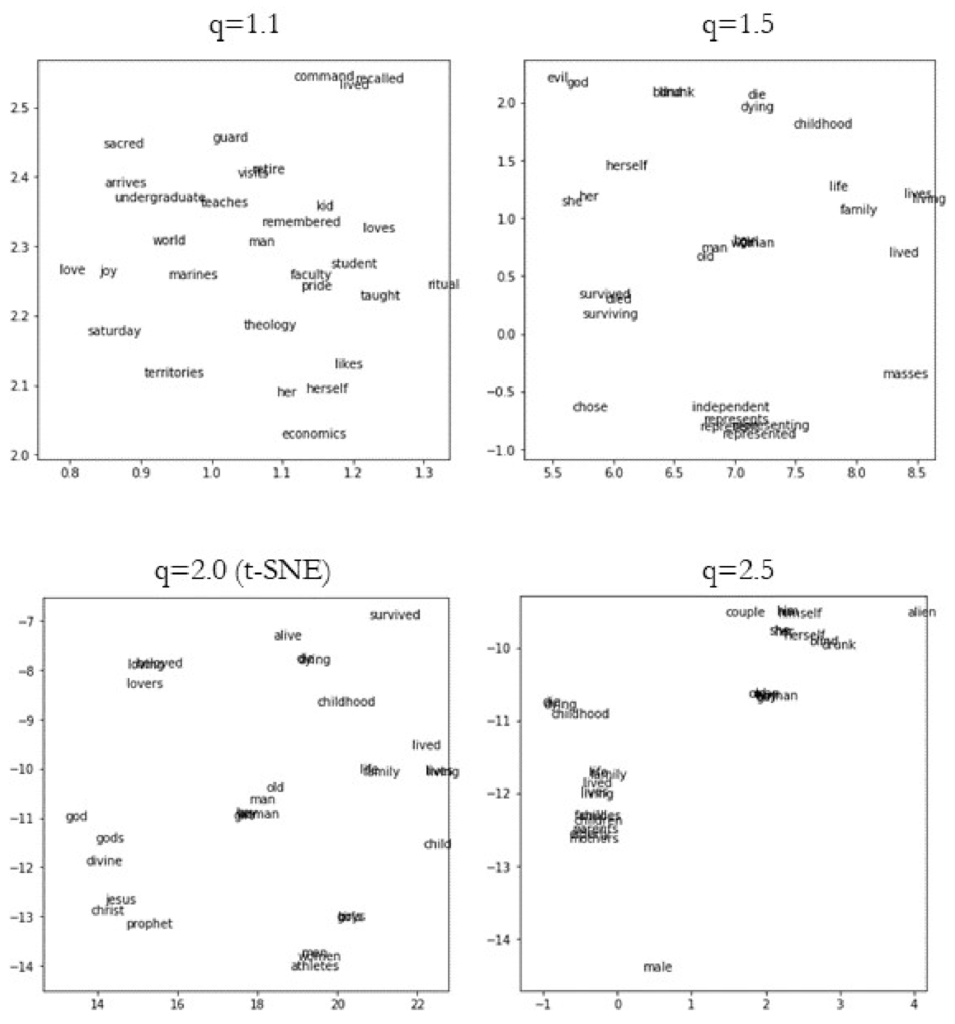
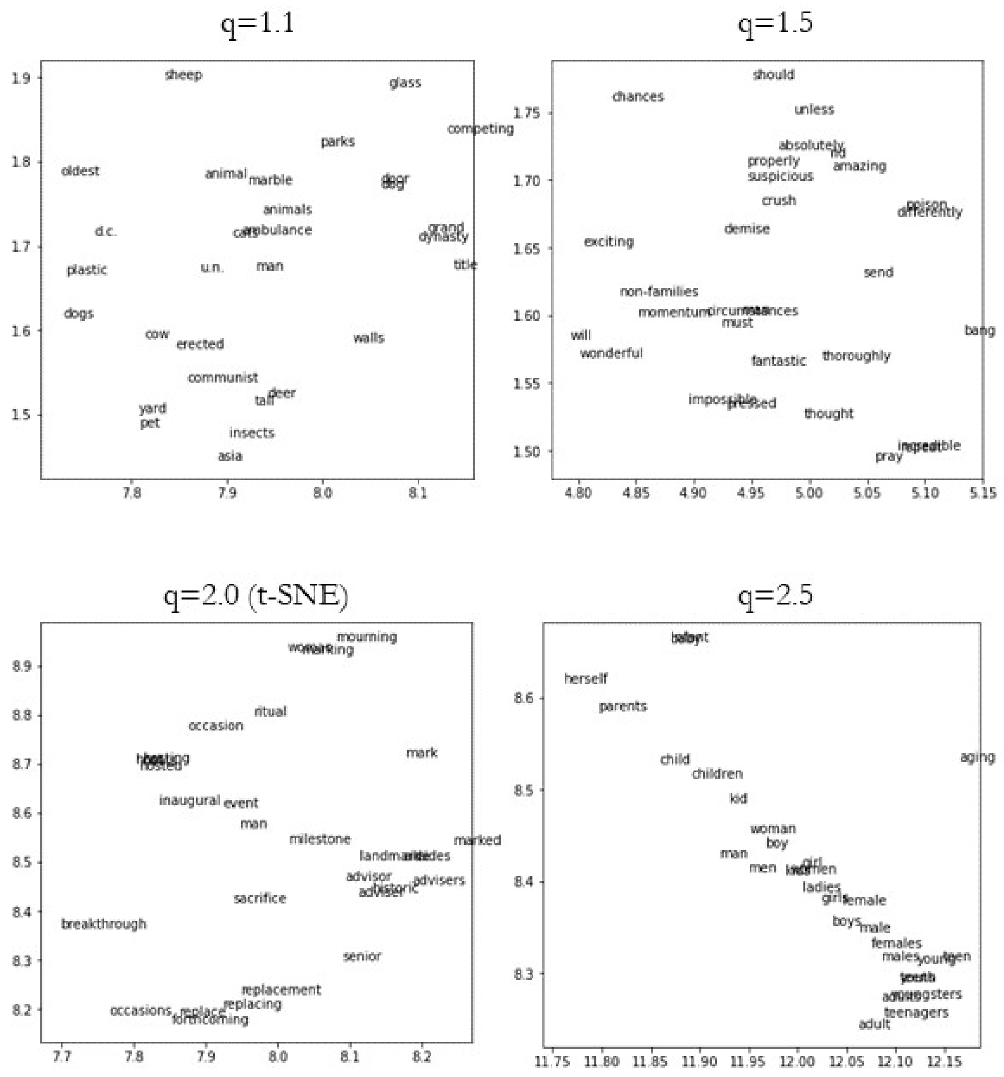


FIGURE 3.8: This figure shows embedding of MNIST, COIL-20, OlivettiFaces, and FashionMNIST dataset by using PCA, Isomap, t-SNE, UMAP, q-SNE, and q-UMAP. The hyperparameters for each embedding of t-SNE, UMAP, q-SNE and q-UMAP are used when the classification accuracy of the Table3.1 is the best.



(a) q-SNE



(b) q-UMAP

FIGURE 3.9: This figure shows embedding of GloVe dataset by using (a)q-SNE, and (b)q-UMAP with different hyperparameter q. When $q = 2.0$ of q-SNE, the embedding is same as t-SNE. These embedding shows 30 word around "man".

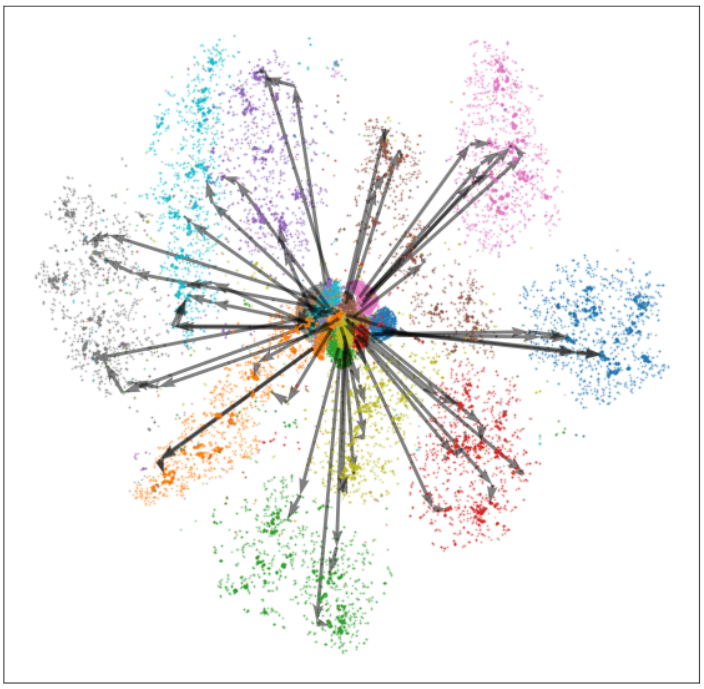


FIGURE 3.10: This figure shows locus of q-SNE embedding on MNIST dataset with hyperparameter $q = 1.1$, $q = 2.0$, and $q = 2.9$. These embedding are plotted at the same time, and locus is written by arrow.

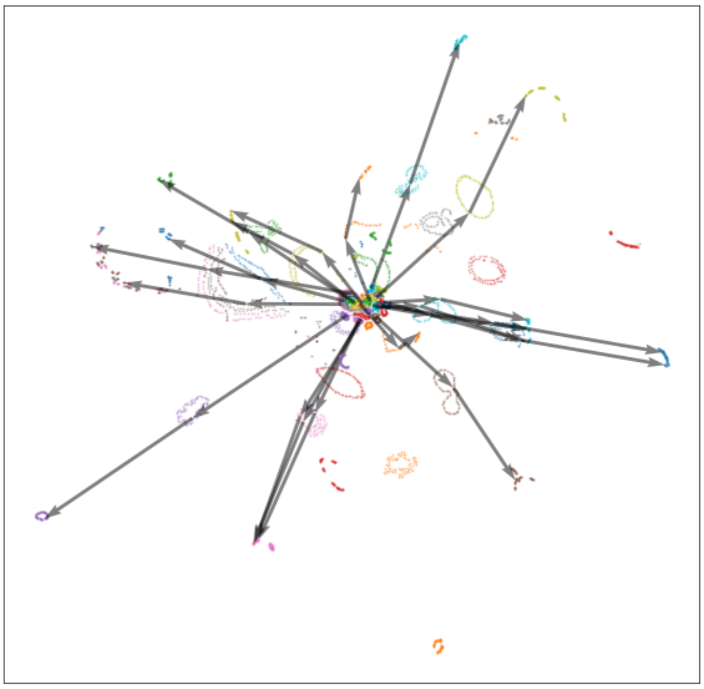


FIGURE 3.11: This figure shows locus of q-SNE embedding on COIL-20 dataset with hyperparameter $q = 1.1$, $q = 2.0$, and $q = 2.9$. These embedding are plotted at the same time, and locus is written by arrow.

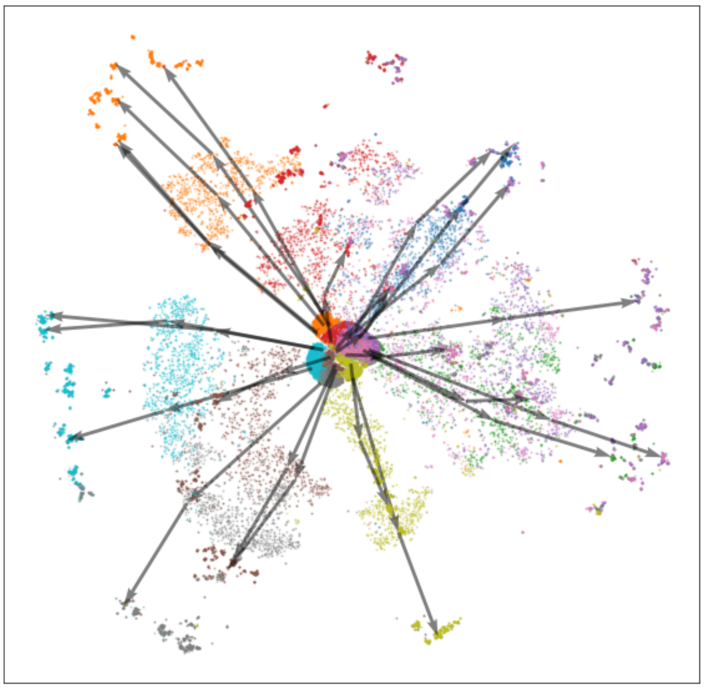


FIGURE 3.12: This figure shows locus of q-SNE embedding on FashionMNIST dataset with hyperparameter $q = 1.1$, $q = 2.0$, and $q = 2.9$. These embedding are plotted at the same time, and locus is written by arrow.

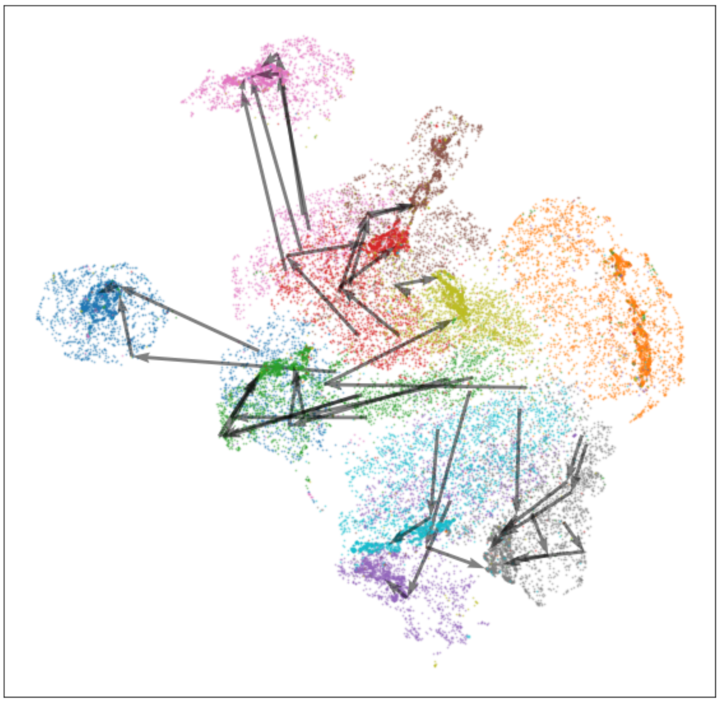


FIGURE 3.13: This figure shows locus of q-UMAP embedding on MNIST dataset with hyperparameter $q = 1.1$, $q = 2.0$, and $q = 2.9$. These embedding are plotted at the same time, and locus is written by arrow.

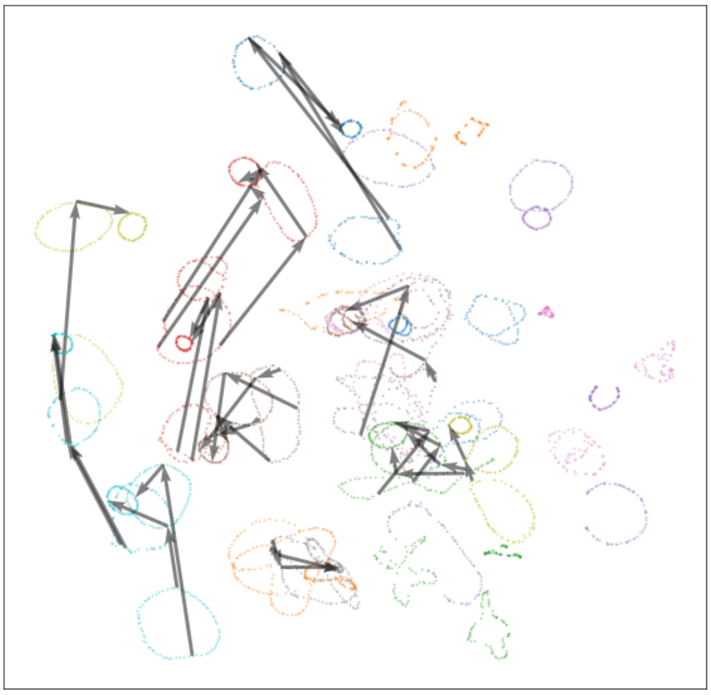


FIGURE 3.14: This figure shows locus of q-UMAP embedding on COIL-20 dataset with hyperparameter $q = 1.1$, $q = 2.0$, and $q = 2.9$. These embedding are plotted at the same time, and locus is written by arrow.

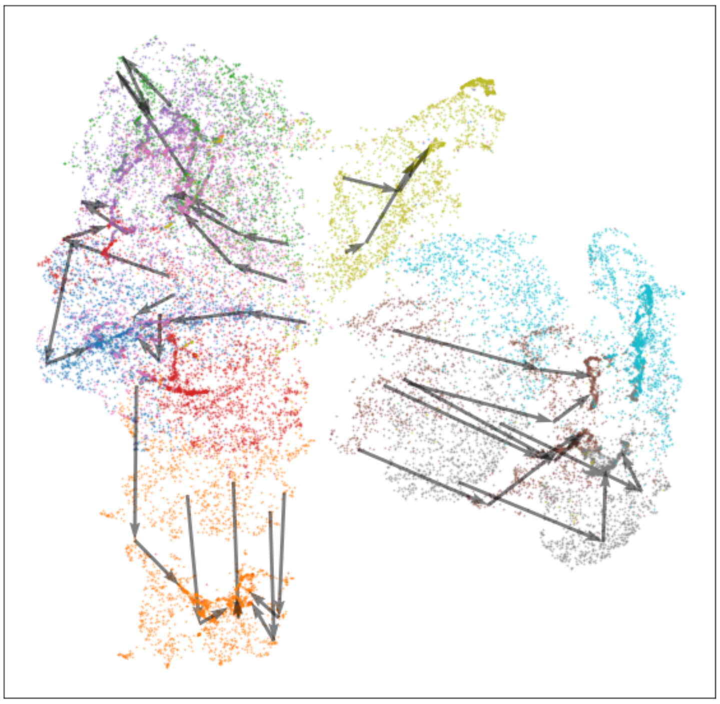


FIGURE 3.15: This figure shows locus of q-UMAP embedding on FashionMNIST dataset with hyperparameter $q = 1.1$, $q = 2.0$, and $q = 2.9$. These embedding are plotted at the same time, and locus is written by arrow.

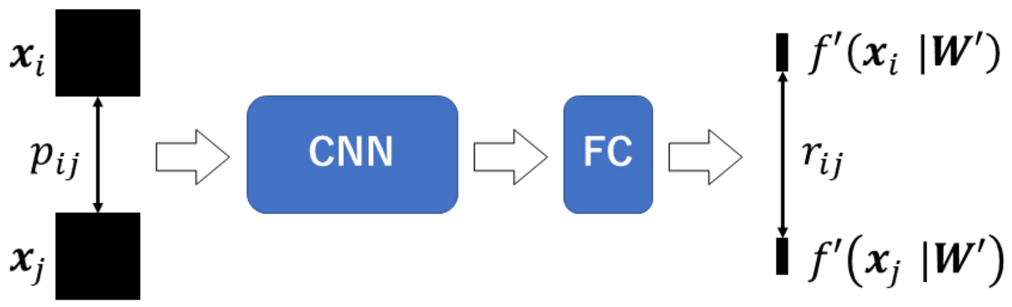


FIGURE 4.1: This figure shows an architecture of parametric q-SNE with a convolutional neural network. FC denotes a fully connected layer.

Chapter 4

Non Linear Parametric Dimensionality Reduction with q-Gaussian Distribution

4.1 Parametric q-SNE with CNN

We propose a novel technique called the parametric q-Gaussian distributed stochastic neighbor embedding (the parametric q-SNE) with a convolutional neural network (CNN). The parametric q-SNE uses q-Gaussian distribution instead of t-distribution of the parametric t-SNE. It can map a new samples in the high-dimensional space into the low-dimensional embedding space by using CNN.

Let f' be the CNN with the ReLU function. The ReLU function is a non-linear activation function. Then the output vector of the CNN for the sample $\boldsymbol{x}_i$ in the high-dimensional space is represented by $f'(\boldsymbol{x}_i|\boldsymbol{W}')$, where $\boldsymbol{W}'$ is weights of the CNN. We assume that the dimension of output vector d where $D > d$. The conditional probability in the high-dimensional space is defined as Eq.(2.16). The joint probability in

TABLE 4.1: This table shows the construction of CNN for experiments. The activation function is ReLU.

Layer	Operator	Output Channels
1	Convolution	32
2	Convolution	32
3	Convolution	64
4	Convolution	64
5	Fully Connected	-

the low-dimensional space is defined as

$$r_{ij} = \frac{(1 + \frac{q-1}{3-q}\|f'(\boldsymbol{x}_i|\boldsymbol{W}') - f'(\boldsymbol{x}_j|\boldsymbol{W}')\|^2)^{-\frac{1}{q-1}}}{\sum_l^N \sum_{k\neq l}^N (1 + \frac{q-1}{3-q}\|f'(\boldsymbol{x}_k|\boldsymbol{W}') - f'(\boldsymbol{x}_l|\boldsymbol{W}')\|^2)^{-\frac{1}{q-1}}}, \tag{4.1}$$

where r_{ii} is 0, and $r_{ij} = r_{ji}$ for $\forall i, j$. The Kullback-Leibler divergence is defined as Eq.(2.18) and is used to train the parameters of the CNN.

The gradient for $\boldsymbol{y}_i$ is given as

$$\frac{\partial C}{\partial \boldsymbol{W}'} = \frac{\partial C}{\partial f'(\boldsymbol{x}_i|\boldsymbol{W}')} \frac{\partial f'(\boldsymbol{x}_i|\boldsymbol{W}')}{\partial \boldsymbol{W}'}, \tag{4.2}$$

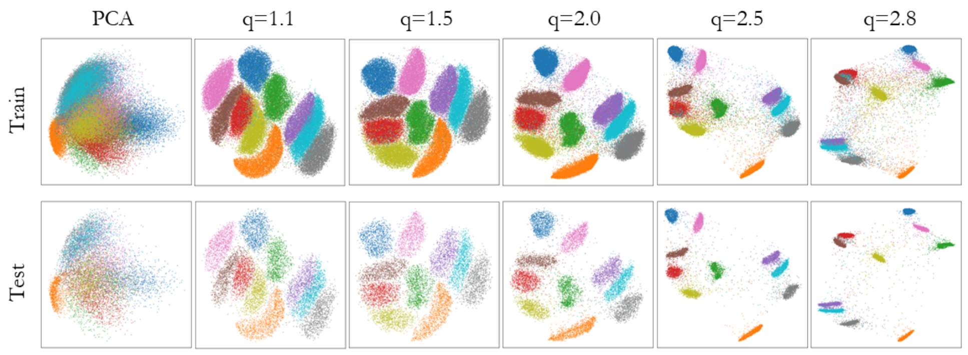
where

$$\begin{aligned} \frac{\partial C}{\partial f'(\boldsymbol{x}_i|\boldsymbol{W}')} = \frac{4}{3-q} \sum_j^N (p_{ij} - r_{ij})(f'(\boldsymbol{x}_i|\boldsymbol{W}') - f'(\boldsymbol{x}_j|\boldsymbol{W}')) \\ (1 + \frac{q-1}{3-q}\|f'(\boldsymbol{x}_i|\boldsymbol{W}') - f'(\boldsymbol{x}_j|\boldsymbol{W}')\|^2)^{-1}, \end{aligned} \tag{4.3}$$

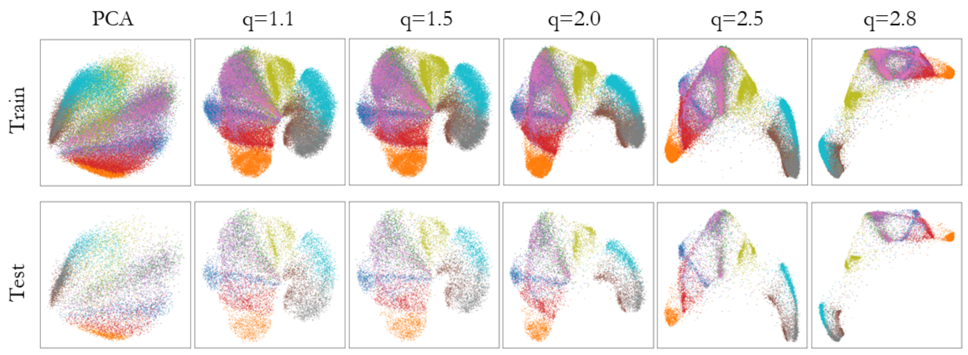
and $\frac{\partial f'(\boldsymbol{x}_i|\boldsymbol{W}')}{\partial \boldsymbol{W}'}$ can be computed by using the standard automatic differentiation.

The architecture of parametric q-SNE with CNN is shown in Fig.4.1. The pairs $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are used in the training. This is very similar with Siamese neural network [8]. To consider the similarity between some samples, such construction is very important.

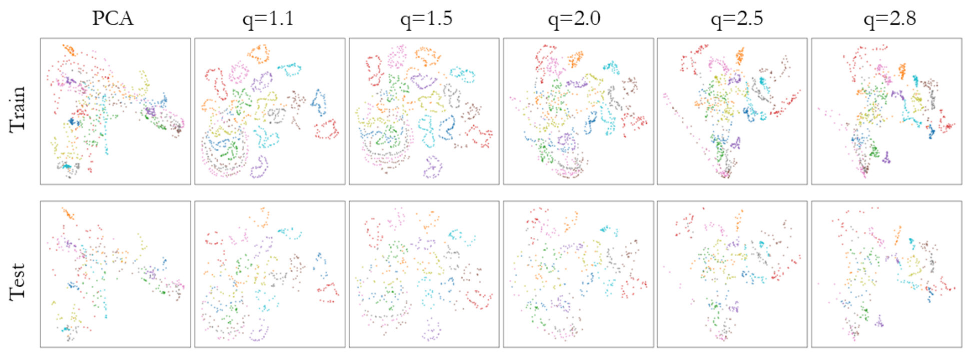
Since the q-Gaussian distribution can express t-distribution when $q = 2.0$, the proposed parametric q-SNE can generate the same low-dimensional embedded space with the parametric t-SNE. By choosing hyperparameter q, the parametric q-SNE gives better results than parametric t-SNE.



(a) MNIST



(b) FashionMNIST



(c) COIL-20

FIGURE 4.2: This figure shows the 2-d mapping of (a) MNIST, (b) FashionMNIST, and (c) COIL-20 dataset by using PCA, parametric t-SNE, and parametric q-SNE. When $q = 2.0$ of parametric q-SNE, the parametric q-SNE becomes the same as parametric t-SNE. The perplexity for each 2-mapping is used when the classification accuracy of the Table4.2 is the best.

TABLE 4.2: This table shows classification accuracy by using parametric q-SNE on MNIST, FashionMNIST, and COIL-20 dataset. The Perp denotes the perplexity. When $q = 2.0$ of parametric q-SNE, then parametric q-SNE becomes the same as parametric t-SNE.

			Parametric q-SNE							
			q							
		Perp	1.1	1.3	1.5	1.8	2.0 (t-SNE)	2.3	2.5	2.8
MNIST	Train	5	0.9455	0.9544	0.9609	0.9621	**0.9668**	0.9652	0.9650	0.9626
		10	0.9331	0.9353	0.9375	0.9519	0.9549	0.9608	0.9645	0.9654
		20	0.9039	0.9364	0.9504	0.9560	0.9590	0.9577	0.9631	0.9651
		30	0.9154	0.9226	0.9323	0.9508	0.9586	0.9604	0.9610	0.9589
		40	0.8911	0.9153	0.9210	0.9477	0.9489	0.9508	0.9611	0.9630
		50	0.8834	0.9211	0.9203	0.9542	0.9561	0.9567	0.9558	0.9625
	Test	5	0.9276	0.9386	0.9492	0.9527	0.9599	0.9594	0.9590	0.9572
		10	0.9070	0.9143	0.9165	0.9389	0.9463	0.9549	0.9592	**0.9606**
		20	0.8712	0.9211	0.9387	0.9466	0.9508	0.9488	0.9565	0.9592
		30	0.8898	0.9030	0.9156	0.9390	0.9500	0.9527	0.9543	0.9512
		40	0.8561	0.8916	0.9002	0.9351	0.9368	0.9407	0.9531	0.9566
		50	0.8462	0.9022	0.8971	0.9438	0.9471	0.9483	0.9471	0.9547
FashionMNIST	Train	5	0.7589	0.7630	0.7627	0.7636	0.7609	0.7613	0.7668	0.7742
		10	0.7676	0.7695	0.7738	0.7757	0.7749	0.7707	0.7707	0.7746
		20	0.7585	0.7628	0.7680	0.7673	0.7677	0.7699	0.7740	0.7749
		30	0.7573	0.7600	0.7629	0.7647	0.7669	0.7664	0.7686	0.7731
		40	0.7557	0.7614	0.7641	0.7708	0.7718	0.7740	0.7740	0.7758
		50	0.7545	0.7639	0.7681	0.7705	0.7755	**0.7765**	0.7725	**0.7765**
	Test	5	0.6652	0.6726	0.6735	0.6692	0.6682	0.6716	0.6762	0.6865
		10	0.6780	0.6785	0.6876	0.6879	0.6875	0.6816	0.6810	0.6865
		20	0.6666	0.6714	0.6753	0.6739	0.6805	0.6775	0.6833	0.6891
		30	0.6646	0.6681	0.6688	0.6749	0.6780	0.6760	0.6788	0.6842
		40	0.6603	0.6695	0.6727	0.6800	0.6860	0.6884	0.6834	0.6863
		50	0.6619	0.6718	0.6783	0.6831	0.6880	0.6892	0.6839	**0.6905**
COIL-20	Train	5	0.9290	0.9306	0.9350	0.9310	0.9162	0.8872	0.8726	0.8684
		10	0.9384	0.9340	**0.9424**	0.9356	0.9092	0.8892	0.8942	0.8816
		20	0.9108	0.9196	0.9196	0.9162	0.9066	0.8848	0.8772	0.8788
		30	0.8920	0.9014	0.9100	0.9058	0.9012	0.8884	0.8870	0.8820
		40	0.8622	0.8922	0.9016	0.8936	0.8990	0.8948	0.8958	0.8862
		50	0.8710	0.8916	0.9008	0.9078	0.8964	0.8900	0.8866	0.8860
	Test	5	0.8327	0.8395	0.8527	0.8605	0.8400	0.8155	0.7927	0.7873
		10	0.8591	0.8491	0.8582	**0.8650**	0.8355	0.8200	0.8250	0.8136
		20	0.8359	0.8373	0.8350	0.8350	0.8277	0.8036	0.8050	0.8109
		30	0.8041	0.8214	0.8168	0.8268	0.8200	0.8036	0.8136	0.8023
		40	0.8000	0.8182	0.8214	0.8277	0.8241	0.8155	0.8250	0.8100
		50	0.8091	0.8214	0.8232	0.8323	0.8177	0.8223	0.8150	0.8209

4.2 Experiments

4.2.1 Mapping Experiments

In this section, we will show the 2-d visualization of image datasets MNIST, Fashion-MNIST, and COIL-20 by using principal component analysis (PCA), the parametric t-SNE, and parametric q-SNE with several hyperparameter q. The MNIST dataset has 60,000 training images and 10,000 test images. Each image is a grayscale hand-written digit. The size of the image is 28×28. The FashionMNIST dataset has 60,000 training images and 10,000 test images. Each image is a grayscale fashion item. The size of the image is 28×28. The COIL-20 dataset has 1,440 images of 20 objects. Each image is that the objects were placed on a motorized turntable against a black background. The turntable was rotated through 360 degrees to vary object pose concerning a fixed camera. Images of the objects were taken at pose intervals of 5 degrees. The size of each image is 128×128. For experiments, we divided randomly 1,000 training images and 440 test images.

In experiments, we use CNN in Table 4.1. Each convolutional layer has a 7×7 kernel with no padding. We use the ReLU function as an activation function. The optimizer is stochastic gradient descent (SGD)[21] with momentum. A learning rate is 0.05 or 0.01 for MNIST or FashionMNIST and COIL-20 respectively. Weight decay is 0.0001 or 0.001 for MNIST and FashionMNIST or COIL-20 respectively. A momentum is 0.8 An epoch is 500 or 3000 for MNIST and FashionMNIST or COIL-20 respectively. The learning rate is multiplied by 0.1 every 100 or 500 epochs for MNSIT and FashionMNIST or COIL-20. The dimension of the output of CNN is 2. For COIL-20 dataset, we applied resize from 128×128 to 32×32 as a pre-processing. Before training, we use fixed 5,000 data points for bachs to prepare the conditional probability in the high-dimensional space.

We show the 2-d mapping in Fig.4.2 by using principal component analysis (PCA), parametric t-SNE, and parametric q-SNE. When $q = 2.0$, the 2-d mapping is the same as parametric t-SNE. The perplexity for each 2-d mapping is used when the classification accuracy of the Table4.2 is the best. According to Fig.4.2, the larger q is, the more samples are clustered. The parametric q-SNE can express different embedding by choosing hyperparameter q.

4.2.2 Comparison Experiments

In this section, we compared classification accuracy by using the parametric t-SNE and parametric q-SNE. The settings of experiments for the training phase are the same as visualization experiments. For classification accuracy, we use k nearest neighbor (k-NN). we fitted k-NN on embeddings of the training sample, and we got test sample classification accuracy by it. In this experiment, we set k to 5. The classification accuracy is averaged 5 trials with different seeds. In Table 4.2, We show the classification accuracy of training or test sample by using parametric t-SNE and parametric q-SNE on MNIST, FashionMNIST, and COIL-20. When $q = 2.0$ of parametric q-SNE, the parametric q-SNE becomes the same as parametric t-SNE. The parametric q-SNE gives better classification accuracy for each dataset than the parametric t-SNE.

Chapter 5

Conclusion

5.1 q-SNE and q-UMAP

In section 3, we described our proposal. We proposed to use q-Gaussian distribution for dimensionality reduction. These are called q-Gaussian distributed stochastic neighbor embedding (q-SNE) and q-Gaussian distributed uniform manifold approximation and projection (q-UMAP). The q-SNE uses q-Gaussian distribution in low-dimensional space instead of t-distribution of t-SNE. The q-UMAP uses q-Gaussian distribution in low-dimensional space instead of curve of UMAP. Through the experiments, our proposal can control intuitively embedding space by choosing hyperparameter q. The UMAP can also control embedding space, but our proposal can control more intuitively. In classification experiments, our proposal achieved higher classification accuracy score than UMAP and t-SNE on MNIST, COIL-20, Olivetti-Faces, and FashionMNIST dataset. The q-SNE and t-SNE need long computational time than UMAP, but q-UMAP can run fast as same as UMAP. Since the q-SNE includes t-SNE, we can use our proposal instead of t-SNE. Since the q-UMAP can control intuitively embedding space than UMAP and give faster computational time as same as UMAP, we can use our proposal instead of UMAP. However, they can not map new coming sample to low-dimensional space, because they don't make mapping function.

5.2 parametric q-SNE

In section 4, we descrived our second proposal. We also proposed parametric q-SNE with convolutional neural network(CNN) to construct the nonlinear mapping. The CNN is trained to minimize q-SNE loss between outputs of CNN. Through the experiments, we can control the embedding and map the new coming sample to same embedding space. In classification experiments, the parametric q-SNE achieves higher classification accuracy score than parametric t-SNE on MNIST, COIL-20, and FashionMNIST dataset.

5.3 Future Work

Our proposal extended the previous works by using q-Gaussian distribution. We confirmed effectiveness from experiments. In our experiments, we evaluated the accuracy scores by using teacher labels, however we can not know the goodness of mapping for each dataset without teacher labels. In such situations, we have to evaluate by seeing the mapping ourselves. The important future work is to find the best hyperparameter q of the q-Gaussian distribution, and to define the way to score the embedding.

Appendix A

Derivation

A.1 Gradient of SNE

In this section, we show the derivation of SNE gradient Eq.(2.15). The SNE minimizes the Kullback-Leibler divergence between conditional probability introduced by Gaussian distribution.

We show the formulation again.

$$p_{j|i} = \frac{\exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma_i^2\right)}{\sum_{k\neq i}^{N} \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_k\|^2/2\sigma_i^2\right)}, \tag{A.1}$$

$$r_{j|i} = \frac{\exp\left(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)}{\sum_{k\neq i}^{N} \exp\left(-\|\boldsymbol{y}_i - \boldsymbol{y}_k\|^2\right)}, \tag{A.2}$$

$$\begin{aligned} C &= \sum_{i}^{N}\sum_{j\neq i}^{N} p_{j|i} \log \frac{p_{j|i}}{r_{j|i}} \\ &= \sum_{i}^{N}\sum_{j\neq i}^{N} p_{j|i} \log p_{j|i} - p_{j|i} \log r_{j|i}. \end{aligned} \tag{A.3}$$

And to consider more simple, we set

$$d_{ij} = \|\boldsymbol{y}_i - \boldsymbol{y}_j\|, \tag{A.4}$$

$$S = \sum_{k\neq i}^{N} \exp\left(-d_{ik}^2\right). \tag{A.5}$$

Then the gradient of the cost function C for $\boldsymbol{y}_i$ is defined as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = \sum_{j}^{N} \left(\frac{\partial C}{\partial d_{ij}} + \frac{\partial C}{\partial d_{ji}}\right)(\boldsymbol{y}_i - \boldsymbol{y}_j). \tag{A.6}$$

the gradient $\frac{\partial C}{\partial d_{ij}}$ is defined as

$$\begin{aligned}\frac{\partial C}{\partial d_{ij}} &= -\sum_u^N \sum_{v\neq u}^N p_{u|v} \frac{\partial (\log r_{u|v})}{\partial d_{ij}} \\ &= -\sum_u^N \sum_{v\neq u}^N p_{u|v} \frac{\partial (\log r_{u|v}S - \log S)}{\partial d_{ij}} \\ &= -\sum_u^N \sum_{v\neq u}^N p_{u|v} \left(\frac{1}{r_{u|v}S} \frac{\partial r_{u|v}S}{\partial d_{ij}} - \frac{1}{S}\frac{\partial S}{\partial d_{ij}} \right) \\ &= -\sum_u^N \sum_{v\neq u}^N \frac{p_{u|v}}{r_{u|v}S} \frac{\partial \exp{(-d_{uv}^2)}}{\partial d_{ij}} + \sum_u^N \sum_{v\neq u}^N \frac{p_{u|v}}{S}\frac{\partial S}{\partial d_{ij}}. \end{aligned} \tag{A.7}$$

The gradient $\sum_u^N \sum_{v\neq u}^N \frac{p_{u|v}}{r_{u|v}S} \frac{\partial \exp{(-d_{uv}^2)}}{\partial d_{ij}}$ is only non zero when $u = i$ and $v = j$. Hence the gradient of $\frac{\partial C}{\partial d_{ij}}$ becomes

$$\frac{\partial C}{\partial d_{ij}} = 2\frac{p_{i|j}}{r_{i|j}S} \exp{(-d_{ij}^2)} - 2\sum_u^N \sum_{v\neq u}^N \frac{p_{u|v}}{S} \exp{(-d_{ij}^2)}. \tag{A.8}$$

Noting that $\sum_u^N \sum_{v\neq u}^N p_{u|v} = 1$ and $r_{i|j} = \frac{\exp{(-d_{ij}^2)}}{S}$, the gradient simplifies to

$$\frac{\partial C}{\partial d_{ij}} = 2p_{i|j} - 2r_{i|j}. \tag{A.9}$$

Finally, we can get the gradient as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = 2\sum_j^N (p_{j|i} - r_{j|i} + p_{i|j} - r_{i|j})(\boldsymbol{y}_i - \boldsymbol{y}_j). \tag{A.10}$$

A.2 Gradient of symmetric SNE

In this section, we show the derivation of symmetric SNE gradient Eq.(2.19). The symmetric SNE minimizes the Kullback-Leibler divergence between joint probability introduced by Gaussian distribution.

We show the formulation again.

$$p_{ij} = \frac{1}{2}(p_i p_{j|i} + p_j p_{i|j}) = \frac{p_{j|i} + p_{i|j}}{2N}, \tag{A.11}$$

$$r_{ij} = \frac{\exp{(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)}}{\sum_l^N \sum_{k\neq l}^N \exp{(-\|\boldsymbol{y}_l - \boldsymbol{y}_k\|^2)}}, \tag{A.12}$$

$$\begin{aligned} C &= \sum_i^N \sum_{j\neq i}^N p_{ij} \log \frac{p_{ij}}{r_{ij}} \\ &= \sum_i^N \sum_{j\neq i}^N p_{ij} \log p_{ij} - p_{ij} \log r_{ij}. \end{aligned} \tag{A.13}$$

And to consider more simple, we set

$$S = \sum_{l}^{N} \sum_{k \neq l}^{N} \exp(-d_{lk}^2). \tag{A.14}$$

Then the gradient of the cost function C for $\boldsymbol{y}_i$ is defined as

$$\begin{aligned} \frac{\partial C}{\partial \boldsymbol{y}_i} &= \sum_{j}^{N} \left(\frac{\partial C}{\partial d_{ij}} + \frac{\partial C}{\partial d_{ji}} \right) (\boldsymbol{y}_i - \boldsymbol{y}_j) \\ &= 2 \sum_{j}^{N} \frac{\partial C}{\partial d_{ij}} (\boldsymbol{y}_i - \boldsymbol{y}_j). \end{aligned} \tag{A.15}$$

the gradient $\frac{\partial C}{\partial d_{ij}}$ is defined as

$$\begin{aligned} \frac{\partial C}{\partial d_{ij}} &= - \sum_{u}^{N} \sum_{v \neq u}^{N} p_{uv} \frac{\partial (\log r_{uv})}{\partial d_{ij}} \\ &= - \sum_{u}^{N} \sum_{v \neq u}^{N} p_{uv} \frac{\partial (\log r_{uv} S - \log S)}{\partial d_{ij}} \\ &= - \sum_{u}^{N} \sum_{v \neq u}^{N} p_{uv} \left(\frac{1}{r_{uv} S} \frac{\partial r_{uv} S}{\partial d_{ij}} - \frac{1}{S} \frac{\partial S}{\partial d_{ij}} \right) \\ &= - \sum_{u}^{N} \sum_{v \neq u}^{N} \frac{p_{uv}}{r_{uv} S} \frac{\partial \exp(-d_{uv}^2)}{\partial d_{ij}} + \sum_{u}^{N} \sum_{v \neq u}^{N} \frac{p_{uv}}{S} \frac{\partial S}{\partial d_{ij}}. \end{aligned} \tag{A.16}$$

The gradient $\sum_{u}^{N} \sum_{v \neq u}^{N} \frac{p_{uv}}{r_{uv} S} \frac{\partial \exp(-d_{uv}^2)}{\partial d_{ij}}$ is only non zero when $u = i$ and $v = j$. Hence the gradient of $\frac{\partial C}{\partial d_{ij}}$ becomes

$$\frac{\partial C}{\partial d_{ij}} = 2 \frac{p_{ij}}{r_{ij} S} \exp(-d_{ij}^2) - 2 \sum_{u}^{N} \sum_{v \neq u}^{N} \frac{p_{uv}}{S} \exp(-d_{ij}^2). \tag{A.17}$$

Noting that $\sum_{u}^{N} \sum_{v \neq u}^{N} p_{uv} = 1$ and $r_{ij} = \frac{\exp(-d_{ij}^2)}{S}$, the gradient simplifies to

$$\frac{\partial C}{\partial d_{ij}} = 2 p_{ij} - 2 r_{ij}. \tag{A.18}$$

Finally, we can get the gradient as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = 4 \sum_{j}^{N} (p_{ij} - r_{ij})(\boldsymbol{y}_i - \boldsymbol{y}_j). \tag{A.19}$$

A.3 Gradient of t-SNE

In this section, we show the derivation of t-SNE gradient Eq.(2.21). The t-SNE minimizes the Kullback-Leibler divergence between joint probability introduced by

Gaussian distribution and t-distribution.

We show the formulation again.

$$r_{ij} = \frac{(1+\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-1}}{\sum_l^N \sum_{k\neq l}^N (1+\|\boldsymbol{y}_k - \boldsymbol{y}_l\|^2)^{-1}}, \tag{A.20}$$

$$\begin{aligned} C &= \sum_i^N \sum_{j\neq i}^N p_{ij} \log \frac{p_{ij}}{r_{ij}} \\ &= \sum_i^N \sum_{j\neq i}^N p_{ij} \log p_{ij} - p_{ij} \log r_{ij}. \end{aligned} \tag{A.21}$$

And to consider more simple, we set

$$S = \sum_l^N \sum_{k\neq l}^N (1+d_{lk}^2)^{-1}. \tag{A.22}$$

Then the gradient of the cost function C for $\boldsymbol{y}_i$ is defined as

$$\begin{aligned} \frac{\partial C}{\partial \boldsymbol{y}_i} &= \sum_j^N \left(\frac{\partial C}{\partial d_{ij}} + \frac{\partial C}{\partial d_{ji}} \right) (\boldsymbol{y}_i - \boldsymbol{y}_j) \\ &= 2 \sum_j^N \frac{\partial C}{\partial d_{ij}} (\boldsymbol{y}_i - \boldsymbol{y}_j). \end{aligned} \tag{A.23}$$

the gradient $\frac{\partial C}{\partial d_{ij}}$ is defined as

$$\begin{aligned} \frac{\partial C}{\partial d_{ij}} &= -\sum_u^N \sum_{v\neq u}^N p_{uv} \frac{\partial (\log r_{uv})}{\partial d_{ij}} \\ &= -\sum_u^N \sum_{v\neq u}^N p_{uv} \frac{\partial (\log r_{uv} S - \log S)}{\partial d_{ij}} \\ &= -\sum_u^N \sum_{v\neq u}^N p_{uv} \left(\frac{1}{r_{uv} S} \frac{\partial r_{uv} S}{\partial d_{ij}} - \frac{1}{S} \frac{\partial S}{\partial d_{ij}} \right) \\ &= -\sum_u^N \sum_{v\neq u}^N \frac{p_{uv}}{r_{uv} S} \frac{\partial (1+d_{uv}^2)^{-1}}{\partial d_{ij}} + \sum_u^N \sum_{v\neq u}^N \frac{p_{uv}}{S} \frac{\partial S}{\partial d_{ij}}. \end{aligned} \tag{A.24}$$

The gradient $\sum_u^N \sum_{v\neq u}^N \frac{p_{uv}}{r_{uv} S} \frac{\partial (1+d_{uv}^2)^{-1}}{\partial d_{ij}}$ is only non zero when $u = i$ and $v = j$. Hence the gradient of $\frac{\partial C}{\partial d_{ij}}$ becomes

$$\begin{aligned} \frac{\partial C}{\partial d_{ij}} &= 2 \frac{p_{ij}}{r_{ij} S} (1+d_{ij}^2)^{-2} - 2 \sum_u^N \sum_{v\neq u}^N \frac{p_{uv}}{S} (1+d_{ij}^2)^{-2} \\ &= 2(p_{ij} - r_{ij})(1+d_{ij}^2)^{-1}. \end{aligned} \tag{A.25}$$

Noting that $\sum_u^N \sum_{v\neq u}^N p_{uv} = 1$ and $r_{ij} = \frac{(1+d_{ij}^2)^{-1}}{S}$, the gradient simplifies to

$$\frac{\partial C}{\partial d_{ij}} = 2(p_{ij} - r_{ij})(1 + d_{ij}^2)^{-1}. \tag{A.26}$$

Finally, we can get the gradient as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = 4\sum_j^N (p_{ij} - r_{ij})(\boldsymbol{y}_i - \boldsymbol{y}_j)(1 + \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-1}. \tag{A.27}$$

A.4 Gradient of UMAP

In this section, we show the derivation of UMAP gradient Eq.(2.28). The UMAP minimizes the Cross Entropy Loss between joint probability introduced by Gaussian distribution and curve which is similar to the probability distribution.

We show the formulation again.

$$p_{j|i} = \exp\left(-\frac{max(0, \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 - \rho_i)}{\sigma_i}\right), \tag{A.28}$$

$$p_{ij} = p_{j|i} + p_{i|j} - pj|ipi|j, \tag{A.29}$$

$$r_{ij} = (1 + a\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b})^{-1}, \tag{A.30}$$

$$CE = -\sum_i^N \sum_{j\neq i}^N p_{ij} \log r_{ij} + (1 - p_{ij}) \log (1 - r_{ij}). \tag{A.31}$$

Then the gradient of the cost function C for $\boldsymbol{y}_i$ is defined as

$$\frac{\partial CE}{\partial \boldsymbol{y}_i} = \sum_j^N \frac{\partial CE}{\partial d_{ij}}(\boldsymbol{y}_i - \boldsymbol{y}_j) \tag{A.32}$$

the gradient $\frac{\partial CE}{\partial d_{ij}}$ is defined as

$$\begin{aligned}\frac{\partial CE}{\partial d_{ij}} &= -\sum_u^N \sum_{v\neq u}^N p_{uv}\frac{\partial(\log r_{uv})}{\partial d_{ij}} + (1 - p_{uv})\frac{\partial(\log(1 - r_{uv}))}{\partial d_{ij}} \\ &= -\sum_u^N \sum_{v\neq u}^N \frac{p_{uv}}{r_{uv}}\frac{\partial r_{uv}}{\partial d_{ij}} - \frac{1 - p_{uv}}{1 - r_{un}}\frac{\partial r_{uv}}{\partial d_{ij}}.\end{aligned} \tag{A.33}$$

The gradient is only non zero when $u = i$ and $v = j$. Hence the gradient of $\frac{\partial CE}{\partial d_{ij}}$ becomes

$$\begin{aligned}
\frac{\partial CE}{\partial d_{ij}} &= -\frac{p_{ij}}{r_{ij}}\frac{\partial r_{ij}}{\partial d_{ij}} + \frac{1-p_{ij}}{1-r_{ij}}\frac{\partial r_{ij}}{\partial d_{ij}} \\
&= -\left(\frac{p_{ij}}{r_{ij}} - \frac{1-p_{ij}}{1-r_{ij}}\right)\frac{\partial r_{ij}}{\partial d_{ij}} \\
&= -\left(\frac{p_{ij}}{r_{ij}} - \frac{1-p_{ij}}{1-r_{ij}}\right)\frac{\partial (1+ad_{ij}^{2b})^{-1}}{\partial d_{ij}} \\
&= \left(\frac{p_{ij}}{r_{ij}} - \frac{1-p_{ij}}{1-r_{ij}}\right)\frac{2abd_{ij}^{2b-1}}{(1+ad_{ij}^{2b})^2}.
\end{aligned} \tag{A.34}$$

Finally, we can get the gradient as

$$\frac{\partial CE}{\partial \boldsymbol{y}_i} = \sum_j^N \left(\frac{p_{ij}}{r_{ij}} + \frac{1-p_{ij}}{1-r_{ij}}\right)(\boldsymbol{y}_i - \boldsymbol{y}_j)\frac{2ab\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b-1}}{(1+a\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^{2b})^2} \tag{A.35}$$

A.5 Gradient of q-SNE

In this section, we show the derivation of q-SNE gradient Eq.(3.2). The q-SNE minimizes the Kullback-Leibler divergence between joint probability introduced by Gaussian distribution and q-Gaussian distribution.

We show the formulation again.

$$r_{ij} = \frac{(1+\frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2)^{-\frac{1}{q-1}}}{\sum_l^N \sum_{k\neq l}^N (1+\frac{q-1}{3-q}\|\boldsymbol{y}_l - \boldsymbol{y}_k\|^2)^{-\frac{1}{q-1}}}, \tag{A.36}$$

$$\begin{aligned}
C &= \sum_u^N \sum_{v\neq u}^N p_{uv}\log\frac{p_{uv}}{r_{uv}} \\
&= \sum_u^N \sum_{v\neq u}^N p_{uv}\log p_{uv} - p_{uv}\log r_{uv},
\end{aligned} \tag{A.37}$$

And to consider more simple, we set

$$S = \sum_l^N \sum_{k\neq l}^N (1+d_{lk}^2)^{-\frac{1}{q-1}}. \tag{A.38}$$

Then the gradient of the cost fucntion C for $\boldsymbol{y}_i$ is defined as

$$\begin{aligned}
\frac{\partial C}{\partial \boldsymbol{y}_i} &= \sum_j^N \left(\frac{\partial C}{\partial d_{ij}} + \frac{\partial C}{\partial d_{ji}}\right)(\boldsymbol{y}_i - \boldsymbol{y}_j) \\
&= 2\sum_j^N \frac{\partial C}{\partial d_{ij}}(\boldsymbol{y}_i - \boldsymbol{y}_j).
\end{aligned} \tag{A.39}$$

the gradient $\frac{\partial C}{\partial d_{ij}}$ is defined as

$$\begin{aligned}
\frac{\partial C}{\partial d_{ij}} &= -\sum_{u}^{N}\sum_{v\neq u}^{N} p_{uv}\frac{\partial(\log r_{uv})}{\partial d_{ij}} \\
&= -\sum_{u}^{N}\sum_{v\neq u}^{N} p_{uv}\frac{\partial(\log r_{uv}S - \log S)}{\partial d_{ij}} \\
&= -\sum_{u}^{N}\sum_{v\neq u}^{N} p_{uv}\left(\frac{1}{r_{uv}S}\frac{\partial r_{uv}S}{\partial d_{ij}} - \frac{1}{S}\frac{\partial S}{\partial d_{ij}}\right) \\
&= -\sum_{u}^{N}\sum_{v\neq u}^{N} \frac{p_{uv}}{r_{uv}S}\frac{\partial(1+\frac{q-1}{3-q}d_{uv}^{2})^{-\frac{1}{q-1}}}{\partial d_{ij}} + \sum_{u}^{N}\sum_{v\neq u}^{N}\frac{p_{uv}}{S}\frac{\partial S}{\partial d_{ij}}. \qquad \text{(A.40)}
\end{aligned}$$

The gradient $\frac{p_{uv}}{r_{uv}S}\frac{\partial(1+\frac{q-1}{3-q}d_{uv}^{2})^{-\frac{1}{q-1}}}{\partial d_{ij}}$ is only non zero when $u = i$ and $v = j$.Hence the gradient of $\frac{\partial C}{\partial d_{ij}}$ becomes

$$\begin{aligned}
\frac{\partial C}{\partial d_{ij}} &= -\frac{p_{ij}}{r_{ij}S}\frac{-1}{q-1}\frac{2(q-1)}{3-q}(1+\frac{q-1}{3-q}d_{ij}^{2})^{-\frac{1}{q-1}-1} \\
&\quad + \sum_{u}^{N}\sum_{v\neq u}^{N}\frac{p_{uv}}{S}\frac{-1}{q-1}\frac{2(q-1)}{3-q}(1+\frac{q-1}{3-q}d_{ij}^{2})^{-\frac{1}{q-1}-1}. \qquad \text{(A.41)}
\end{aligned}$$

Noting that $\sum_{u}^{N}\sum_{v\neq u}^{N} p_{uv} = 1$ and $r_{ij} = \frac{(1+\frac{q-1}{3-q}d_{ij}^{2})^{-\frac{1}{q-1}})}{S}$, the gradient simplifies to

$$\begin{aligned}
\frac{\partial C}{\partial d_{ij}} &= \frac{p_{ij}}{r_{ij}}\frac{2}{3-q}r_{ij}(1+\frac{q-1}{3-q}d_{ij}^{2})^{-1} - \frac{2}{3-q}r_{ij}(1+\frac{q-1}{3-q}d_{ij}^{2})^{-1} \\
&= \frac{2}{3-q}(p_{ij}-r_{ij})(1+\frac{q-1}{3-q}d_{ij}^{2})^{-1}. \qquad \text{(A.42)}
\end{aligned}$$

Finally, we can get the gradient as

$$\frac{\partial C}{\partial \boldsymbol{y}_i} = \frac{4}{3-q}\sum_{j}^{N}(p_{ij}-r_{ij})(\boldsymbol{y}_i-\boldsymbol{y}_j)(1+\frac{q-1}{3-q}\|\boldsymbol{y}_i-\boldsymbol{y}_j\|^{2})^{-1}. \qquad \text{(A.43)}$$

A.6 Gradient of q-UMAP

In this section, we show the derivation of q-UMAP gradient Eq.(3.4). The q-UMAP minimizes the Cross Entropy Loss between joint probability introduced by Gaussian distribution and q-Gaussian distribution.

We show the formulation again.

$$p_{j|i} = \exp\left(-\frac{max(0, \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 - \rho_i)}{\sigma_i}\right), \tag{A.44}$$

$$p_{ij} = p_{j|i} + p_{i|j} - pj|ipi|j, \tag{A.45}$$

$$r_{ij} = \left(1 + \frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)^{-\frac{1}{q-1}}, \tag{A.46}$$

$$CE = -\sum_i^N \sum_{j \neq i}^N p_{ij} \log r_{ij} + (1 - p_{ij}) \log(1 - r_{ij}). \tag{A.47}$$

Then the gradient of the cost function C for $\boldsymbol{y}_i$ is defined as

$$\frac{\partial CE}{\partial \boldsymbol{y}_i} = \sum_j^N \frac{\partial CE}{\partial d_{ij}}(\boldsymbol{y}_i - \boldsymbol{y}_j) \tag{A.48}$$

the gradient $\frac{\partial CE}{\partial d_{ij}}$ is defined as

$$\begin{aligned}\frac{\partial CE}{\partial d_{ij}} &= -\sum_u^N \sum_{v \neq u}^N p_{uv}\frac{\partial(\log r_{uv})}{\partial d_{ij}} + (1 - p_{uv})\frac{\partial(\log(1 - r_{uv}))}{\partial d_{ij}} \\ &= -\sum_u^N \sum_{v \neq u}^N \frac{p_{uv}}{r_{uv}}\frac{\partial r_{uv}}{\partial d_{ij}} - \frac{1 - p_{uv}}{1 - r_{un}}\frac{\partial r_{uv}}{\partial d_{ij}}.\end{aligned} \tag{A.49}$$

The gradient is only non zero when $u = i$ and $v = j$. Hence the gradient of $\frac{\partial CE}{\partial d_{ij}}$ becomes

$$\begin{aligned}\frac{\partial CE}{\partial d_{ij}} &= -\frac{p_{ij}}{r_{ij}}\frac{\partial r_{ij}}{\partial d_{ij}} + \frac{1 - p_{ij}}{1 - r_{ij}}\frac{\partial r_{ij}}{\partial d_{ij}} \\ &= -\left(\frac{p_{ij}}{r_{ij}} - \frac{1 - p_{ij}}{1 - r_{ij}}\right)\frac{\partial r_{ij}}{\partial d_{ij}} \\ &= -\left(\frac{p_{ij}}{r_{ij}} - \frac{1 - p_{ij}}{1 - r_{ij}}\right)\frac{\partial\left(1 + \frac{q-1}{3-q}d_{ij}^2\right)^{-\frac{1}{q-1}}}{\partial d_{ij}} \\ &= \left(\frac{p_{ij}}{r_{ij}} - \frac{1 - p_{ij}}{1 - r_{ij}}\right)\frac{2}{3-q}\left(1 + \frac{q-1}{3-q}d_{ij}^2\right)^{-\frac{1}{q-1}-1}.\end{aligned} \tag{A.50}$$

Finally, we can get the gradient as

$$\frac{\partial CE}{\partial \boldsymbol{y}_i} = \frac{2}{3-q}\sum_j^N \left(\frac{p_{ij}}{r_{ij}} + \frac{1 - p_{ij}}{1 - r_{ij}}\right)(\boldsymbol{y}_i - \boldsymbol{y}_j)\left(1 + \frac{q-1}{3-q}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2\right)^{-\frac{1}{q-1}-1}. \tag{A.51}$$

Bibliography

[1] Motoshi Abe, Junichi Miyao, and Takio Kurita. "Adaptive Neuron-wise Discriminant Criterion and Adaptive Center Loss at Hidden Layer for Deep Convolutional Neural Network". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–8.

[2] Motoshi Abe, Junichi Miyao, and Takio Kurita. "q-SNE: Visualizing Data using q-Gaussian Distributed Stochastic Neighbor Embedding". In: *arXiv preprint arXiv:2012.00999* (2020).

[3] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network". In: *2017 International Conference on Engineering and Technology (ICET)*. Ieee. 2017, pp. 1–6.

[4] Moez Baccouche et al. "Sequential deep learning for human action recognition". In: *International workshop on human behavior understanding*. Springer. 2011, pp. 29–39.

[5] G. Bebis and M. Georgiopoulos. "Feed-forward neural networks". In: *IEEE Potentials* 13.4 (1994), pp. 27–31. DOI: 10.1109/45.329294.

[6] Mikhail Belkin and Partha Niyogi. "Laplacian eigenmaps and spectral techniques for embedding and clustering". In: *Advances in neural information processing systems*. 2002, pp. 585–591.

[7] David M Chan et al. "t-SNE-CUDA: GPU-Accelerated t-SNE and its Applications to Modern Data". In: *2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. IEEE. 2018, pp. 330–338.

[8] Davide Chicco. "Siamese neural networks: An overview". In: *Artificial Neural Networks* (2020), pp. 73–94.

[9] Kenneth Ward Church. "Word2Vec". In: *Natural Language Engineering* 23.1 (2017), pp. 155–162.

[10] Michael AA Cox and Trevor F Cox. "Multidimensional scaling". In: *Handbook of data visualization*. Springer, 2008, pp. 315–347.

[11] John P Cunningham and Zoubin Ghahramani. "Linear dimensionality reduction: Survey, insights, and generalizations". In: *The Journal of Machine Learning Research* 16.1 (2015), pp. 2859–2900.

[12] Pierre Demartines and Jeanny Hérault. "Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets". In: *IEEE Transactions on neural networks* 8.1 (1997), pp. 148–154.

[13] Li Deng. "The mnist database of handwritten digit images for machine learning research [best of the web]". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[14] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[15] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[16] Kaiming He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[17] Geoffrey E Hinton. "A practical guide to training restricted Boltzmann machines". In: *Neural networks: Tricks of the trade*. Springer, 2012, pp. 599–619.

[18] Geoffrey E Hinton and Sam T Roweis. "Stochastic neighbor embedding". In: *Advances in neural information processing systems*. 2003, pp. 857–864.

[19] Elad Hoffer and Nir Ailon. "Deep metric learning using triplet network". In: *International workshop on similarity-based pattern recognition*. Springer. 2015, pp. 84–92.

[20] James M. Joyce. "Kullback-Leibler Divergence". In: *International Encyclopedia of Statistical Science*. Ed. by Miodrag Lovric. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 720–722. ISBN: 978-3-642-04898-2. DOI: 10.1007/978-3-642-04898-2_327. URL: https://doi.org/10.1007/978-3-642-04898-2_327.

[21] Nikhil Ketkar. "Stochastic gradient descent". In: *Deep learning with Python*. Springer, 2017, pp. 113–132.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012), pp. 1097–1105.

[23] Wentian Li et al. "Application of t-SNE to human genetic data". In: *Journal of bioinformatics and computational biology* 15.04 (2017), p. 1750017.

[24] Nelson Liu. *scikit-learn olivetti faces dataset olivettifaces.mat*. 2016. DOI: 10.6084/m9.figshare.3829989.v2. URL: https://figshare.com/articles/dataset/olivettifaces_mat/3829989/2.

[25] Wei Liu et al. "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[26] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.

[27] Popescu Marius et al. “Multilayer perceptron and neural networks”. In: *WSEAS Transactions on Circuits and Systems* 8 (July 2009).

[28] Leland McInnes, John Healy, and James Melville. “Umap: Uniform manifold approximation and projection for dimension reduction”. In: *arXiv preprint arXiv:1802.03426* (2018).

[29] Sebastian Mika et al. “Fisher discriminant analysis with kernels”. In: *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*. Ieee. 1999, pp. 41–48.

[30] Vinod Nair and Geoffrey E Hinton. “Rectified linear units improve restricted boltzmann machines”. In: *Icml*. 2010.

[31] Karl Pearson. “LIII. On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[32] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[33] Jeffrey Pennington, Richard Socher, and Christopher D Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[34] Leif E Peterson. “K-nearest neighbor”. In: *Scholarpedia* 4.2 (2009), p. 1883.

[35] C Rate and C Retrieval. “Columbia object image library (coil-20)”. In: *Computer* (2011).

[36] Paulo E Rauber, Alexandre X Falcao, Alexandru C Telea, et al. “Visualizing Time-Dependent Data Using Dynamic t-SNE.” In: (2016).

[37] Joseph Redmon and Ali Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[38] Sam T Roweis and Lawrence K Saul. “Nonlinear dimensionality reduction by locally linear embedding”. In: *science* 290.5500 (2000), pp. 2323–2326.

[39] John W Sammon. “A nonlinear mapping for data structure analysis”. In: *IEEE Transactions on computers* 100.5 (1969), pp. 401–409.

[40] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. “Kernel principal component analysis”. In: *International conference on artificial neural networks*. Springer. 1997, pp. 583–588.

[41] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[42] Masaru Tanaka. *Geometry of Entropy*. Series on Stochastic Models in Informatics and Data Science. CORONA PUBLISHING CO.,LTD., 2019.

[43] Jian Tang et al. “Visualizing large-scale and high-dimensional data”. In: *Proceedings of the 25th international conference on world wide web*. 2016, pp. 287–297.

[44] Joshua B Tenenbaum, Vin De Silva, and John C Langford. "A global geometric framework for nonlinear dimensionality reduction". In: *science* 290.5500 (2000), pp. 2319–2323.

[45] Bruce Thompson. "Canonical correlation analysis". In: *Encyclopedia of statistics in behavioral science* (2005).

[46] Laurens Van Der Maaten. "Accelerating t-SNE using tree-based algorithms". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 3221–3245.

[47] Laurens Van Der Maaten. "Barnes-hut-sne". In: *arXiv preprint arXiv:1301.3342* (2013).

[48] Laurens Van Der Maaten. "Learning a parametric embedding by preserving local structure". In: *Artificial Intelligence and Statistics*. 2009, pp. 384–391.

[49] Kilian Q Weinberger, Fei Sha, and Lawrence K Saul. "Learning a kernel matrix for nonlinear dimensionality reduction". In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 106.

[50] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747* (2017).

[51] Bolei Zhou et al. "Learning deep features for scene recognition using places database". In: (2014).