Manifold Learning Method for Large Scale Dataset Based on Gradient Descent

Yunhe Wang [1], Yuan Gao, Chao Xu
Key Laboratory of Machine Perception (Ministry of Education)
Peking University, Beijing, China

Abstract. Dimension reduction is a research hotspot in recent years, especially manifold learning for high-dimensional data. Cause high-dimensional data have complex nonlinear structures there are many researchers focus on nonlinear methods. The memory cost and running time are too large and difficult to operate when the scales of data are tremendously large. In order to solve this problem, we utilized the gradient descent to search the low-dimensional embedding. It replaced the eigenvalue decomposition of a large sparse matrix of LLE (Linear Locally Embedding) algorithm. The time complexity is lower than before and storage memory is declined obviously. Experimental results demonstrated our approach performed well than the original algorithm. Furthermore, our approach can be applied to other manifold method or other research fields such as information retrieval and feature extraction.

Keywords: manifold learning; LLE; gradient descent; time complexity.

1. Introduction

In recent years, dimensional reduction has received intense attention [1, 2]. There are many manifold learning methods have been proposed such as Locally Linear Embedding (LLE) [12], Isomap [11] and Laplacian Eigenmaps [5]. There are also many variants based on LLE and Spectral Clustering [6, 7].

However, there also have a lot of weaknesses caused by the scale of high-dimensional data we need to process is especially large. The first problem is to

---

[1]Yunhe Wang(✉)
Key Laboratory of Machine Perception (Ministry of Education), Peking University. 100871 Beijing, China
e-mail: princelnk023@gmail.com

store the large matrix. Although the matrix of LLE is a sparse matrix, it is much bigger than the high-dimensional data itself. The second problem is the running time of eigenvalue decomposition is enormous.

LLE algorithm is a classical manifold learning method, which uses neighborhood's information to reconstruct the high-dimensional data, and then maintains the locally relationship during the embedding process. In this paper, we focus on the improvement of LLE algorithm.

Our contribution in this paper is utilizing numerical method to search the low-dimensional embedding data. In this opinion, the gradient descent method [8] is introduced to search the low-dimensional embedding data. In this manner, the time complexity and memory consumption both reduced.

This paper is organized as follows: In Section 2, some related works of manifold learning methods are briefly described. In Section 3, in order to reduce the time complexity, we exploit gradient descent to search the low-dimensional embedding data. Section 4 shows the experimental results with different scale. Some conclusions and further studying are discussed in Section 5.

2. Related Works

This section reviews the state-of-the-art methods of manifold learning and several numerical searching algorithms.

Dimensionality reduction methods earliest can be traced back to principal component analysis (PCA) in 1901 [10], it has become a research hotspot in recent years. Science 2000, manifold learning has been developed rapidly, many classical algorithms such as LLE, Isomap etc. has been proposed [5, 11, 12, 14].However, existing dimensionality reduction methods are involved in large-scale matrix eigenvalue decomposition whether they are linear or nonlinear methods.

We follow LLE as an example [3, 4], when the D-dimensional data's scale is N and the embedding space is d-dimensional. We need to solve an eigenvalue decomposition problem with a large sparse matrix. The time complexity of this processing is $O(dN^2)$. Another major cost is to calculate the k-nearest neighborhood. Its time complexity is $O(DN^2)$. Although it is a polynomial time complexity, but there exist almost ten million features in a dataset for image retrieval [9]. In addition, store the large sparse matrix is very difficult. So, we need a more effective algorithm with low-level time complexity for image retrieval. In fact, there are many researchers endeavor to enhance the nonlinear manifold learning method, whether in precision or accuracy [13, 15]. However, they didn't solve the problems of LLE essentially.

In this paper, we endeavor to find a numerical method to reduce the running time and time complexity of LLE, thus it can be exploited to large-scale dataset. In this opinion, gradient descent was introduced to search the embedding data.

3. Fast Manifold Learning

Our manifold learning method is based on the Locally Linearly Embedding (LLE) [12]. Although many researchers presented some methods that performed well than LLE, but all of them are derived from it. So, we only need to focus on reduction the computing time of LLE algorithm.

3.1 LLE Algorithm

We briefly outline the steps of LLE algorithm firstly.

Step 1. Searching the *k*-nearest neighborhood of each data.

Step 2. Reconstruct the high-dimensional data exploiting their neighborhood's information and the error function can be calculated as follows.

$$\varepsilon(W) = \sum_i \left| X_i - \sum_j W_{ij} X_j \right|^2 \tag{1}$$

X_j denotes the j-th nearest neighborhood and W_{ij} is the reconstruction weight.

Step 3. Embedding the data to d-dimensional space. The significant of the algorithm is to find a set of low-dimensional data those can keep the locally linear relationship extract from the original data. The objective function is to minimum the total square error.

$$\Phi(Y) = \sum_i \left| Y_i - \sum_j W_{ij} Y_j \right|^2 \tag{2}$$

The d-dimensional data Y are the embedding results. In general LLE algorithm, this problem is converted into an eigenvalue decomposition problem of a large-scale sparse matrix. However, we can get the analytical solution by this way. But the time complexity is exponential level often. So, it is very difficult to get the embedding results when the scale of original data is very large. Focus on this problem we endeavor to search Y quickly.

3.2 Gradient Descent

Inspired by the training method of BP Neural Network, it trains the network's weights matrix utilizing the gradient descent [16, 18]. It is a numerical algorithm for optimization with linear time complexity and it costs less memory.

Cause there will exist some locally information in every dimension of *X*. We randomly select *d*-dimensional data to generate *Y* as the initial solution. The objective function is equivalent to minimizing Eqn. 2. Then, we can calculate the gradient of every individual of *Y* as follows.

$$\nabla \Phi\left(Y\right)=\left[\frac{\partial \Phi}{\partial Y_1},\frac{\partial \Phi}{\partial Y_2},\dots\frac{\partial \Phi}{\partial Y_n}\right] \tag{3}$$

Simultaneous Eqn.2 and Eqn.3 we can get:

$$\frac{\partial \Phi}{\partial Y_i}=2Y_i-2\sum_j W_{ij}Y_j \tag{4}$$

In gradient descent algorithm, optimal solution is obtained through continuous iteration. The increment of Y_i can be expressed as the product of its gradient and the learning rate.

$$\Delta Y_i=-\eta\frac{\partial \Phi}{\partial Y_i} \tag{5}$$

It is clearly that we get the optimal solution when the value of Eqn.2 is equal to 0. But, we can't get the exact analytical solution when the scale of *X* is enormous. However, we only need to search a set of *d*-dimensional data with the similar locally linear relationship as the original data *X*. So, we can simply set the termination condition as whether the error between two adjacent iterations is less than a threshold.

$$\left\|\Phi\left(Y\right)^{(n+1)}-\Phi\left(Y\right)^{(n)}\right\|^2\le\theta \tag{6}$$

Although we can't obtain the optimal solution of *Y* as we presented above. But, our approach can reduce the memory significantly cause we only need to calculate *Y* and its increment in every iteration. Especially, the method we presented will reduce the running time excellently when the scale of *X* is large cause its complexity is linear. Furthermore, the memories we cost are less than LLE algorithm cause we don't need to store the large sparse matrix.

Finally, we detail our fast manifold learning method based on the gradient descent, as shown in Algorithm 1.

Algorithm 1 Fast LLE Exploiting Gradient Descent

Input: Original data *X*, *k*-number of nearest neighbors
Output: *d*-dimensional embedding data
Locally Linear Embedding
for each data *x* **do**
  search the *k*-nearest neighbors
  calculate the reconstruction weight
end for
Gradient Descent
random generation the *d*-dimensional embedding data *Y*
while the error between two iterations less than the threshold **do**
  calculate the gradient increment of *Y*
  upload the *Y* of current iteration
  record the value of error function
end while
return the *d*-dimensional data *Y*

4. Experiment Results

The experiments are tested on a group of Swiss roll data have different scales. We have implemented the proposed algorithm using MATLAB 2012b with 4G memories. Firstly, we compared the two methods' efficiency of dimension reduction illustrated in Fig. 1.

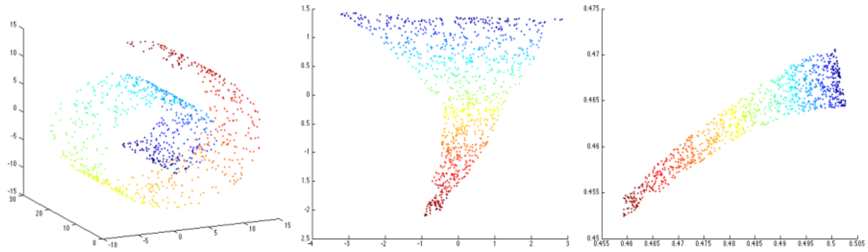

Fig. 1 Comparison of two methods: the first is a Swiss Roll data with 1000 points, the second picture is the embedding of LLE and the third is the results of our approach.

As showed in Fig.1, our method can better maintain the global structure. Another experiment was constructed on a Swiss roll with "hole" [17]. Swiss Roll can also be accurately embedding as shown in Fig. 2.

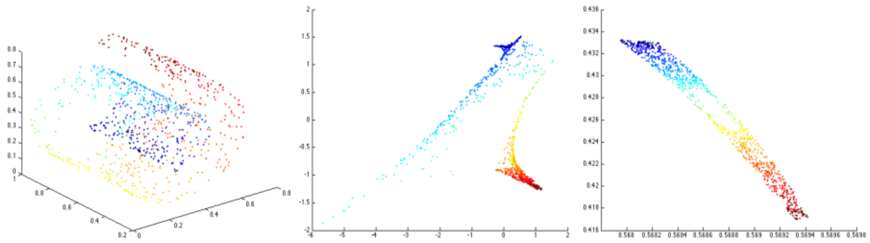

Fig. 2 Comparison of two methods, LLE can’t embed accuracy when the Swiss Roll have a hole inside. But, our approach has got a very excellent result.

Then, we detailed the running time of our approach on different scale data and iterations, as illustrated in Tab. 1. We can clearly observe that the running time increasing linearly with the number of iterations increasing, also the same to the scale of data. Especially, the increment of dimension won’t lead to increasing of running time.

Table 1 Running times with different conditions.

Iteration	Low-Dimension	Scale	Running Time (average ten times)
1000	2	1000	5.3s
5000	2	1000	27s
10000	2	1000	54s
10000	2	2000	111s
10000	3	2000	111s
10000	2	10000	550s

Finally, we detail the running time of two methods. As shown in Fig. 3, two experiments are both constructed on the same data from 1000 to 10000 points. For small-scale data, original algorithm’s running time is shorter cause the slow convergence of the iteration method. However, it will increase faster than our method with the data’s scale increasing. Compare with the original LLE algorithm, our approach based on gradient descent is obviously more suitable for large-scale data.

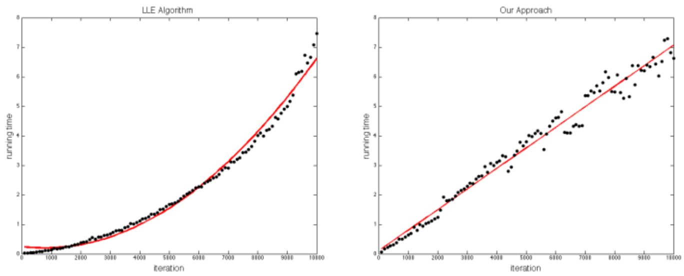

Fig. 3 Running times of LLE and our approach (wherein, the black points are measured data), it is obviously that our approach has a lower time complexity.

The problems of LLE algorithm for large-scale data can be divided into two parts: store the large sparse matrix and eigenvalue decomposition of it. Although it is a sparse matrix, but the memory of most personal computers don't have enough memories to store it. In the approach we presented, the extra data we need to store are the gradient increment of embedding data *Y*. So, occupy memories are twice than *Y*. Furthermore, our method has a low-level time complexity. It is more rational than the original method when the scale of data is large.

5. Conclusion

In this paper, we have presented a novel manifold learning approach that exploits gradient descent method for solving the low-dimensional embedding data of LLE algorithm. The main idea is to minimize the error function of reconstruction by an iterative method. Our experiments are constructed on a group of Swiss Roll data and results demonstrated the applicability of our method. The method we proposed can also be used in other applications.

Future work focuses on: (i) find more reasonable termination conditions to improve the convergence speed; (ii) try to use other rational numerical methods to improve the convergence speed; (iii) apply manifold learning method on the field of image retrieval.

6. References

1. Androulakis J, Peter S C, Li H, et al. Dimensional Reduction: A Design Tool for New Radiation Detection Materials[J]. Advanced Materials, 2011, 23(36): 4163-4167.
2. Izenman A J. Introduction to manifold learning[J]. Wiley Interdisciplinary Reviews: Computational Statistics, 2012, 4(5): 439-446.
3. Saul L K, Roweis S T. Think globally, fit locally: unsupervised learning of low dimensional manifolds[J]. The Journal of Machine Learning Research, 2003, 4: 119-155.
4. Liou C Y, Cheng W C. Manifold construction by local neighborhood preservation[C]. Neural Information Processing. Springer Berlin Heidelberg, 2008: 683-692.
5. Belkin M, Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering[C]. NIPS. 2001, 14: 585-591.
6. Zhang T, Yang J, Zhao D, et al. Linear local tangent space alignment and application to face recognition[J]. Neurocomputing, 2007, 70(7): 1547-1553.
7. Zhang Z, Zha H. Principal manifolds and nonlinear dimension reduction via local tangent space alignment[J]. arXiv preprint cs/0212008, 2002.
8. Burges C, Shaked T, Renshaw E, et al. Learning to rank using gradient descent[C]. Proceedings of the 22nd international conference on Machine learning. ACM, 2005: 89-96.
9. Li Y, Geng B, Zha Z, et al. Query expansion by spatial co-occurrence for image retrieval[C]. Proceedings of the 19th ACM international conference on Multimedia. ACM, 2011: 1177-1180.

10. Pearson K. LIII. On lines and planes of closest fit to systems of points in space[J]. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 1901, 2(11): 559-572.
11. Tenenbaum J B, De Silva V, Langford J C. A global geometric framework for nonlinear dimensionality reduction[J]. Science, 2000, 290(5500): 2319-2323.
12. Roweis S T, Saul L K. Nonlinear dimensionality reduction by locally linear embedding[J]. Science, 2000, 290(5500): 2323-2326.
13. Yu K, Zhang T, Gong Y. Nonlinear learning using local coordinate coding[C]//Advances in Neural Information Processing Systems. 2009: 2223-2231.
14. Niyogi X. Locality preserving projections[C]. Neural information processing systems. 2004, 16: 153.
15. Zhou Y, Barner K. Locality constrained dictionary learning for nonlinear dimensionality reduction[J]. 2012.
16. Watrous R L. Learning algorithms for connectionist networks: Applied gradient methods of nonlinear optimization[J]. Technical Reports (CIS), 1988: 597.
17. Lee J A, Verleysen M. Nonlinear dimensionality reduction of data manifolds with essential loops[J]. Neurocomputing, 2005, 67: 29-53.
18. Riedmiller M, Braun H. A direct adaptive method for faster backpropagation learning: The RPROP algorithm[C]. Neural Networks, 1993., IEEE International Conference on. IEEE, 1993: 586-591.