CLUSTERING BASED ON MULTIPLE MODELS

Tianyi Wang[1]
University of Science and Technology of China
Hefei, 230026
wtyyy@mail.ustc.edu.cn

Wenyi Jin[1]
University of Science and Technology of China
Hefei, 230026
jinwenyi@mail.ustc.edu.cn

Guangyu Li[1]
University of Science and Technology of China
Hefei, 230026
flipped@mail.ustc.edu.cn

January 26, 2024

ABSTRACT

We employ multiple classic clustering models to the dataset MNIST and visualize their clustering results.To further optimize the clustering performance performance regarding our dataset, we combine two different manifold learning methods with GMM model for the final clustering algorithm. The idea of the above algorithm(N2D) was put forward by Ryan McConville et al.(2020).

***K*eywords** Clustering · Machine Learning · Dimension Reduction

1 Introduction

Clustering is a fundamental technique used in data analysis to group similar objects together based on their inherent characteristics. In unsupervised learning, the labeled information of the training samples is unknown, and the purpose of clustering learning is to learn from the unlabeled samples to reveal the intrinsic nature and regularity of the data, and provide a basis for further data analysis. It is based on similarity, i.e. in a cluster There is more similarity between patterns than between patterns that are not in the same cluster.

Clustering attempts to divide the samples in the data set into several usually non-intersecting subsets, each subset is called a "cluster". It should be noted that these concepts are unknown in advance for the clustering algorithm, and the clustering process automatically form the cluster structure, and the conceptual semantics corresponding to the cluster need to be grasped and named by the user.

Clustering can be used as a separate process to find the internal distribution structure of data, and can also be used as a precursor to other learning tasks such as classification. For example, in some commercial applications, it is necessary to identify the type of new users, but defining "user type" may not be easy for merchants. At this time, it is often possible to cluster user data first, define each cluster as a class according to the cluster results, and then practice classification models based on these classes used to identify the type of new users.

Based on learning strategies, people have designed many types of clustering algorithms, like K-means Clustering, Fuzzy C-means Clustering etc. In this paper, we focus on K-Means Clustering,Mixture of Gaussian(GMM), and Density Peaks Clustering(DPC), applying them to the MINIST dataset. To achieve data visualization, we also introduce some basic dimensionality reduction algorithms for our datasets.

It's also notable that in this paper, we make a few improvements to the classic GMM model based on the algorithm proposed by Ryan McConville et al.(2020), combining the model with two different manifold learning methods to optimize the clustering performance.

This article is organized as follows: Section 2 explains three clustering models investigated in our study including their algorithms. Section 3 introduces two approaches to reduce dimentions. Section 4 gives out a brief introduction of the data set and reveals the experimental results and discussion. Section 5 presents the optimized clustering algorithm and the testing result of it based on the same dataset. Section 6 briefly summarized the experimental results of our work. Section 7 is dedicated to our critical reflections throughout the process of conducting this report. The rest part is references.

2 Clustering Models

2.1 K-Means

The k-means algorithm is a well-known **prototype clustering algorithm**, which minimizes the squared error for the resulting cluster division $C = \{C_1, C_2, ..., C_k\}$. Intuitively, Equation 1 depicts the closeness of the samples within the cluster around the mean vector, and the smaller the E value, the higher the similarity of the samples within the cluster.

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||_2^2 \tag{1}$$

The greedy strategy is used to minimize 1, and the solution method is approximated through iterative optimization. The algorithm[1] process is as follows:

Algorithm 1 K-Means

Input: Sample set D = $\{\boldsymbol{x_1}, ..., \boldsymbol{x_m}\}$;
Number k.
Output:
Initialize k mean-vectors$\{\mu_1, ..., \mu_k\}$selected from D randomly.
repeat
  set $C_i = \emptyset (1 \leq i \leq k)$
  for j = 1,2,...,m **do**
    Compute distance: $d_{ij} = ||x_j - \mu_i||_2 \quad (1 \leq i \leq k)$;
    Ensure the label: $\lambda_j = \mathbf{argmin}_{i \in \{1,...,k\}} d_{ji}$;
    Assign$\{\boldsymbol{x_j}\}$ to the appropriate clusters: $C_{\lambda_j} = C_{\lambda_j} \bigcup \{\boldsymbol{x_j}\}$;
  end for
  for i = 1,...,k **do**
    Compute the new mean-vector: $\boldsymbol{\mu}_i' = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$;
    if $\boldsymbol{\mu}_i' \neq \boldsymbol{\mu}_i$ **then**
      $\boldsymbol{\mu}_i \leftarrow \boldsymbol{\mu}_i'$
    else
      remain $\boldsymbol{\mu}_i$
    end if
  end for
until NO RENEW
return $\mathcal{C} = \{C_1, ..., C_k\}$

2.2 Mixture of Gaussian(GMM)

Unlike k-means, which uses prototype vectors to describe the clustering structure, Gaussian mixed clustering uses a probabilistic model to express clustering prototypes.We can define a Gaussian mixed distribution

$$p_{\mathcal{M}}(\boldsymbol{x}) = \sum_{i=1}^{k} \alpha_i \cdot p(\boldsymbol{x} | \boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}) \tag{2}$$

The distribution consists of k mixed components, each of which corresponds to a normal distribution.μ_i, Σ_i are the parameters of the ith mixed component, and $\alpha_i > 0$ is called mixture coefficient, $\sum_{i=1}^{k} \alpha_i = 1$.

For 2, the model parameters are estimated with maximum likelihood, i.e., maximized (logarithmic) likelihood, and the EM algorithm is often used to solve the problem iteratively. We omit the detailed derivation proof of the algorithm, which is described as follows:

$$\gamma_{ji} = p_{\mathcal{M}}(z_j = i|\boldsymbol{x}) = \frac{\alpha_i \cdot p(\boldsymbol{x_j}|\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i})}{\sum_{l=1}^{k} \alpha_l \cdot p(\boldsymbol{x_j}|\boldsymbol{\mu_l}, \boldsymbol{\Sigma_l})} \tag{3}$$

Algorithm 2 GMM

Input: Sample set D = $\{\boldsymbol{x_1}, ..., \boldsymbol{x_m}\}$;
Number k.
Output:
Initialize parameters$\{(\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)|1 \leq i \leq k\}$
repeat
  for j = 1,2,...,m **do**
    Compute posterior probability by 3:$\gamma_{ji} = p_{\mathcal{M}}(z_j = i|\boldsymbol{x})$;
  end for
  for i = 1,2,...,k **do**
    Compute new mean vector: $\boldsymbol{\mu}'_i = \frac{\sum_{j=1}^{m} \gamma_{ji} x_j}{\sum_{j=1}^{m} \gamma_{ji}}$;
    Compute new covariance matrix: $\boldsymbol{\Sigma}'_i = \frac{\sum_{j=1}^{m} \gamma_{ji} (\boldsymbol{x_j} - \boldsymbol{\mu'_i})(\boldsymbol{x_j} - \boldsymbol{\mu'_i})^{\boldsymbol{T}}}{\sum_{j=1}^{m} \gamma_{ji}}$;
    Compute new mixture coefficient:$\alpha'_j = \frac{\sum_{j=1}^{m} \gamma_{ji}}{m}$;
  end for
  Renew parameters:
  $\boldsymbol{\mu}_i \leftarrow \boldsymbol{\mu}'_i, \boldsymbol{\Sigma}_i \leftarrow \boldsymbol{\Sigma}'_i, \alpha_j \leftarrow \alpha'_i$
until The stop condition is met
$C_i = \emptyset (1 \leq i \leq k)$
for j = 1,...,m **do**
  Ensure the label: $\lambda_j = \mathbf{argmax}_{i \in \{1,2,...,k\}} \gamma_{ji}$;
  Assign$\{\boldsymbol{x_j}\}$ to the appropriate clusters: $C_{\lambda_j} = C_{\lambda_j} \bigcup \{\boldsymbol{x_j}\}$
end for
return $\mathcal{C} = \{C_1, ..., C_k\}$

In each iteration, the posterior probabilityγ_{ji} that each sample belongs to each Gaussian component is calculated based on the current parameters(E Step), and the model parameters are updated based on the maximum likelihood estimate(M Step).

2.3 Density Peaks Clustering[8]

Similar to the K-medoids method, it has its basis only in the distance between data points. Like DBSCAN and the mean-shift method, it is able to detect nonspherical clusters and to automatically find the correct number of clusters. The cluster centers are defined, as in the mean-shift method, as local maxima in the density of data points. However, unlike the mean-shift method, our procedure does not require embedding the data in a vector space and maximizing explicitly the density field for each data point.

The algorithm has its basis in the assumptions that cluster centers are surrounded by neighbors with lower local density and that they are at a relatively large distance from any points with a higher local density. For each data point i, we compute two quantities: its local density ρ_i and its distance δ_i from points of higher density. Both these quantities depend only on the distances d_{ij} between data points, which are assumed to satisfy the triangular inequality. The local density ρ_i of data point i is defined as

$$\rho_i = \sum_j \chi(d_{ij} - d_c) \tag{4}$$

where $\chi(x) = 1 \text{if} x < 0$ and $\chi(x) = 0 \text{otherwise}$, and d_c is a cutoff distance. The algorithm is sensitive only to the relative magnitude of ρ_i in different points, implying that, for large data sets, the results of the analysis are robust with respect to the choice of d_c.

δ_i is measured by computing the minimum distance between the point i and any other point with higher density:

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}) \tag{5}$$

For the point with highest density, we take $\delta_i = \max_j(d_{ij})$. Note that δ_i is much larger than the typical nearest neighbor distance only for points that are local or global maxima in the density. Thus, **cluster centers** are recognized as points for which the value of δ_i is anomalously large.

This observation, which is the core of the algorithm.We will look for them through the **Decision Plot** and, of course, if the Decision Plot is not obvious, the decision values $\gamma_i = \rho_i \times \delta_i$, can also be considered.

Algorithm 3 DPC

Input: Sample set D = $\{\boldsymbol{x_1}, ..., \boldsymbol{x_m}\}$;
Output:
Ensure the cutoff distance : d_c;
for i,j = 1,2,...,m **do**
  Compute distance: $d_{ij} = d(\boldsymbol{x_i}, \boldsymbol{x_j})$;
end for
for i = 1,2,...,m **do**
  Compute local density by 4: $\rho_i = \sum_j \chi(d_{ij} - d_c)$;
  Compute relative distance by 5: $\delta_i = \min_{j:\rho_j > \rho_i}(d_{ij})$;
  Or $\delta_i = \max_j(d_{ij})$
end for
Draw Decision Plot;
Set threshold of ρ, δ;
Pick the cluster centers with high ρ_i and δ_i;
for j = 1,...,m **do**
  Ensure the label: $\lambda_j = \mathbf{argmin}_{i \in \{1,2,...,k\}} d_{ji}$;
  Assign$\{\boldsymbol{x_j}\}$ to the appropriate clusters: $C_{\lambda_j} = C_{\lambda_j} \bigcup \{\boldsymbol{x_j}\}$
end for
return $\mathcal{C} = \{C_1, ..., C_k\}$

2.4 Other Clustering Methods

In fact, this is just the tip of the iceberg of clustering methods, in addition to that, there are density-based DBSCAN, hierarchical clustering, spectral clustering, and so on. Due to space constraints, we can't introduce them one by one, but you can refer to them for details.

3 Dimensionality Reduction Methods

Problems such as sparse data samples and difficult distance calculation in high-dimensional situations are serious obstacles faced by all machine learning methods, which are called "dimensionality disasters". [2]An important way to alleviate the dimensionality disaster is dimensionality reduction, that is, to transform the original high-dimensional attribute space into a low-dimensional "subspace" through some mathematical transformation, in which the sample density is greatly increased, and the distance calculation also becomes easier.

Since the Minst dataset we chose has 784 dimensions, we applied dimensionality reduction techniques to our dataset for calculation and visualization easily.

3.1 Principal component analysis(PCA)

Principal component analysis is one of the most commonly used methods for dimensionality reduction. The main idea of PCA is to map n-dimensional features to k(< n) dimension, which is a new orthogonal feature, also known as principal components, which is a reconstructed k-dimensional feature on the basis of the original n-dimensional feature. We expect it to have the following characteristics:

Re-proximal reconstruction: The sample points are close enough to the hyperplane.

Maximum separability: The specimen points are projected in this hyperplane as far apart as possible.

PCA only needs to retain the mean vector of W and the sample mean, and the new sample can be projected into the low-dimensional space by simple vector subtraction and matrix multiplication. We discard some of the information, but it is often necessary for the sake of dimensionality reduction and noise reduction;

Algorithm 4 PCA

Input: Sample set D = $\{\boldsymbol{x_1}, ..., \boldsymbol{x_m}\}$;
The number of low-dimensional spatial dimensions: d'
Output:
1: **for** j = 1,2,...,m **do**
2: Centralization: $\boldsymbol{x_i} = \boldsymbol{x_i} - \frac{1}{m}\sum_{i=1} m\boldsymbol{x_i}$;
3: **end for**
4: Compute Covariance Matrix$\boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}}$;
5: Eigenvalue decomposition (or singular value decomposition);
6: Take the eigenvector corresponding to the largest k eigenvalues$\boldsymbol{w}_1, ..., \boldsymbol{w}_k$
7: **return** Projection matrix$\mathcal{W} = \{w_1, ..., w_k\}$;Sample Mean.

3.2 Mainfold Learning

Manifold learning is a class of dimensionality reduction methods that draw on the concept of topological manifolds. Manifold learning assumes that the data is uniformly sampled in a high-dimensional Euclidean space in a low-dimensional manifold structure, and it aims to recover the low-dimensional manifold structure from the high-dimensional sampled data, that is, to find the low-dimensional manifold in the high-dimensional space, and find the corresponding embedding map to achieve dimension reduction or data visualization.

Manifold learning techniques, such as t-SNE, enable effective visualization of high-dimensional data, revealing underlying patterns and structures. [5] They preserve local relationships between nearby data points, accurately representing the intrinsic structure of the data. Also these methods are robust to noise and outliers, focusing on capturing underlying structure rather than irrelevant features.

We expect it to have the following characteristics:

Local preservation: The local relationships between nearby data points are preserved in the low-dimensional embedding.

Non-linear mapping: Manifold learning techniques aim to capture and represent non-linear structures in the data.

Dimensionality reduction: Manifold learning methods reduce the dimensionality of the data while preserving its essential structure.

Topology preservation: The topological structure of the data, such as clusters and boundaries, is preserved in the low-dimensional embedding.

Intrinsic geometry: Manifold learning techniques aim to capture the intrinsic geometry of the data, revealing its underlying organization.

Non-Euclidean distances: Manifold learning methods often utilize non-Euclidean distance measures to capture the complex relationships between data points.

Robustness to noise: Manifold learning algorithms are designed to be robust to noise and outliers in the data.

Visualization: Manifold learning techniques are often used for visualizing high-dimensional data in a lower-dimensional space.

Algorithm 5 Manifold Learning

Input: Sample set $D = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$; The number of low-dimensional spatial dimensions: d'
Output:
1: **for** $j = 1, 2, \ldots, m$ **do**
2: Preprocessing step for data point $\boldsymbol{x}_j$;
3: **end for**
4: Compute pairwise distances or affinity matrix $\boldsymbol{S}$ based on D;
5: Construct the graph or neighborhood structure based on $\boldsymbol{S}$;
6: Perform dimensionality reduction to obtain a low-dimensional embedding $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}$;
7: **return** Low-dimensional embedding $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}$;

4 Application

4.1 Data

4.1.1 Aggression

We use the simple dataset "Aggression" to test the obvious differences of clustering performances of three models.

The "Aggression" dataset used in this paper consists of a matrix of text samples and aggression labels. The dataset is organized into rows and columns, with each row representing a specific text sample and each column representing a different attribute or feature of that sample. It is a valuable resource in the field of natural language processing and social science research, providing insights into aggressive behavior in different contexts.

4.1.2 MNIST

The dataset we use in this paper, MNIST, is a matrix of 1,000 28x28 images, and the gray-scale is represented by numbers. It is a widely used benchmark dataset in the field of machine learning and computer vision. It stands for the Modified National Institute of Standards and Technology dataset, and it consists of a large collection of grayscale images of handwritten digits. The dataset has been carefully labeled, making it ideal for training and evaluating various image classification algorithms.

In this paper, we leverage the MNIST dataset to evaluate the performance of our proposed algorithm in the task of clustering.

4.2 Definitions of Measurement

Firstly, before the experiment part, we introduce all the evaluation metrics used in this paper.

- **Accuracy (ACC):** In clustering, accuracy (ACC) is defined as the best match between the ground truth and the predicted clusters.

$$ACC = \max_{m} \frac{\sum_{i=1}^{n} \mathbf{1}\{y_i = m(c_i)\}}{n} \tag{6}$$

  where y represents the ground truth labels, c represents the cluster labels, and m enumerates mappings between clusters and labels.

- **Normalized Mutual Information (NMI):** The Normalized Mutual Information (NMI) can be viewed as a normalization of the mutual information to scale the results between 0 and 1, where 0 indicates no mutual information and 1 indicates perfect correlation.

$$NMI = \frac{2I(y, c)}{H(y) + H(c)} \tag{7}$$

  where y represents the ground truth labels, c represents the cluster labels, H measures the entropy, and I is the mutual information between the ground truth labels and the cluster labels.

- **Davies-Bouldin Index (DBI):** DBI is a measure of clustering quality that takes into account the compactness and separation of the clustering results. A smaller DBI value indicates a better clustering effect.

$$DBI = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \tag{8}$$

  where k is the number of clusters, σ_i is the average distance of all samples in the ith cluster to the cluster center c_i, and $d(c_i, c_j)$ is the distance between c_i and c_j.

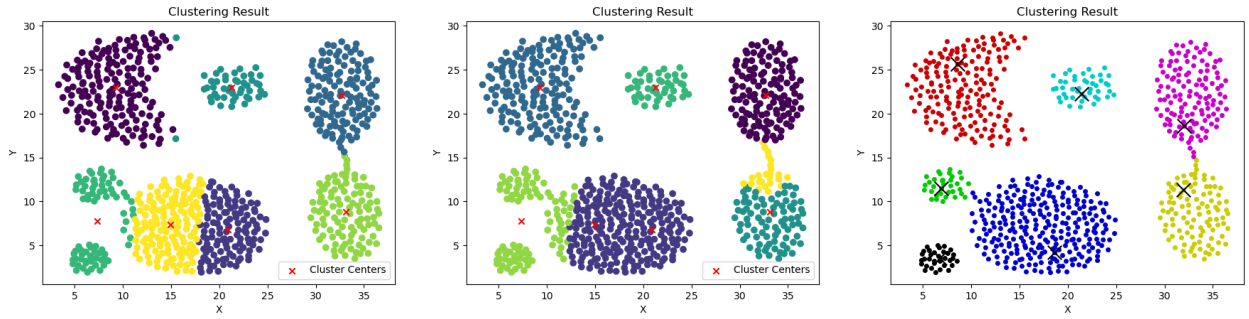
- **Adjusted Rand Index (ARI):** ARI is a metric used to measure the similarity of two data distributions, often used to assess the consistency between the clustering results and the true classification label.

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{9}$$

  where n is the total number of samples, n_{ij} represents the number of samples of category i in the ground truth labels and category j in the clustering result, a_i is the number of samples of category i in the ground truth labels, and b_j is the number of samples of category j in the clustering result.

4.3 Experiment and Discussion

Let's start by looking at the difference on a simple data set, Aggregation.



(A) Based on K-means(7) (B) Based on GMM (C) Based on DPC

FIGURE 1: Clustering Results On Aggregation

It can be found that for the first two, the key is the selection of the number of hyper-parameter clusters, but although the number of clusters is correct, all the sample points cannot be correctly classified. The outstanding advantage of this method is that we do not need to try different hyper-parameters multiple times, and d_c is more robust than K, which is also mentioned many times in the original paper[8].

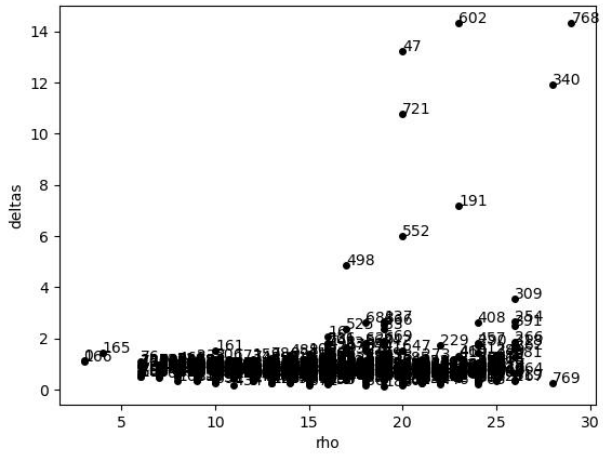


FIGURE 2: Decision Plot On Aggregation

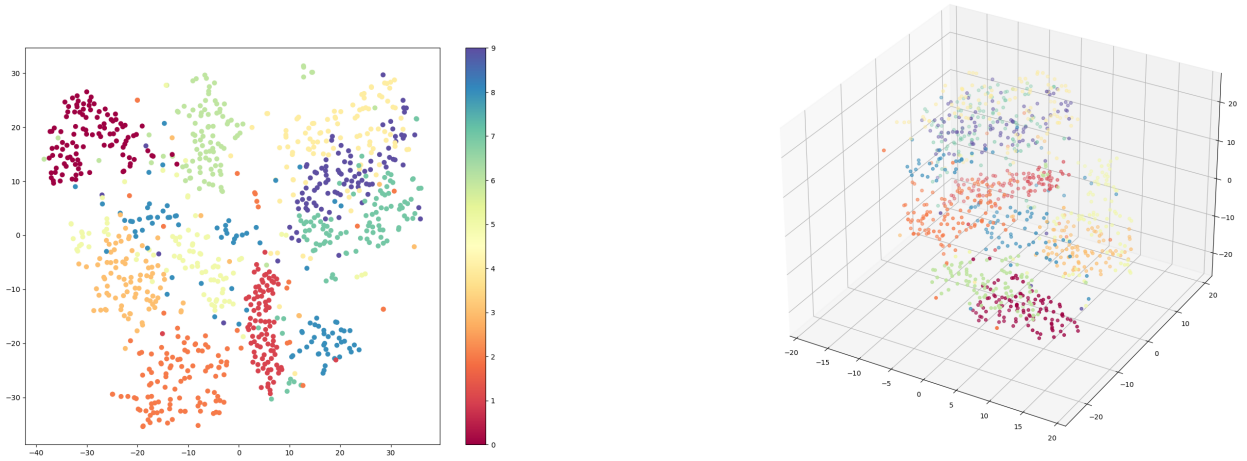
The clustering centers are determined by the decision graph above.

Table 1: Results of Aggression dataset in different method

	Davies-Bouldin Index	Adjusted Rand Index
K-means	0.7313	0.6
GMM	0.7058	0.5
DPC	0.5036	-

4.3.1 Clustering First

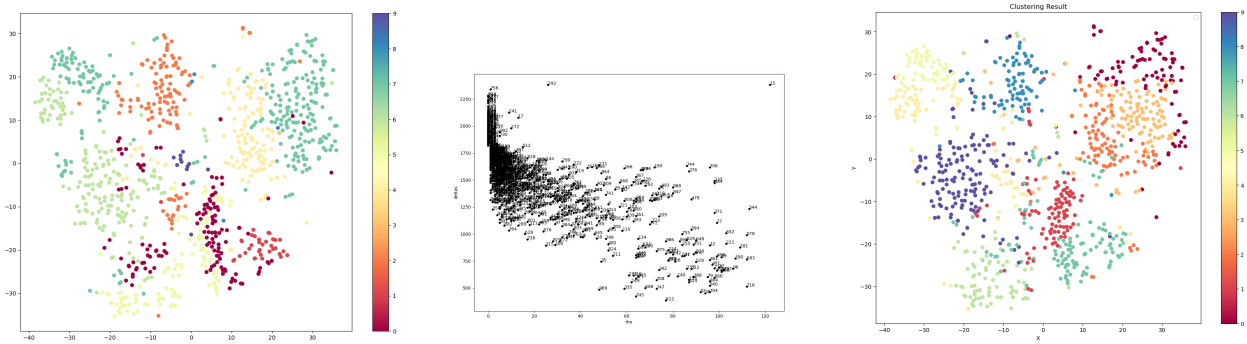
We visualize Minst dimensionality reduction(5) as follows:



(A) Minst dimensionality reduction(2D) (B) Minst dimensionality reduction(3D)

FIGURE 3: Minst dimensionality reduction

The corresponding clustering images are shown below. The strategy we adopt is to **cluster first** and then reduce dimensionality.



(A) Based on DPC(2D) (B) Decision Plot On Minst (C) Based on k

FIGURE 4: Clustering Results On Minst By T-SNE

Compared to the previous dataset, we found that there were a lot of data points, which has a high δ but lowρ, called **Outliers**(or halo).

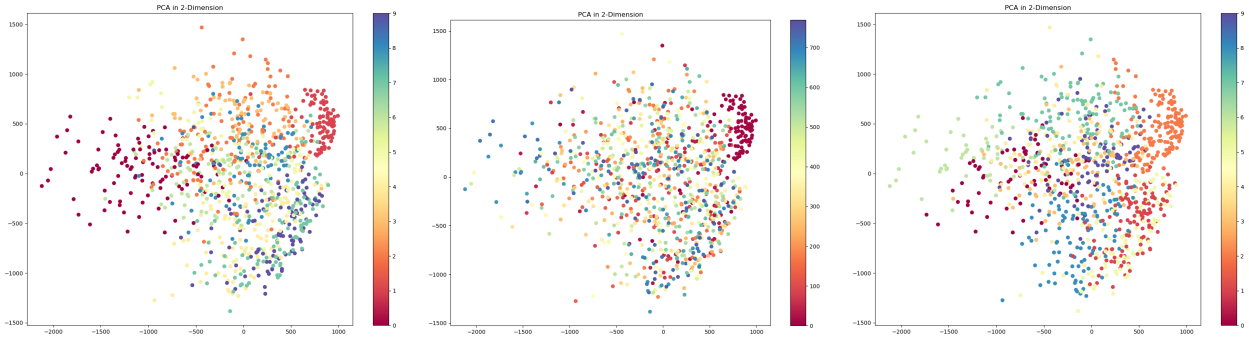
Similarly, PCA can also visualize clustering results.

Table 2: Results of MNIST dataset in different method

	Davies-Bouldin Index	Adjusted Rand Index
K-means	2.7803	0.3088
DPC	3.7451	0.2451

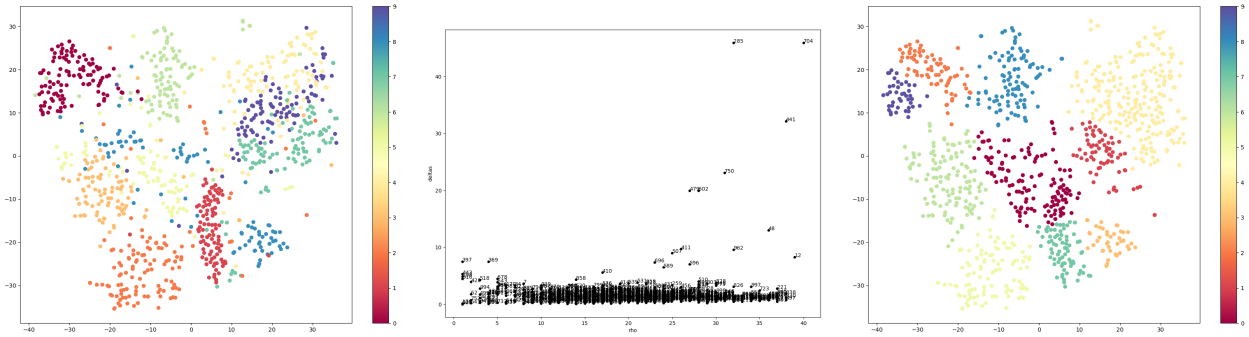
4.3.2 Dimensionality Reducion first

Now, let's take a different strategy: reduce dimensionality first and then cluster. The corresponding clustering images are shown below.



(A) Origin (B) Based on DPC(2D) (C) Based on k

FIGURE 5: Clustering Results On Minst By PCA



(A) Minst dimensionality reduction(2D) (B) Decision Plot On Minst (C) Based on DPC(2D)

FIGURE 6: Clustering Results On Minst By T-SNE

The difference from the previous section is that dimensionality reduction affects the accuracy of clustering. We'll compare the impact of dimensions later. Here, since there are too many images, we use a table description, and the generation of images is detailed in the appendix code.

Table 3: Results of MNIST dataset in different method

	Davies-Bouldin Index	Adjusted Rand Index
K-means(T-SNE, 2D)	0.7557	0.4832
K-means(T-SNE, 3D)	0.9924	0.4490
DPC(T-SNE, 2D)	0.7926	0.4459
DPC(T-SNE, 3D)	1.1181	0.4304
K-means(PCA, 2D)	0.8295	0.2175
K-means(PCA, 3D)	1.0122	0.2294
DPC(PCA, 2D)	1.0176	0.2231
DPC(PCA, 3D)	1.5835	0.2281

We can find that the first dimensionality reduction obviously shows better performance on the same method than the first clustering approach, which illustrates the necessity of dimensionality reduction. At the same time, with the difference of dimensionality, the clustering index will change accordingly, but whether it is linear is still debatable.

We can also see that compared with linear PCA, the nonlinear manifold learning method has outstanding performance in data processing, but at the same time, it cannot deny the convenience of PCA as a simple and feasible method.

Generally speaking, for clustering, we should combine the actual data set, introduce experts or feature selection if necessary, that is, **dimensionality reduction**, specific analysis of specific problems, and select appropriate reduction methods and clustering methods, so as to deal with the changeable problem in the data world.

5 Improvement

In the clustering algorithm, there are two main ways to improve, one is to change parameters and cluster number, the other is to improve the way of feature extraction(i.e. Dimensionality Reducion).

In this section, we introduce a method named N2D [6] which relies primarily on the combination of two different manifold learning methods and use a GMM for the final clustering algorithm. The first manifold learning method is an autoencoder, which while learning a representation, does not explicitly take local structure into account. We will show that by augmenting the autoencoder with a manifold learning technique which explicitly takes local structure into account, we can increase the quality of the representation learned in terms of clusterability. And the second is UMAP, which seeks to accurately represent local structure, but has been shown to also better incorporating global structure.

5.1 Autoencoder

An autoencoder is a deep neural network consisting of two key components.[3] The first is the encoder, which attempts to learn a function which maps the input x to a new feature vector (h = f(x)). The second component is the decoder, which attempts to learn a function which maps the learned feature space back to the original input space (r = g(h)). In other words, it is a neural network which attempts to copy its input to its output. This is typically achieved via a form of regularization, for example by forcing the network to compress the input into a lower dimensional space. The learning process can be described as minimizing the loss function L(x, g(f(x))), where L is a function which penalizes g(f(x)) for being dissimilar to x. One such loss may be the Mean Squared Error (MSE).

5.2 UMAP

UMAP[7] is a recently proposed manifold learning method, which seeks to accurately represent local structure, but has been shown to also better incorporating global structure. UMAP relies on three assumptions, namely that the data is uniformly distributed on a Riemannian manifold, that the Riemannian metric is locally constant and that the manifold is locally connected. From these assumptions it is possible to model the manifold with a fuzzy topological structure. The embedding is found by searching for a low dimensional projection of the data that has the closest possible equivalent fuzzy topological structure

5.3 Algorithm

We summarize the high level steps of our proposed method N2D as:

Algorithm 6 N2D

1: Apply an autoencoder to the raw data to learn an initial epresentation.
2: Re-embed the autoencoded embedding by UMAP.
3: Given this new, more clusterable embedding, apply GMM to discover the clusters.

More concisely, we may also simply represent N2D as

$$C = F_C(F_M(F_A(X))) \tag{10}$$

where C is the final clustering, F_C is the clustering algorithm, F_M is the manifold learner, F_A is the autoencoder and X is the original data.

5.4 Experiment

5.4.1 Datasets

- MNIST: A traditional benchmark dataset consisting of 70,000 handwritten digits belong to 10 different classes.
- MNIST-test: A subset of the MNIST dataset, containing only the test set of 10,000 images.

5.4.2 Evaluation Metrics

We will use four standard evaluation metrics mentioned in Sec.3.2 for validating the performance of unsupervised clustering algorithms. In both cases, values range between 0 and 1, where higher values correspond to better clustering performance.

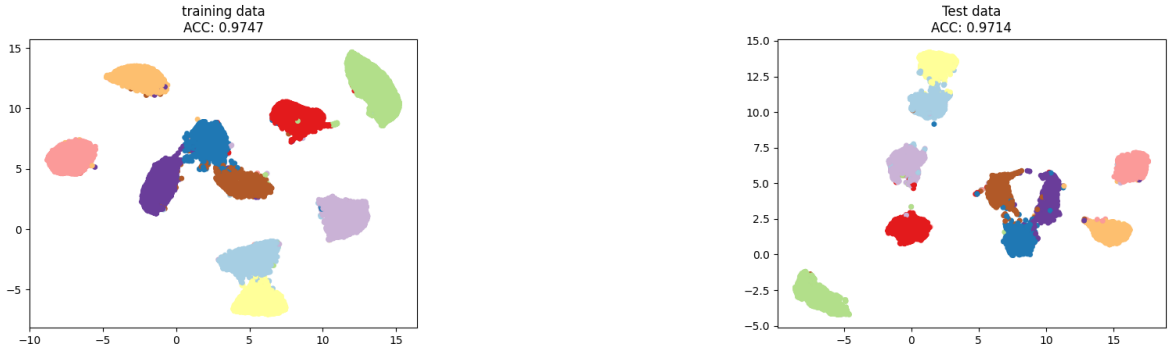
5.4.3 Experiment Settings

We base our autoencoder on the architecture described by Xie et al. [28], which is a fully connected Multi-Layer Perceptron (MLP). As typical with autoencoders, the decoder network is a mirror of the encoder. All layers use ReLU activation. The optimizer is Adam. We train the autoencoder on for 1000 epochs for all datasets. We use UMAP with the following default parameter set across all datasets. The number of neighbours is 20, the number of dimensions is the number of clusters, and the minimum distance between each point in the manifold is 0. We use a GMM for the final clustering algorithm, where each component has its own general covariance matrix, and there are c components, where c is the number of clusters.

5.4.4 Result

Table 4: Results of N2D

	Davies-Bouldin Index	Adjusted Rand Index	Accurancy	Normalized Mutual Information(NMI)
N2D	-	0.9376	0.9714	0.882



(A) Clustering in training set

(B) Clustering in test set

FIGURE 7: N2D Clutering Visualization

From the visualization results[4], we can see that the N2D model successfully distinguishes different handwritten digits by clustering. Compared to some of the classical models mentioned earlier, N2D also performs better on DBI and ARI values. Otherwise, the ACC and NMI values of N2D are also surprisingly high, which shows that N2D modeling is very successful. However, as we can tell from the visualization result, N2D does not perform as well on some numbers as DPC.

6 Conclusion

We can see from Tab.1 that K-means algorithm achieved a Davies-Bouldin Index(DBI) of 0.7313 and an Adjusted Rand Index(ARI) of 0.6, while the Gaussian Mixture Model (GMM) achieved a slightly lower DBI of 0.7058 and an ARI of 0.5. This suggests that the GMM method performed slightly worse than K-means in terms of clustering quality for some simple dataset like 'Aggression'. Meanwhile, The Density Peak Clustering (DPC) algorithm achieved the lowest DBI of 0.5036, we can conclude that DPC performs the best for simple datasets.

The results of Tab.2 indicate that K-means struggled to effectively cluster the MNIST dataset, as the high DBI suggests poor separation between the clusters and the low ARI indicates limited agreement with the ground truth labels. Meanwhile, DPC algorithm achieved a higher DBI of 3.7451 and a lower ARI of 0.2451. These results suggest that DPC also faced challenges in effectively clustering the MNIST dataset. We can say that we should put more efforts into optimizing classic clustering methods when it comes to high-dimensional data.

Based on the results of Tab.1 and Tab.2, it can be concluded that:

- The K-means algorithm generally performed better on the Aggression dataset compared to the MNIST dataset.

- The DPC algorithm showed promising results on the Aggression dataset with a low DBI, indicating good separation between clusters.
- However, both K-means and DPC struggled to effectively cluster the MNIST dataset, as shown by their high DBI and low ARI.

These findings highlight the importance of considering the characteristics and complexity of the dataset when selecting and evaluating clustering algorithms. Further investigations and experiments with alternative methods may be necessary to achieve better clustering results on the MNIST dataset, given that our models don't perform very well on it.

With Tab.2 and Tab.3, it shows that:

- Dimensionality reduction is necessary. When we use dimensionality reduction without changing other parameters, the ARI is increased from 0.2451 to 0.4459, almost double the former effect.
- Nonlinear manifold learning is outgoing in some ways. PCA is easier but inaccurate.
- The optimal dimension of dimensionality reduction is uncertain.

PCA is a linear mapping, but at the same time, it is simple and loses a lot of the original information of the data, so you should be cautious when using it. Of course, we should be cautious about dimensionality reduction techniques. As we can see, it's not that the dimensionality is about high, and the more information there is, the better the accuracy will be. Sometimes, it may be the opposite: low dimensions are more effective. As a result, many times, dimensionality reduction and clustering can only rely on traversal.

From Tab.4, we can see that:

- The ARI for the N2D model is 0.9376, indicating a relatively high level of agreement between the clustering results and the ground truth labels. This suggests that the N2D model performed well in terms of assigning data points to their correct clusters.
- The N2D model achieved an accuracy of 0.9714, indicating a high percentage of correctly classified data points.
- The Normalized Mutual Information (NMI) for the N2D model is 0.882, indicating a strong level of mutual information shared between the clustering results and the ground truth labels.

Comparing the N2D model with the former two models (K-means and DPC) regarding their clustering performances on the MNIST dataset, the N2D model demonstrates several advantages:

- The N2D model achieved the best clustering performance in terms of the DBI, indicating superior separation between clusters.
- The N2D model also outperformed the previous models in terms of the ARI and NMI, suggesting better agreement with ground truth labels and more accurate clustering assignments.
- These results indicate that the N2D model is capable of effectively capturing the inherent structure and patterns within the high-dimensional MNIST dataset.

Therefore, based on these findings, the N2D model shows promise as a powerful approach for clustering high-dimensional datasets like MNIST, offering improved performance compared to traditional clustering algorithms such as K-means and DPC. Further research and experimentation with the N2D model could potentially yield even better clustering results and advance our understanding.

7 Some Thinking

7.1 Dimensionality Reduction Before Clustering

When dealing with high-dimensional matrix datasets, we can consider using dimensionality reduction techniques such as PCA to preprocess the data before further analysis. In this paper, we apply clustering methods on the dataset before using PCA to reduce the dimensions, thus the code execution can be a bit slow by comparison, adding difficulty to data processing. We can work on that in the future. The followings are some benefits of utilizing dimensionality reduction techniques such as PCA as a preprocessing step before handling high-dimensional matrix datasets that we come up with, which might be helpful for future research:

- **Reduced computational complexity**:
  High-dimensional datasets often suffer from the curse of dimensionality, where the computational cost and resource requirements increase exponentially with the number of dimensions. By applying PCA or similar methods to reduce the dimensionality, we can significantly reduce the computational complexity of subsequent data processing tasks, making them more efficient and feasible.
- **Improved interpretability**:
  High-dimensional datasets can be difficult to interpret and visualize. By reducing the dimensionality using PCA, we can transform the data into a lower-dimensional space where patterns and relationships can be more easily understood and visualized. This can aid in gaining insights and understanding the underlying structure of the data.
- **Noise reduction and feature selection**:
  Dimensionality reduction techniques like PCA can help in filtering out noise and identifying the most important features or components in the data. By retaining only the most relevant dimensions, we can reduce the impact of irrelevant or noisy features, leading to improved data quality and more accurate analysis results.
- **Enhanced model performance**:
  High-dimensional datasets can pose challenges to machine learning models, such as overfitting and reduced generalization ability. By reducing the dimensionality, we can alleviate these issues and improve the performance of machine learning models. The reduced feature space can focus on capturing the most informative aspects of the data, leading to more robust and accurate models.

7.2 Neural Network Construction

- **Problems in Construction and Future Research** Actually, when we first started designing the final assignment, we plan to design a convolutional neural network as the dimensionality reduction method before the clustering algorithm in the Improvement part. However, the Conv+K-Mean model doesn't perform well in the clustering while each of these components excellently performs the task of classifying the MNIST dataset, which makes we really confused. This may be due to the unreasonable design of convolutional neural network and the failure of feature extraction. In this case, our future research interest is also determined, that is, clustering task based on neural network feature extraction.
- **Neural Network's Performance in MNIST Classification** As we mentioned before, we design some neural networks for the feature extraction but in vain. However, these neural network performs well in the MNIST classifying task, thus we decide to put these neural network's result in this part as an add-on. And some of the neural network models I use are as follows
  - Convolutional Neural Network(CNN)
  - Bidirectional Recurrent Neural Network(BRNN)

Table 5: Results of Neural Networks

	Convolutional Neural Network	Bidirectional Recurrent Neural Network
Classify Accurancy	98.96%	97.33 %

References

[1] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007.

[2] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems*, volume 15, pages 721–728, Cambridge, MA, USA, 2003. The MIT Press.

[3] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[4] D.A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, 2000.

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

[6] Ryan McConville, Raul Santos-Rodriguez, Robert J Piechocki, and Ian Craddock. N2d: (not too) deep clustering via clustering the local manifold of an autoencoded embedding, 2020.

[7] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[8] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014.