Approximation of Geodesic Distances by Autoencoder Networks

Jiaxi Yang

Abstract— **Autoencoder network is a kind of network structure in current deep neural network. In recent years, autoencoder networks have been used in the tasks of dimensionality reduction of various data and feature extraction of complex things. Geodesic distance describes the local shortest path between two points on a spatial manifold, which is an essential measure of the manifold.This paper uses the Swiss roll function as the main research function, combines the auto-encoding network with the geodesic distance, uses the auto-encoding network to input the coordinates of the points in the space as input values into the auto-encoding network, and extracts and trains the features of the point coordinates. A new set of high-dimensional coordinates is obtained, so that the Euclidean space distance obtained from this set of high-dimensional coordinates can be closer to the geodesic distance in the actual space. The research content of this paper mainly includes the following three parts:**

(1) The development, concept and theory of auto-encoding network are introduced, and the principle of auto-encoding training is briefly described.

(2) The Floyd algorithm is used to solve the geodesic distance between two points of the Swiss roll function, and the approximate geodesic distance is obtained.

(3) Use the auto-encoding network to train and test the coordinates of the points in the Swiss roll function to obtain a new set of coordinates, calculate the Euclidean space distance of the new set of coordinates, and use this set of Euclidean distances with the geodesic distance measured by Floyd Make a comparative estimate and calculate the error.

Index Terms—**Autoencoder Network, Geodesic Distance, Swiss Roll Function, Feature Extraction, Floyd's Algorithm**

I. Introduction

In 1986, Rumelhart proposed the concept of self-encoding network, and applied it to high-dimensional complex data processing, which promoted the development of neural network. In 2006, Hinton improved the original self-encoding network structure, and then produced DAE, which applied the unsupervised layer-by-layer greedy training algorithm and the BP algorithm to pre-train the hidden layer and systematically parameterize the entire neural network. The optimization adjustment significantly reduces the performance index of the neural network, and effectively improves the bad situation that the BP algorithm tends to fall into the local minimum. In 2007, Benjio deepened the research of DAE and proposed the concept of sparse autoencoder. In 2008, in order to prevent overfitting, Vincent proposed a noise reduction auto-encoder, adding corrupt vectors to the input data, and achieved good results. In 2010, Salah proposed shrinking autoencoder networks, which limited the process of dimensionality raising and dimensionality reduction. In 2011, Jonathan proposed convolutional autoencoders for building convolutional neural networks. In 2012, Taylor discussed the connection between DAE and unsupervised feature learning in depth, detailing how to use autoencoders to build different types of deep structures. Hinton, Bengio and Vincent et al. compared the performance of prototype autoencoder network, sparse autoencoder network, denoising autoencoder network, shrinkage autoencoder network, convolutional autoencoder network and RBM and other structures, which provided insights for future research and practice. refer to. In 2013, Telmo studied the performance of DAE with different cost function training, which pointed out the direction for the development of cost function optimization strategies.

The idea of self-encoding networks has always been part of the historical landscape of neural networks. Traditional autoencoder networks are often used for dimensionality reduction or feature learning. In recent years, the association of autoencoding networks with latent variable model theory has brought autoencoding networks to the forefront of generative modeling. Autoencoder networks can be viewed as a special case of feedforward networks and can be trained using the exact same technique. But unlike general feedforward networks, autoencoder networks can also be trained using recirculation, a learning algorithm based on comparing the activations of the original input and the activations of the reconstructed input.

The Restricted Boltzmann Machine (RBM) was originally named the Harmonium by the inventor Paul Smuggler in 1986, but it was not until the invention of the fast learning algorithm by Jeffrey Hinton in the mid-2000s. The Boltzmann machine was only known to the world. Restricted Boltzmann machines have been used in different aspects such as dimensionality reduction, classification, collaborative filtering, feature learning, and topic modeling. Depending on the task, restricted Boltzmann machines can be trained using supervised or unsupervised learning methods.

II. Related Work

A. Autoencoder Network

Neural network is an algorithmic mathematical model that imitates the behavioral characteristics of animal neural network for information processing. This kind of network depends on the complexity of the system, and achieves the purpose of processing information by adjusting the interconnected relationship between a large number of

Manuscript received October 09, 2022.
Jiaxi Yang, School of Computer Science and Technology, Tiangong University, Tianjin, China.

internal nodes.

Autoencoder network is a type of neural network. It is a neural network that uses backpropagation algorithm to make the output value equal to the input value. It first compresses the input into a latent space representation, and then reconstructs the output through this representation.

The auto-encoding network consists of three parts, namely the encoder, the hidden layer and the decoder:

Encoder: This part compresses the input into a latent space representation, which can be represented by an encoding function $h = f(x)$.

Decoder: This part will reconstruct the input from the latent space representation, which can be represented by a decoding function $r = g(h)$.

Therefore, the entire autoencoder network can be described by a function $g(f(x)) = r$ where the output r is close to the original input x. Simply put, the auto-encoding network encodes the input expression X into a new expression Y, and then decodes Y back to X, which is an unsupervised learning algorithm.

An autoencoder network is a data compression algorithm in which the data compression and decompression functions are data-dependent, lossy, and automatically learned from samples. In most of the mentioned cases of autoencoders, the functions of compression and decompression are implemented by neural networks.

(1) Autoencoders are data-dependent, which means that autoencoders can only compress those data that are similar to the training data.

(2) Autoencoders are lossy, meaning that the decompressed output is degenerate compared to the original input, which is different from lossless compression algorithms.

(3) Autoencoders are automatically learned from data samples, which means that it is easy to train a specific encoder on inputs of a given class without any new work being done.

Building an autoencoder requires the following three tasks: building an encoder, building a decoder, and setting a loss function. A measure of the information lost due to compression. Encoders and decoders are generally parameterized equations and can be derived with respect to the loss function, typically using a neural network. The parameters of the encoder and decoder can be optimized by minimizing the loss function.

B. Geodesic distance

Geodesic distance is to find the shortest distance between two points on the surface of a three-dimensional object.

On a sphere, there are two points A and B. The straight line segment in the space from point A to point B is the Euclidean distance of two points AB. The curve along the spherical surface from point A to point B is geodesic Wire. That is to say, if you want to get from point A to point B, you cannot proceed in a straight line from the sphere, but only in a curved line from the surface of the sphere. Among the countless geodesic lines from A to B, the length of the shortest one is the geodesic distance of AB.

III. Solving for Geodesic Distance Approximation

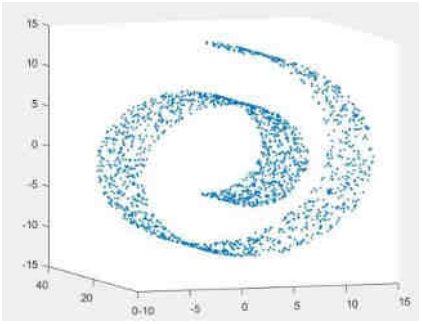
Geodesic distance is an essential measure of a manifold. The significance of machine learning is to flatten high-dimensional and complex manifolds, so that the Euclidean distance of new coordinate values can be used to approximate its geodesic distance. The content of this chapter processes the data using an autoencoding network, compares it with its geodesic distance, and presents its analysis.

In this paper, the Swiss roll function is selected as the function graph to be calculated.The swiss roll function is:

$$\begin{cases} t = \dfrac{3\pi}{2} \times (1 + 2 \times rand) \\ x = t \times \cos(t) \\ y = 21 \times rand \\ z = t \times \sin(t) \end{cases} \quad (1)$$

Among them, rand is a random number between. The swiss roll pattern is:

Figure 1. Swiss roll function



First, write the Floyd algorithm function according to the concept of Floyd algorithm:

$$function[dist, mypath, o] = myfloyd(a, sb, db) \quad (2)$$

Among them, a represents the adjacency matrix refers to the Euclidean distance between i and j, which can be directed, sb represents the label of the starting point, db represents the label of the end point, dist represents the shortest distance, mypath represents the shortest path, o Represents the geodesic distance of each point to other points. After that, the data distance matrix of the function is brought into the function to calculate the geodesic distance between every two points.

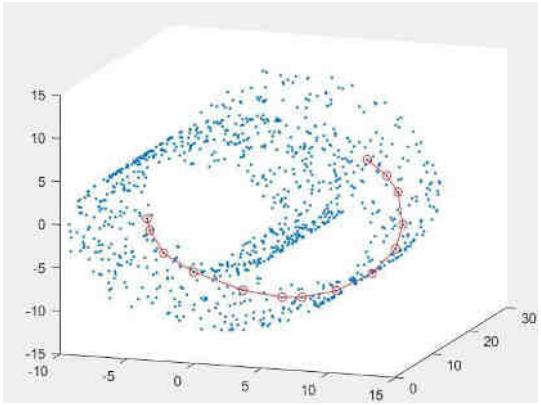
First of all, the shortest distance between two points obtained by Floyd's algorithm can be approximately equal to the geodesic distance between two points, but it is not the real geodesic distance.

In the experiment, when N increases gradually, with the continuous increase of the number of Swiss roll points, it can be approximated as a Swiss roll manifold, and the approximate distance between two points is also closer to the real manifold. Geodesic distance between points. At the same time, when the selected k value continues to increase, although the approximate distance between all two points can be calculated, because the selected k value is too large, the Euclidean distance between two points may appear in the Euclidean distance matrix. The Euclidean distance of the space is not an approximate geodesic distance, so the shortest

distance between the two points finally found cannot be approximated as a geodesic distance. When the selected k value continues to decrease, the distance between the points in the Euclidean distance matrix will decrease, and the Euclidean distance across the space will also decrease, and the shortest distance between two points cannot be approximated. At the same time, the obtained shortest distance can be closer to the geodesic distance. However, due to the low value of k selected, it may cause that two points cannot be connected, that is to say, the approximate geodesic distance cannot be formed between the two points.

Therefore, when we select the k value, we should select an appropriate k value, which cannot be too large or too small. It is necessary to connect all points to form the shortest distance, and to make the entire shortest distance closer to reality. the geodesic distance.

Figure 2. Geodesic distance when k=20



IV. SELF-ENCODING NETWORK SOLUTION

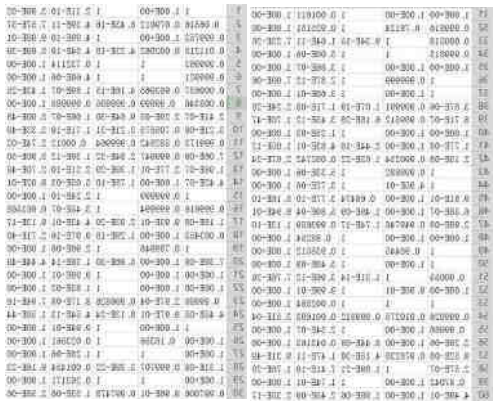
A. *Experiment*

In this paper, matlab is used to solve the self-encoding network. First, the Swiss roll function is used to generate 1000 coordinate points in the coordinate axis.

In this experiment, a self-encoding network with 3 nodes in the input layer and 5 nodes in the output layer is used.

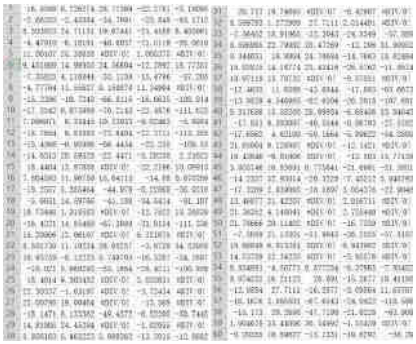
After that, the values of the X, Y, and Z axes of the 1000 points are brought into the auto-encoding network as three input values, and the training, testing and input are carried out to obtain 1000 groups of training data with 5 values in each group.

Table 1. Output 5D coordinate activation value



Since the auto-encoding network uses the activation function of sigmoid, that is, the function F, it is necessary to use the inverse function of sigmoid to restore the 5 values of each set of data to the original values.

Table 2. Five-dimensional coordinate original value



In the training process, it can be regarded as encoding the three-dimensional coordinates of a point into five-dimensional coordinate values, and then by calculating the Euclidean distance of the five-dimensional coordinates between the two points, the distance between the two points is approached to the geodesic distance.

B. *Error Analysis*

This paper uses a method to measure the error rate of Euclidean distance and geodesic distance, that is, $G_M(i,j)$ represents the geodesic distance between point i and point j in the upper triangular matrix in the geodesic distance matrix calculated by Floyd. Use $G_E(i,j)$ to represent the Euclidean distance between point i and point j in the upper triangular matrix in the Euclidean distance matrix calculated by the self-encoding. So we can write the error metric as:

$$d = \frac{2}{N(N-1)} \sum_{(i,j)} \frac{\left|G_M(i,j) - G_E(i,j)\right|}{\left|G_E(i,j)\right|} \qquad (3)$$

Smaller values of d occur when pairs of geodesic and Euclidean distances have values that are closer to each other. For an ideal flat manifold, the values of c and d should be equal to 0.

Through calculation, we can get d=0.5705, which means that the error rate of the Euclidean distance calculated by the self-encoding network and the geodesic distance calculated by the Floyd algorithm is 0.5705.

In the experiment, it can be found that in multiple experiments with the same set of data and the same number of output nodes, the error rate will fluctuate slightly, but because the change is too small, it can be ignored in this experiment. The error rate does not change.

Through experiments, we can initially find that when the number of output layer nodes is about 10, the error rate does not change much, and it basically remains at about 0.6. However, when the number of output layer nodes is too large, the error rate will start to become larger, that is, the Euclidean The error between the distance and the geodesic distance

becomes larger. Therefore, when we select the output layer nodes, we should not select too many output nodes, and should select about 10 output nodes to ensure the minimum error rate.

V. CONCLUSION

A Through this work, it is learned that the self-encoding network is composed of multiple processing layers, including an input layer, multiple hidden layers and an output layer. Each layer is connected to each other through nodes or neurons. Each hidden layer is used before The output of a layer serves as its input value. And the self-encoding network is unsupervised training. When a set of data is input, the self-encoding network will train itself and obtain the output value.

First, an auto-encoding network with three input values and five output values is used. First, the Swiss roll function is used to calculate the coordinate values of 1000 points on the coordinate axis, and then the Floyd algorithm is used to calculate the 1000 points every two Approximate geodesic distance between points. Then use the X, Y, Z axis coordinate values of 1000 points as the three input values of the input layer of the auto-encoding network, and perform self-training and testing to obtain 1,000 sets of 5-dimensional data, and use these 1,000 sets of five-dimensional data for five-dimensional data. Calculate the Euclidean distance and compare it with the approximate geodesic distance measured by Floyd, and compare the approximation of the two through the analysis of the error rate. Later, when the number of output nodes was changed to about 10, it was found that the error rate did not change much, and was basically maintained at about 0.6. When the output nodes were too high, the error rate began to increase. Therefore, when selecting the number of output nodes, About 10 output nodes should be selected, and too many output nodes should not be selected.

However, there are some defects in the whole experiment. First, the distance between two points found by Floyd's algorithm is only an approximate solution of the geodesic distance, not the real geodesic distance. Second, during the training of the autoencoder network, the output value will appear #DIV/0! , the new coordinates of the entire point cannot be calculated.

REFERENCES

[1] Pratik Prabhanjan Brahma, Dapeng Wu, Fellow, IEEE, and Yiyuan She,“Why Deep Learning Works: A Manifold Disentanglement Perspective”.1997

[2] Y. Bengio, A. Courville, and P. Vincent, “Representation learning:A review and new perspectives,” IEEE Trans. Pattern Anal. Mach. Intell.,vol. 35, no. 8, pp. 1798–1828, Aug.2013

[3] Geoffrey E. Hinton, Simon Osindero, Yee-Whye Teh,“A fast learning algorithm for deep belief nets”, Neural Computation,2006

[4] Q. Huang and D. O. Wu, “Flatten a curved space by kernel [applications corner],” IEEE Signal Process. Mag., vol. 30, no. 5, pp. 132–142,Sep. 2013.

[5] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” Science, vol. 313, no. 5786, pp. 504–507,2006.

[6] L. Cayton, “Algorithms for manifold learning,” Univ. California,San Diego, San Diego, CA, USA, Tech. Rep. CS2008-0923,2005

[7] J. M. Lee, Introduction to Smooth Manifolds. New York, NY, USA:Springer-Verlag, 2002.

[8] Lakshmi Priya Muraleedharan; Shyam Sundar Kannan; Ramanathan Muthuganapathy， “Autoencoder-based part clustering for part-in-whole retrieval of CAD models”.2019

[9] G. Hinton et al., “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” IEEE Signal Process. Mag., vol. 29, no. 6, pp. 82–97, Nov. 2012.