http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 11 (3): 380-383, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.380.383

A Fast Manifold Learning Algorithm

Lukui Shi and Junhua Gu
School of Computer Science and Engineering, Hebei University of Technology,
Tianjin 300401, China

Abstract: A fast manifold learning algorithm was proposed which only preserved the similarities or dissimilarities between each point and some objects. The embedded coordinates were obtained through optimizing the part stress function with the deepest gradient descent method. Experiments showed that the method could find the true structure of the manifold and had lower complexity. At the same time, we discussed the effectiveness of the number of far points and the initial value to the embedded results.

Key words: Manifold learning, mds, isomap, deepest gradient descent method

INTRODUCTION

The goal of dimensionality reduction is to map the high dimensional data into a low dimensional space and compress the data in size. PCA and MDS are classical linear techniques. Recently, manifold learning algorithms have been developed to perform nonlinear dimensionality reduction, such as ISOMAP (Tenenbaum *et al*., 2000), LLE (Roweis and Saul, 2000), LTSA (Zhang and Zha, 2004), Laplacian Eigenmaps (Belkin and Niyogi, 2003), Multilayer autoencoders (Hinton and Salakhutdinov, 2006) and so on.

ISOMAP is a generalization of MDS. It is becoming more and more difficult for MDS and ISOMAP to solve the full eigenvector problem with the increasing sample size. To overcome their fault, LMDS (Landmark MDS) (De Silva and Tenenbaum, 2003) only preserves distances or geodesic distances between some landmark points. However, random landmark point sets often lead to worse results.

In this study, we proposed a fast manifold learning algorithm which only preserved the similarities or dissimilarities between each point and its k nearest neighbors as well as some random far points. Experiments showed that the method was effective.

OVERVIEW OF MDS AND ISOMAP

Classical MDS is realized by matching the ordinal level of similarities or dissimilarities among all objects in two spaces. For the goal, Kruskal (1964) proposed a measure called the stress function which shows that how much the embedded data express the original data:

$$\text{stress} = \sqrt{\sum_{ij}(d_{ij} - \Delta_{ij})^2 \Big/ \sum_{ij} d_{ij}^2} \quad (1)$$

where, d_{ij} and Δ_{ij} are respectively the similarity or dissimilarity in the original space and in the embedded space.

ISOMAP replaces Euclidean distances with the geodesic distances on the base of MDS. The algorithm contains three steps. Firstly, the neighborhood graph is built. Then, the geodesic distances between all pairs are estimated by calculating the shortest path distances in the neighborhood graph. Lastly, the low dimensional coordinates are found by applying MDS to the geodesic distance matrix. The detail can be referred to Tenenbaum *et al*. (2000).

THE FAST ALGORITHM

The basic idea: In MDS and ISOMAP, solving the full eigenvector problem leads to higher complexity. Contrarily, LLE, Laplacian Eigenmaps and LTSA have lower complexity because the eigenvector problem is sparse in these methods. If MDS and ISOMAP also solve a sparse eigenvector problem, its execution will greatly speed up. Therefore, it is very necessary to construct a spare similarity or dissimilarity matrix. How can we build such a matrix? We can only preserve the similarities or dissimilarities between some points but all pairs.

LMDS has lower complexity through only preserving the distances between some landmark points. But random landmark sets don't often represent the true topology of the manifold and lead to worse results. Then how to reduce the complexity on the premise of guaranteeing the embedded quality? The problem can be solved by

Corresponding Author: Lukui Shi, School of Computer Science and Engineering, Hebei University of Technology, Tianjin 300401, China

combining the local structure and the global structure of the manifold. We can preserve the similarities or dissimilarities between each point and some points including its nearest neighbors and some far points.

Let X contain N samples. The goal of dimensionality reduction is to force Δ_{ij} to match d_{ij} for each possible pair (i, j). To do that, the stress function is used to measure the error between the pairwise distances in two spaces in MDS. The embedding is carried out by minimizing Eq. 1. Because only some similarities or dissimilarities are preserved in the new method, Eq. 1 can be changed to:

$$part\_stress = \sqrt{\sum_i \sum_{j\in\Omega} (d_{ij} - \Delta_{ij})^2 \Big/ \sum_i \sum_{j\in\Omega} d_{ij}^2} \quad (2)$$

where, Ω is the set containing the nearest neighbors and some random far points. We call Eq. 2 as the part stress function.

According to the part stress, we can introduce the part energy function:

$$E = \frac{1}{2}\lambda \sum_i \sum_{j\in\Omega} (d_{ij} - \Delta_{ij})^2 \quad (3)$$

where, λ is a learning rate parameter. Let:

$$E_i = \frac{1}{2}\lambda \sum_{j\in\Omega} (d_{ij} - \Delta_{ij})^2$$

Equation 3 is rewritten as:

$$E = \sum_i E_i \quad (4)$$

The minimization is made through the deepest gradient descent method. Let ∇E_i is the gradient with respect to y_i, then:

$$\Delta y_i = \nabla E_i = -\lambda \sum_{j\in\Omega} \frac{(d_{ij} - \Delta_{ij})}{\Delta_{ij}} (y_j - y_i) \quad (5)$$

Equation 5 gives the method to update the embedding coordinates for each step.

The description: The method is an iterative procedure where the low dimensional coordinates are gained through preserving the similarities or dissimilarities between each point and some points. During the iterative process, the part stress function is used as the object function. To simplify the algorithm, Euclidean distances are used to construct the neighborhood graph and the dissimilarity measure is only employed. The fast algorithm can be described as follows:

Input: X containing N samples, the neighborhood size k_1, the number k_2 of far points, the embedded dimension d, the max learning steps L, the min part stress s and the learning rate λ

Output: The embedded coordinates Y

Step 1: Decide k_1 nearest neighbors and randomly choose k_2 far points for each point

Step 2: Define the dissimilarity measure

Step 3: Estimate the dissimilarities between each point and its k_2 distant points

Step 4: Initialize Y with PCA or random method and set l = 1

Step 5: Figure out the part stress. If the part stress is smaller than s, return Y

Step 6: Update Y. Execute the following steps for each point i

- Calculate Δy_i according to Eq. 5
- Set $y_i^{l+1} = y_i^l + \Delta y_i$

Step 7: Set l = l+1. If l is bigger than L, return Y, else go to step 5

If Euclidean distances are used as the dissimilarity measure, the step 3 can be skipped. If the geodesic distances are as the dissimilarity measure, they can be approximated with the shortest path on the neighborhood graph. The shortest path can be computed with Dijkstra's algorithm.

The complexity analysis: In the method, a main computational cost is the iterative process. Its time complexity is O (c (k_1+k_2) N) (c is a constant). Usually, the sum of k_1 and k_2 is far below N. However, a full N×N eigenvector problem has O (N^3) complexity in MDS and ISOMAP. At the same time, a full N×N dissimilarity matrix needs to be stored. The space complexity is O (N^2). But, we only store a (k_1+k_2)×N matrix. Its space complexity is O((k_1+k_2)N). Another main cost is to estimate the dissimilarities between all pairs. The time complexity of the part is equal for the new algorithm and MDS. Apparently, the complexity of the fast algorithm is great degraded.

EXPERIMENTS

We tested the performance of the new algorithm and discussed some facts influencing the performance. The experiments included four parts: comparison of the performance between the new method and ISOMAP,

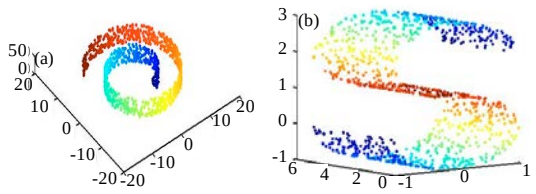

Fig. 1: Two data sets including 1000 samples. (a) Swiss roll data set and (b) S-curve data set

comparison of the efficiency of two methods, effectiveness of the number of far points and the initial value to the embedded results. In experiments, we used Kruskal (1964) stress to measure the performance and noted it as ks. The distance in two spaces was, respectively Euclidean distance and the geodesic distance. All programs were implemented in MATLAB.

Comparison of the performance between two methods: To compare the performance of ISOMAP and the proposed algorithm, we tested them on two data sets: Swiss roll data set (Tenenbaum *et al*., 2000) and S-curve data set (Roweis and Saul, 2000). Each set contained 1000 samples (Fig. 1).

Firstly, we ran ISOMAP once on each data set and computed the stress. Then, the fast algorithm was executed on the two sets. k_2 was 5, 10, 15 and 20. The initial value was set with PCA. For each case, the algorithm ran 50 times. We counted the min value, the max value, the mean value and the standard deviation of ks of 50 times. The four values were, respectively noted as min, max, mean and sd. The results were given in Table 1 and 2. Here, k_1 was 7 on Swiss roll data set and 20 on S-curve data set.

From the experiments, the stress from the new algorithm was close to or smaller than that from ISOMAP. Namely, the performance of the fast algorithm was close or superior to that of ISOMAP through only preserving the distances between each point and some objects.

Comparison of the executing efficiency of two methods: Theoretically, the complexity of the new algorithm was lower than that of ISOMAP. We compared their executing efficiency on Swissroll data set. We figured out the time of the iterative process for the fast method. For ISOMAP, we computed the time of solving the full N×N eigenvector problem. To validate the relation between the executing efficiency and the sample size, we ran them with different sample size. The size was from 1000 to 8000. Here, k_1= 7 and k_2 = 5, 10, 15, 20. The initial value was set with PCA.

Table 1: Results on Swissroll data set

	Fast algorithm				
ISOMAP	k_2	Min.	Max.	Mean	SD
0.0256	5	0.0262	0.0322	0.0284	0.0014
	10	0.0233	0.0237	0.0235	0.0001
	15	0.0227	0.0232	0.0229	0.0001
	20	0.0225	0.0229	0.0226	0.0001

Table 2: Results on S-curve data set

	Fast algorithm				
ISOMAP	k_2	Min.	Max.	Mean	SD
0.0066	5	0.0211	0.0245	0.0223	0.0008
	10	0.0195	0.0240	0.0219	0.0011
	15	0.0200	0.0254	0.0218	0.0010
	20	0.0198	0.0237	0.0212	0.0008

Table 3: Efficiency comparison between two algorithms for different sample size

		Fast algorithm (sec)			
Sample size	ISOMAP (sec)	$k_2 = 5$	$k_2 = 10$	$k_2 = 15$	$k_2 = 20$
1000	0.6240	2.7012	2.4915	2.3086	4.4760
2000	2.1100	3.3181	1.8563	1.4551	1.3879
3000	4.6570	4.6535	2.5735	1.9968	1.9421
4000	7.7360	6.0887	3.3256	2.5459	2.4221
5000	11.8140	7.3681	3.9369	3.0074	2.8496
6000	17.6450	8.3773	4.2411	2.9720	2.7502
7000		10.7493	5.6882	4.2804	4.0884
8000		11.7656	6.5844	5.5876	6.5873

The max iterative step was 100 and the min part stress was 0.02. For each case, the mean time of 10 times was calculated. The results were shown in Table 3.

From the results, ISOMAP could not run when the sample size was more than 6000 because the matrix was too big. For smaller sample set, the executing efficiency of ISOMAP was higher than one of the new algorithm. The efficiency of the fast method was higher and higher than one of ISOMAP with the increasing sample size.

Effectiveness of the number of far points to the embedded results: In the algorithm, one needed randomly select some far points before running the program. How many should we select far points? Namely, how does the number of far points affect the results? To this end, we ran the method on Swiss roll data set with 1000 samples for various numbers of far points. Here, k_1 = 7 and k_2 = 0, 1, 3, 5, 10, 20. For each case, we computed statistic values of ks for 50 times. During each execution, the max iterative step was 200 and the initial value was set with PCA. The results were given in Table 4.

Table 4: Values of ks for different far points

k_2	Min.	Max.	Mean	SD
0	0.6303	0.6303	0.6303	0.0000
1	0.1014	0.3327	0.1992	0.0508
3	0.0268	0.0362	0.0289	0.0016
5	0.0244	0.0270	0.0252	0.00052
10	0.0232	0.0246	0.0236	0.0003
20	0.0225	0.0234	0.0227	0.0002

Table 5: Results for the initial value with PCA

k_2	Min.	Max.	Mean	SD
1	0.2841	0.4314	0.3518	0.0309
3	0.0497	0.1000	0.0668	0.0097
5	0.0262	0.0322	0.0284	0.0014
10	0.0233	0.0237	0.0235	0.0001
20	0.0225	0.0229	0.0226	0.0001

Table 6: Results for the random initial value

k_2	Min.	Max.	Mean	SD
1	0.2925	0.5627	0.4519	0.0720
3	0.0867	0.4839	0.2265	0.0807
5	0.0471	0.2163	0.1376	0.0444
10	0.0236	0.1654	0.0640	0.0499
20	0.0225	0.1347	0.0270	0.0199

Experiments showed that the number of far points greatly influenced the embedded results. Without far points, the results were worse for the global information was ignored. The results would be greatly improved while only selecting one far point. The more far points were chosen, the better the results were. However, the results were near while far points were more than five.

Effectiveness of the initial value to the embedded results: In the fast algorithm, we needed to give an initial value of the low dimensional embedding before starting the iteration. Then, how did the initial value affect the results? In this section, we discussed how to set the initial value. Here, we adopted two methods to set the initial value. One was random initial value; the other was to use the results from PCA. Here, $k_1 = 7$ and $k_2 = 1, 3, 5, 10, 20$. For each case, we computed the statistic values of ks for 50 executions. The results were given in Table 5 and 6 for two cases.

As shown in the experiments, the initial value greatly affected the embedded results, especially, far points were less. The results were closer and closer with the increasing number of far points for two cases. For the initial value with PCA, the min value, the max value and the mean value of the stress were near while far points was more than five. The standard deviation was also very small. However, the three values were very various and the standard deviation was also bigger for the random initial value.

CONCLUSIONS

MDS and ISOMAP which have higher complexity, are typical manifold learning methods. Although LMDS has lower complexity, random landmark point sets often lead to worse results. A fast manifold learning method was proposed which only preserved the similarities or dissimilarities between each point and some points including its nearest neighbors and some far points. Experiments showed that the method could discover the manifold embedded in the high dimensional space and had higher efficiency. Experiments also displayed that the better results would be gained when far points were more than five. Moreover, the initial value also influenced the embedded results. Then, we will check the performance of the algorithm in the real applications.

ACKNOWLEDGMENT

The study was supported by Research Project of Application Foundation and Advanced Technology of Tianjin of China (No. 10JCZDJC16000).

REFERENCES

Belkin, M. and P. Niyogi, 2003. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput., 15: 1373-1396.

De Silva, V. and J.B. Tenenbaum, 2003. Global versus local methods in nonlinear dimensionality reduction. Adv. Neural Inform. Process. Syst., 15: 705-712.

Hinton, G.E. and R.R. Salakhutdinov, 2006. Reducing the dimensionality of data with neural networks. Science, 313: 504-507.

Kruskal, J.B., 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika, 29: 1-27.

Roweis, S.T. and L.K. Saul, 2000. Nonlinear dimensionality reduction by locally linear embedding. Science, 290: 2323-2326.

Tenenbaum, J.B., vin de Silva and J.C. Langford, 2000. A global geometric framework for nonlinear dimensionality reduction. Science, 290: 2319-2323.

Zhang, Z. and H. Zha, 2004. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. SIAM J. Sci. Comput, 26: 313-338.