Manifold discovery for high-dimensional data using deep method

Jingjin Chen¹, Shuping Chen¹, Xuan Ding¹
¹Chongqing Aerospace Polytechnic, Chongqing, CO 40021, China
Corresponding author: Jingjin Chen (e-mail: jingjinchen2020@126.com).

ABSTRACT It is a challenge for manifold discovery from the data in the high-dimensional space, since the data in the high-dimensional space is sparsely distributed, which hardly provides rich information for manifold discovery so as to be possible to obtain deformed manifolds. To address this issue, this paper designed a deep model based on Brenier theorem for manifold discovery. Since Brenier theorem can find the optimal transportation mass distance between the reconstructed data distribution and the original data distribution, the manifold discovered from in the reconstructed data distribution can be as close to the original manifold as possible. Results not only show that the proposed method wins over competing methods both the precision of manifold discovery and resistance to data sparsity, but also show those non-linear architectures with deep paradigms outperform those architectures without deep paradigms in terms of manifold discovery. We find that the loss function derived by Brenier theorem can help that models minimize the error between the reconstructed manifold and the original manifold. Moreover, the manner constraining neurons with norm-2 is better than that of with norm-1 in both easing data sparsity and improving the precision of for manifold discovery.

INDEX TERMS Deep model, high dimensionality, manifold, sparsity.

I. INTRODUCTION
The distribution of the data in the high-dimensional space has a certain geometric shape. From a geometrical point of view, manifold is a geometric structure in the high-dimensional space, i.e., a collection of points in the space. In many cases, the high-dimensional data resides around a low-dimensional manifold. Manifold discovery assumes that the distribution of the data in the high-dimensional space is located on a lower-dimensional manifold, and data analysis is performed based on this assumption.

Real-world data is usually high dimensionality, while high dimensionality has a negative effect on the stability and learning ability of a model [1]. Since manifolds are an explicit form of data, manifolds exhibit a difference on datasets of different dimensions [2]. Data distribution in the high-dimensional space becomes more and more sparsity as data dimensionality increases, so that the data hardly provides rich information for manifold discovery. Moreover, data sparsity caused by high dimensionality may obtain deformed manifolds [3]. Sparsity is the intrinsic feature of the data, so it is tough task to disel sparsity effects when we process the relationships of between variables within a dataset. Consequently, it is a tough task for manifold discovery from the data in the high-dimensional space.

Manifold learning methods are often used as manifold discovery because of the capability of preserving local features to data, such as multi-dimensional scaling (MDS), Isometric Mapping (Isomap) and Locally Linear Embedding (LLE), etc. Where, MDS method constructs a low-dimensional space by using the paired similarity of samples, so that the distance between sample pairs in the constructed low-dimensional space is as consistent as possible with the distance between sample pairs in the corresponding high-dimensional space, e.g., the method in [4]. As for LLE method, a linear weight matrix is used to reconstruct local features, unfortunately, the manifold discovered by LLE method cannot be a closed form. Unlike LLE method, Isomap method maintains global structural information so that the geodesic distance relations between any pair of points existing in high-dimensional space remains unchanged in low-dimensional space, for example, the geometrical structure is preserved by Isomap method in the [5]. Since Isomap method can reveal the intrinsic geometric structure of manifold by preserving geodesic.
distance, it has outstanding ability for dimensionality reduction in high-dimensional data [6][7].

Certainly, deep architectures have been successful used for manifold discovery because they use many non-linearity layers to learn compact non-linear representations [8]. Deep architectures are not only able to learn deep features [9], but also interpret the learned features, for instance, the deep model [10] successfully discovers manifold from sparse high-dimensional data. Similarly, these examples in [11-13]. In addition, the stacked autoencoder (SAE) [14] was also used for deep manifold extraction, although the model extracts multi-manifold structure feature representations from complex data, it difficulty suitable for sparse data. Moreover, the complicated network structures of the model may increase the computational complexity during processing high-dimensional data. Similarly, these instances in [15-17], however, these models in [15-17] cannot address the sparsity of the data.

The motivation in this paper is to discover manifold from the data in the high-dimensional space, and visualize the relationships of between the high-dimensional data by using the captured manifolds. However, the final goals aims at figuring out some approaches of overcoming the curse of dimensionality, and providing some suggestions for resisting the data sparsity caused by high dimensionality. Hence, a deep model was proposed for achieving the studied targets. To ensure that the manifold discovered from the reconstructed data distribution can approach the original manifold as closely as possible, the loss function was derived from the view of the optimal transportation mass mapping.

Contributions. We summarize main contributions of this work as follows.

1. The loss function derived by Brenier theorem is beneficial for minimizing the error between the manifold discovered from the reconstructed data distribution and the original manifold, since Brenier theorem allows the data reconstructed distribution to be as close as possible to the original data distribution by calculating the optimal transportation mass distance between these two distributions.

2. It is recommended to constrain neurons with norm-2 in terms of easing data sparsity and improving the precision of for manifold discovery, instead of constraining neurons with norm-1.

3. Those nonlinear architectures with deep paradigms outperform those architectures without possessing deep paradigms in preventing the extraction of deformed manifolds.

II. METHODS

A. THEORY

Lemma 1 [18] (Brenier theorem). Suppose $X$ and $Y$ are the Euclidean space $\mathbb{R}^n$, and the transportation cost is the quadratic Euclidean distance $c(x, y) = |x - y|^2$. If $u$ is absolutely continuous and $u, v$ have finite second order moments, then there exists a convex function $\mu : X \rightarrow \mathbb{R}$, such that the gradient map $\nabla \mu$ gives the unique solution to the Monge’s problem, where $u$ is called Brenier’s potential, $v_\mu$ is called Brenier map or the optimal transportation mapping. In general, $u$ is not unique.

Lemma 1 implies that Brenier’s potential is equivalent to the optimal transportation mass mapping. Brenier’s potential $u$ and the Kantorovich’s potential $\phi$ is related by [19], as following

$$u(x) = \frac{1}{2} |x|^2 - \phi(x) \quad (1)$$

Using Eq. (2), the Brenier’s potential can be converted to the Kantorovich’s potential, so we can obtain the Brenier’s potential through calculating the Kantorovich’s potential. The calculation of the Kantorovich’s potential is equivalent to calculation the Wasserstein distance [20]. Wasserstein distance $W_c(u, v)$ between two measurements $u$ and $v$ is given by [21], having that

$$W_c(\mu, v) := \max_{\nu} \left\{ \int_{\mathbb{R}^n} \phi(x)d\mu(x) - \int_{\mathbb{R}^n} \phi(y)d\nu(y) \right\} \quad (2)$$

Above analysis shows that finding a loss function from the optimal transportation point of view is equivalent to calculation the Wasserstein distance. Wasserstein distance is difficult to calculate, currently, there are only two cases that can be calculated, one is the case that the dimensionality is equal to 1, and the other is the Gaussian distribution [22]. In practical applications, Gaussian distribution can be used to calculate the Wasserstein distance [22]. Hence, Eq. (2) can be converted to Eq. (3).

$$\left\{ W_c(X, Y) = \sqrt{\sum_i m_i - \mathbb{E}} + \text{tr}[\sum_i \Sigma_i^{-1/2} \Sigma_i^{1/2} \Sigma_i^{-1/2} \Sigma_i^{1/2}] \right\}$$

$$c(x) = \frac{\mathbb{E}}{\sum_i \Sigma_i^{-1/2} \Sigma_i^{1/2} \Sigma_i^{-1/2} \Sigma_i^{1/2}}$$

(3)

Where $\text{tr}[\cdot]$ is a trick of a matrix. $m1, m2$ are the mean of $X$ and $Y$, respectively.

B. MODEL IMPLEMENTATION

To achieve manifold discovery from the data in the high-dimensional space, and to visualize the relationships of between the high-dimensional data by the captured manifolds, a deep model with double hidden-layers was designed, namely Two-DM, as shown in Fig.1, through using the backpropagation manner to update the loss function $L_f(w, b)$ in Eq. (8), the error between the extracted manifold and the original manifold can be minimized.

In the proposed Two-DM, hidden layers include encoding hidden-layers and decoding hidden-layers,
respectively, the two types of hidden layers are described as following.

In the encoder, the input $E_{ni}(k,i)$ and the output $E_{out}(k,i)$ of the $k$-th $(k=1,2)$ hidden layer in the $i$-th $(i>0)$ iteration are calculated using Eq. (4) and Eq.(5), respectively. Having

$$E_{out} = \Theta(w^i_k E_{ni}(k,i) + b^i_k)$$  \hspace{1cm} (4)$$

$$E_{in}(k,i) = E_{out}(k - 1,i)$$  \hspace{1cm} (5)$$

In the decoder, correspondingly, the input $D_{ni}(n,i)$ and the output $D_{out}(n,i)$ of the $n$-th $(n=1,2)$ hidden layer in the $i$-th iteration are calculated as following.

$$D_{out} = \Theta(w^i_n D_{ni}(n,i) + b^i_n)$$  \hspace{1cm} (6)$$

$$D_{in}(n,i) = D_{out}(n - 1,i)$$  \hspace{1cm} (7)$$

Where $\Theta(\cdot)$ is an activation function. $w, b$ are the weights and bias of each hidden layer, respectively.

The loss function $L_f(w, b)$ of Two-DM consists of the Wasserstein distance $W_f(X,Y)$ in Eq. (3) and squared loss error $\Delta e$, as follow

$$L_f(w, b) = \min[\Delta e + W_f(X,Y)]$$

$$\Delta e = \sum\|e_{ni} - \hat{e}_{out}\|$$

$$\hat{e}_{out} = \Theta(\sum w \bullet \Theta(\sum w \bullet e_{ni}) + b) + b$$

Where $e_{ni}, \hat{e}_{out}$ are the inputting and the reconstructed inputting, respectively.

Compared with other optimizer, e.g., SGD, Momentum and Nesterov, etc, Adam is suitable for high-dimensional data than them.

Learning rate. Adam can provide dynamic learning rates for different tasks [23]. Clearly, we do not need to initialize the learning rate for Two-DM after using Adam as the optimizer.

Activation function. We opt for the eLu (Exponential Linear Units) function, i.e.,

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ \sigma(e^x - 1), & \text{otherwise} \end{cases}$$

Since eLu can solve the gradient vanishing problem, and it also has the advantages of activation function ReLU. Moreover, the convergence speed is much faster than activation function Sigmoid and tanh.

For the rest of hyper parameters, their default values were used since they have no substantial effects on results.

| Table 1 Algorithm 1. Training for Two-DM |
|------------------------------------------|
| 1 Initialization hyper parameters.       |
| 2 Input sample dataset $X$;              |
| 3 $X_{train}$ is gotten by random selecting 80% of $X$;  |
|   */ training set */                     |
| 4 $X_{test}$ is gotten by random selecting 20% of $X$; |
|   */ testing set */                     |
| 5 for $q=1$ to $Q$ do:                     |
| 6 Using gradient descent method to train Two-DM$q$; |
| 7 Calculate training accuracy $TraAcc(q) = Two-DM_q(X_{train})$; |
| 8 Test the Two-DM;                        |
| 9 Calculate testing accuracy $TeAcc(q) = Two-DM(X_{test})$; |
| 10 end for                                |
| 11 Select $q$ so that $q_{max} = \arg\max_q(TraAcc(q))$; |
| 12 Get the maximum training accuracy $TraAcc = Two-DM_q(X_{train}, q_{max})$ in the $q_{max}$-th iteration; |
| 13 Get the maximum testing accuracy $TeAcc = Two-DM(X_{test}, q_{max})$; |
| 14 Output $TraAcc, TeAcc$;                |

Training deep models is to train their hyper parameters, so that the hyper parameters are optimal. During the training of Two-DM, we dynamically adjust the iteration epoch by observing the training accuracy until the model converges, and then stop training. Noting that, as for the Two-DM, during training, we did not turn off any neurons and did not implement any constraint for neurons. This is to verify that our method can resist data sparsity in a certain extent.

Table 1 shows the training algorithm of Two-DM. In algorithm 1, hyper parameters are firstly initialized and the division of sample data is given in between step1 to step4.

Then, the proposed Two-DM is trained $q$ times until it converges, as shown in step5 to step10. During training, gradient descent method is used to update the weight $w$. After Two-DM is well trained, the testing set is inputted to test it. Finally, the maximum training accuracy and testing accuracy are outputted as shown in the procedure of between step11 and step14.

![FIGURE 1. The structure of the model.](image)
III. EXPERIMENT SETTING

A. DATASETS
The four synthetic datasets are generated using the same dimensionality by considering different sparsity, of each synthetic dataset has the same dimension, i.e., 1000 dimensions. From datasets s1 to s4, data sparsity reduces gradually. In addition, five real-world datasets [24] are also considered from three aspects, including data dimensionality, data sparsity and data volume. Table 2 lists the details of four synthetic datasets and five real-world datasets.

| Table 2 Description of experiment datasets |
|--------------------------------------------|
| Synthetic Datasets | Data Dimension | Data volume | Sparseness |
| s1               | 1000           | 3600        | 0.90       |
| s2               | 1000           | 3600        | 0.93       |
| s3               | 1000           | 3600        | 0.96       |
| s4               | 1000           | 3600        | 0.98       |

| Real-world Datasets (Serial number) | Data Dimension | Data volume | Sparseness |
|-----------------------------------|----------------|-------------|------------|
| ML-prove (R1)                     | 58             | 6115        | 0.9865     |
| CNAE (R2)                         | 857            | 1080        | 0.9355     |
| FPS (R3)                          | 4813           | 3600        | 0.9368     |
| Zemberek-Stemmed (R4)             | 5692           | 3600        | 0.9372     |
| TTC-Original (R5)                 | 7507           | 3600        | 0.9402     |

B. ASSESSMENT METRICS AND COMPETING METHODS
The receiver operating characteristic curve (ROC) and corresponding area under curve (AUC) are used to assess the accuracy of methods. To observe the effects of sparsity on manifold discovery, data sparsity needs to be calculated. For calculation of data sparsity, Patrik [25] give a calculation method, which is based on the relationship between norm-1 and norm-2. Hence, we use the formula given by Patrik to assess data sparsity, having that

\[ sparseness(x) = \frac{\sqrt{n} - (\sum |x_i|) / \sqrt{\sum x_i^2}}{\sqrt{n} - 1} \]  \hspace{1cm} (9)

Where \( x \) is a dataset. The item \( sparseness(x) \in [0,1] \). The assessment result of data sparsity is inversely proportional to the value of \( sparseness(x) \).

We compared the Two-DM with the manifold learning methods, e.g., LLE [5], MDS [5], Isomap [5]. In addition, the deep model SAE [14] was used as a comparison. To address convincing results, for the four competing methods, the optimal parameters of them observed in the corresponding literature are adopted. Unless otherwise stated, all experiments run on the same GPU, using the same environment.

IV. RESULTS

A. ACCURACY OF MANIFOLD DISCOVERY
Results on the synthetic datasets show that the performance of Two-DM wins that of competitors, as shown in Fig.2 (a). From the perspective of data sparsity, in the case of a fixed data volume and dimensionality, the performance of competitor drops as data gets more sparse in Fig.2 (a), while the Two-DM still shows advantages on sparse datasets. Even on extremely sparse datasets, for instance, on the synthetic dataset s4 (sparsity is equal to 0.98), the accuracy of Two-DM also reaches 0.91, whose AUC is 33% higher than that of LLE. (LLE obtains the lowest accuracy). Clearly, manifold learning approaches have no advantages for manifold discovery on the sparse high-dimensional datasets. To clarify the detailed experimental results, Fig.3 presents the visual representations of manifolds discovery.

![Figure 2](image-url)  
**FIGURE 2.** Accuracy of manifold discovery.

Results on real-world datasets in Fig.2 (b) show that the performance of Two-DM outperforms that of competitors. From the primary results, we clearly observe that the performance of Two-DM and competing methods declines in encountering the ‘curse of dimensionality’ and high sparsity, e.g., on the dataset R5. In the case, Two-DM still has advantages (AUC reaches 0.77), while the precision of competitive methods are all low below 0.62. This means that those non-linear architectures with deep paradigms exceed those non-linear architectures without deep paradigms, e.g., MDS, and those linear architectures without deep paradigms, e.g., LLE, in terms of manifold discovery in sparse high-dimensional space.
FIGURE 3 Visualization of manifolds discovery on the synthetic datasets

FIGURE 4 Training/testing accuracy of Two-DM.

(a) Results on synthetic datasets.

(b) Results on real-world datasets.
Fig. 4 displays the training and testing accuracy of the Two-DM on the synthetic and real-world datasets. It can be seen that the proposed method did not appear the tendency of over-fitting. Compared with the competing methods, Two-DM can effective handle sparse high-dimensional datasets, because the loss function derived by the Brenier theorem can minimize the error between predicted results and real results, so as to present a good robust. Moreover, the data distribution reconstructed by the optimal transportation mass distance can be as close to the original data distribution as possibility, so as to greatly reduce the probability of manifold distortion obtained from the reconstructed data distribution. As such, the proposed Two-DM has more advantages than competitors in terms of manifolds discovery in sparse high-dimensional datasets.

B. Improved precision comparison
To improve the precision of manifold discovery through overcoming data sparsity, we carried out two group experiments on the four synthetic datasets and the five real-world datasets. In Two-DM, we used norm-1 and norm-2 to constrain weight of neurons, respectively.

![AUC of synthetic datasets](image)

(a) AUC of synthetic datasets

![AUC of real-world datasets](image)

(b) AUC of real-world datasets

FIGURE 5 Results of overcoming data sparsity

Results in Fig. 5 show that the two approaches can overcome data sparsity, thereby improving the precision of manifold discovery. Clearly, constraining weight of neuron by using norm-2 is better than that of by using norm-1 in terms of overcoming data sparsity. Certainly, we only explore some approaches of improved precision on manifold discovery from the view of overcoming sparsity. Hence, the value of our approach is to provide some insights for overcoming data sparsity.

Several observations can be obtained from Fig. 2-5.
(i) The loss function derived by the Brenier theorem can help that models minimize the error between the manifold discovered from the reconstructed data distribution and the original manifold.
(ii) Those non-linear architectures possessing deep paradigms have more advantages than those architectures without possessing deep paradigms on manifold discovery.
(iii) Using norm-2 to constrain neurons wins using norm-1 to constrain neurons in easing data sparsity and improving the precision of manifold discovery.

V. CONCLUSION
In this work, we investigated the problem of manifold discovery in the high-dimensional space. To address this issue, we designed a deep model based on Brenier theorem. Results show that the proposed method wins competing methods in the accuracy of manifold discovery. This means that those non-linear architectures possessing deep paradigms exceed those non-linear and linear architectures without deep paradigms in terms of manifold discovery from the data in the high-dimensional space. In the future, we will explore the approaches of being resistant to noise for manifold discovery.

Declarations
The authors declare no conflict of interest

Acknowledgment
This work was supported by the Chongqing Municipal Education Commission of China under Grant 173227.

REFERENCE
[1] Decebal Constantin Mocanu, Elena Mocanu and Peter Stone. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science [J], Nature Communications 9 (2018), 1-18.
[2] Peter Andras. High-Dimensional Function Approximation With Neural Networks for Large Volumes of Data [J], IEEE Transactions on Neural Networks and Learning Systems 29 (2018), 500-508.
[3] Alicia N Schep, Beijing Wu and Jason D. BuenrostroChromVAR: inferring transcription-factor-associated accessibility from single-cell epigenomic data [J], Nature Methods 14 (2017), 975-978.
[4] Yonathan Aflalo, Anastasia Dubrovina and Ron Kimmel. Spectral Generalized Multi-dimensional Scaling [J], International Journal of Computer Vision 118 (2016), 380-392.
[5] Bo Yang, Ming Xiang and Yupei Zhang. Multi-manifold Discriminant Isomap for visualization and
Ayasha, S., Hanif, M. K., and Talib, R. Overview and comparative study of dimensionality reduction techniques for high dimensional data [J], *Information Fusion* **59** (2020), 44–58.

Zheng, X., Yuan, Y., and Lu, X. Dimensionality reduction by spatial-spectral preservation in selected bands [J], *IEEE Transactions on Geoscience and Remote Sensing* **55**(9) (2017), 5185–5197.

Segler, M. H. S., Preuss, M. and Waller, M. P. Planning chemical syntheses with deep neural networks and symbolic AI [J], *Nature* **555** (2018), 604-610.

Md Junayed Hasan, Muhammad Sohaib, Jong-Myon Kim. An Explainable AI-Based Fault Diagnosis Model for Bearings [J], *Sensors* **21**(4070) (2021), 1-34.

Lei Xiong, Kui Xu and Kang Tian. SCALE method for single-cell ATAC-seq analysis via latent feature extraction, *Nature Communications* **10** (2019), 1-10.

Yue Deng, Feng Bao and Qionghai Dai. Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning [J], *Nature Methods* **16** (40) (2019), 311-314.

Houssam BENBRAHIM, Haná HACHIMI and Aouatif AMINE. Deep transfer learning with Apache Spark to detect COVID-19 in chest X-ray images [J], *Romanian Journal of Information Science and Technology* **23** (2020), S117-S129.

Mihai-Corneliu and CRISTESCU. Machine learning techniques for improving the performance metrics of functional verification [J], *Romanian Journal of Information Science and Technology* **24** (2021), 99-116.

Chenliang Liu, Kai Wang, Yalin Wang and Xiaofeng Yuan. Learning deep multi-manifold structure feature representation for quality prediction with an industrial application [J], *IEEE Transactions on Industrial Informatics* **24** (2021), 1-1.

Aparajita Khan and Pradipta Maji. Multi-Manifold Optimization for Multi-View Subspace Clustering [J], *IEEE Transactions on Neural Networks and Learning Systems* **19** (2021), 1-13.

Tong Zhang, Wenming Zheng and Zhen Cui. Deep Manifold-to-Manifold Transforming Network for Skeleton-Based Action Recognition [J], *IEEE Transactions on Multimedia* **22** (2020), 2926-2937.

Wei Hu, Qianjiang Hu, Zehua Wang and Xiang Gao. Dynamic Point Cloud Denoising via Manifold-to-Manifold Distance [J], *IEEE Transactions on Image Processing* **30** (2021), 6168-6183.

Brenier and Yann. Polar factorization and monotone rearrangement of vector-valued functions [J], *Commun. Pure Appl. Math* **44**(4) (1991), 375–417.

Chen, Haodi, and Huang. Convergence rate estimates for aleksandrov’s solution to the monge-ampere equation [J], *Siam Journal On Numerical Analysis* **57** (1)(2019), 173-191.

Na Lei, Kehua Su and Li Cui. A geometric view of optimal transportation and generative model [J], *Computer Aided Geometric Design* **68** (2019), 1-28.

Kantorovich and L.V. On a problem of Monge, Usp [J], *Mat. Nauk* **3** (1948), 225–226.

Panaretos, Victor M., and Yoav Zemel. Statistical aspects of Wasserstein distances [J] *Annual Review of Statistics and Its Application* (2018).

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization [J], *arXiv preprint arXiv* 1412.6980v8 (2015).

[24] [OE/OL] http://archive.ics.uci.edu/ml/index.php

Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints [J], *Journal of Machine Learning Research* **5** (2004), 1457-1469.