Algorithm for stochastic Neighbor Embedding: Conjugate Gradient, Newton, and Trust-Region

제홍모^{0} ,남기정, 최승진
포항공과 대학교 컴퓨터 공학과
{ invu71^{0}, lovegod , seungjin }@postech.ac.kr

Algorithm for stochastic Neighbor Embedding: Conjugate Gradient, Newton, and Trust-Region

Hongmo Je^{0} .Kijoeng Nam .Seungjin Choi
Department of Computer Science, Pohang University of Science and Technology

Abstract

Stochastic Neighbor Embedding(SNE) is a probabilistic method of mapping high-dimensional data space into a low-dimensional representation with preserving neighbor identities. Even though SNE shows several useful properties, the gradient-based naive SNE algorithm has a critical limitation that it is very slow to converge. To overcome this limitation, faster optimization methods should be considered by using trust region method we call this method fast TR SNE. Moreover, this paper presents a couple of useful optimization methods(i.e. conjugate gradient method and Newton's method) to embody fast SNE algorithm. We compared above three methods and conclude that TR-SNE is the best algorithm among them considering speed and stability. Finally, we show several visualizing experiments of TR-SNE to confirm its stability by experiments.

1. Introduction

Dimensionality reduction is a fundamental problem in a variety of areas such as machine learning, pattern recognition, exploratory data analysis, data visualization, and so on. Many methods of embedding objects, described by high-dimensional vectors or by pairwise dssimilarities, into a lower-dimensional space, have been extensively studied such as PCA, MDS[2], Laplacian eigenmap [1], Isomap[7], LLE[5], and LLC[6]. Recently *sochastic neighbor embedding* (SNE) was proposed as a probabilistic embedding method in [3]. In contrast to other nonlinear dimensionality reduction methods, SNE is a probabilistic approach that preserves the distribution of neighbor identities. The probabilistic framework on dimensionality reduction makes it easy embedding without any constraints. However, SNE using a steepest descent method (which considers only gradient information) to find optimal solution, it suffers from the slow convergence. To solve this problem we suggest SNE based on trust-region method named TR- SNE, and compare TR-SNE with SNE based on other optimization algorithms such as Conjugate Gradient, and Newton methods. From the results of experiment we conclude TR-SNE is the best algorithm in the way of both speed and stability.

2. Stochastic Neighbor Embedding

Denote by $x_t \in ¡^{D}$ an object described by a D -dimensional vector. The vector $\overset{\imath}{x} \in ¡^{DN}$ is a long vector that is constructed by stacking $\{x_t\}$ in a single column. The image of x_t is denoted by $y_t \in ?^{D}$ $(d \quad D)$ and the vector $\overset{u}{y} \in ¡^{dN}$ is constructed in a similar manner. The original SNE algorithm [3] is described below.

Step 1 **Neighbors Selection** Select neighbors by ε *neighborhoods or* k *nearest neighbors.*

Step 2 **Computing p_{ij} and q_{ij}** Compute the probability, p_{ij}, that x_i would pick x_j found in Step 1 as its neighbor:

$$p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ik}^2)} \quad (1)$$

where d_{ij}^2 are dissimilarities between two objects x_i and x_j in the high-dimensional space and σ_i is a Gaussian kernel width usually set by hand. The dissimilarities are computed by the scaled Euclidean distance

$$d_{ij}^2 = \frac{\left\| x_i - x_j \right\|^2}{2\sigma_i^2} \quad (2)$$

In the low-dimensional space, the *induced* probability q_{ij} (with a fixed variance) that the image y_i pick y_j as its neighbor, is described by

$$q_{ij} = \frac{\exp(-\left\| y_i - y_j \right\|^2)}{\sum_{k \neq i} \exp(-\left\| y_i - y_k \right\|^2)}. \quad (3).$$

Step3 **A Cost Function** The aim of the embedding is to match p_{ij} and q_{ij} as well as possible. This is achieved by minimizing a cost function, sum of Kullback-Leibler divergences between p_{ij} and q_{ij} for each object. The cost function is given by

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (4)$$

Step4 **Embedding through Steepest Descent** The set of images, $\overset{u}{y}$ in the lower-dimensional space, are updated by a gradient-descent method which has the form

$$\vec{y}^{(k+1)} = \vec{y}^{(k)} - \alpha^{(k)} \nabla C^{(k)} \tag{5}$$

where $\alpha^{(k)}$ is a learning rate and the gradient ∇C is given by

$$\nabla C = \left[\left(\frac{\partial C}{\partial y_1} \right)^{T}, K, \left(\frac{\partial C}{\partial y_N} \right)^{T} \right] \tag{6}$$

$$\frac{\partial C}{\partial y_i} = 2 \sum_j (y_i - y_j)(p_{ij} - q_{ij} + p_{ji} - q_{ji}). \tag{7}$$

3. Trust-Region methods

Trust-region methods [4] define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this trust region. In effect, they choose the direction and length of the step simultaneously. If a step is not acceptable, they reduce the size of the region and find a new minimizer. In general, the step direction changes whenever the size of the trust region is altered.

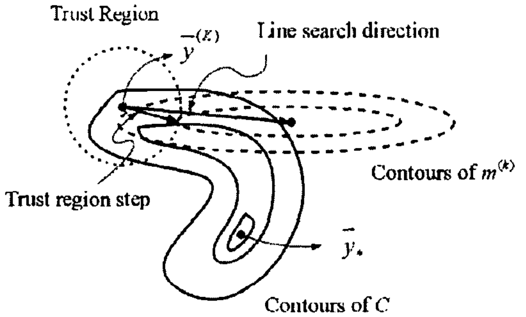

Fig. 1. An illustration of the trust-region method

Fig. 1 illustrates a trust-region approach for the minimization of an objective function C in which the current point lies at one end of a curved valley while the minimizer $\vec{y}_*$ lies at the other end. A quadratic model function $m^{(k)}$, whose elliptical contours are shown as dashed lines, is based on function and derivative information at $\vec{y}^{(k)}$ and possibly also on information accumulated from previous iterations and steps. A line search method based on this model searches along the step to the minimizer of $m^{(k)}$, but this direction allows only a small reduction in C even if an optimal step is taken. A trust-region method, on the other hand, steps to the minimizer of $m^{(k)}$ within the dotted circle, which yields a more significant reduction in C and a better step. And, the step $\vec{p}$ is obtained by solving the following subproblem:

$$\min_{\|\vec{p}\| \le \Delta^{(k)}} m^{(k)}(\vec{p}) = C^{(k)} + \left[\nabla C^{(k)} \right]^{T} \vec{p} + \frac{1}{2} \vec{p} B^{(k)} \vec{p} \tag{8}$$

There are three strategies for finding approximate solution of Eq. (8). The first strategy is the *dogleg* ,the second strategy is the *two-dimensional subspace minimization* ,the third strategy is the *Steighaug's approach.* (See Ch. 4 and 6 in [4] for further details) Trust-region methods guarantee the global convergence, which is stated in the theorem. (See theorem and proof in [4])

4.Conjugate Gradient methods for SNE algorithm

Our interest in the conjugate-gradient method to fast SNE algorithm is twofold. It is one of the most useful techniques for solving large systems of equations (In general, SNE becomes large-scale problem as number of the data points grows.), and it can also be adapted to nonlinear optimization problems (Objective function Eq. (4) is nonlinear equation to be optimized.). We choose Fletcher and Reeves'(Fletcher and C.M.Reeves, 1964) nonlinear conjugate-gradient method to solve SNE (we name it *FR-CG-SNE*) since objective function is not convex quadratic function. It is one of the earliest known techniques for solving large-scale nonlinear optimization problems. The key features of this algorithm are that it requires no matrix storage and is faster than the steepest descent method. Fletcher and Reeves showed that an extension is possible by making two simple changes in Linear Conjugate gradient. To make direction of $\vec{p}^{(k)}$ a descent direction , step length must satisfy the strong Wolfe conditions, or Armijo backtracking conditions. . (See Ch. 3 and 5 in [4] for further details)

5. Newton methods for SNE algorithm

Pure Newton method with unit steps converges rapidly once it approaches a minimizer. This simple algorithm is inadequate for general use, however, since it may fail to converge to a solution from remote starting points. Even if it does converge, its behavior may be erratic in regions where the function is not convex. To obtain global convergence , converging to stationary point, we require the search direction $\vec{p}^{(k)} = -\nabla^2 C\left(y^{(k)}\right)^{-1} \nabla C\left(y^{(k)}\right)$ to be a descent direction, which will be true here if the Hessian $\nabla^2 C\left(y^{(k)}\right)$ is positive definite. However, if the hessian matrix is not positive definite or is close to being singular, $\vec{p}^{(k)}$ may be an ascent direction. To guarantee stability, we modify the Hessian matrix $B^{(k)}$ as $\nabla^2 C\left(y^{(k)}\right) + E(k)$, where $E(k) = 0$ if $\nabla^2 C\left(y^{(k)}\right)$ is sufficiently positive definite; otherwise, $E(k)$ is chosen to ensure that $B^{(k)}$ is sufficiently positive definite. We use hessian matrix $\nabla^2 C\left(y^{(k)}\right)$. In the update rule given by $\vec{y}^{(k+1)} = \vec{y}^{(k)} + \alpha^{(k)} \vec{p}^{(k)}$, step size $\alpha^{(k)}$ is required to be satisfied the Wolfe, Goldstein, or Armijo backtracking conditions.

6. Trust-region methods for SNE algorithm

The trust-region method require a model Hessian matrix B . Since it is possible to compute the exact Hessian $\nabla^2 C(\vec{y})$ in SNE, we replace B in Eq. (8) by $\nabla^2 C(\vec{y})$. The Hessian matrix $\nabla^2 C(\vec{y}) \in ¡^{dN \times dN}$ is computed as

$$\nabla^2 C(\vec{y}) = \begin{bmatrix} \dfrac{\partial^2 C}{\partial y_1 \partial y_1} & L & \dfrac{\partial^2 C}{\partial y_1 \partial y_N} \\ M & O & M \\ \dfrac{\partial^2 C}{\partial y_N \partial y_1} & L & \dfrac{\partial^2 C}{\partial y_N \partial y_N} \end{bmatrix} \tag{9}$$

where

$$\frac{\partial^2 C}{\partial y_i \partial y_j} = -2(p_{ji} - q_{ji} + p_{ij} - q_{ij}) diag(y_{1i}, \mathrm{K}, y_{di}) . \quad (12)$$

Note that p_{ij} is asymmetric, so B is also asymmetric. Therefore we use a symmetric form defined by $\frac{B^T + B}{2}$ instead of B.

We use the Steihaug method which employs the conjugate-gradient (CG) method with a Steihaug's termination test. Compared to the dogleg and the subspace method where solving the linear system of involving B or $(B + \alpha I)$ (for some $\alpha \in$ ¡) is costly, the Steihaug method is a TR-Newton- CG when B is an exact Hessian of the objective function. The Steihaug method has several attractive properties: (1) It requires no matrix factorization, so we can exploit the sparse structure of the Hessian $\nabla^2 C$ without worrying about fill-in during a direct factorization; (2) When the Hessian matrix is positive definite, the Newton-CG method approximates the pure Newton step more and more closely as the solution y_* is approached, so rapid convergence is also possible. When Hessian matrix is not positive definite, we can make it positive definite by adding λI . In this paper we use TR Newton-CG method implemented through the *fminunc* in Matlab Toolbox.

7. Numerical Experimental Results

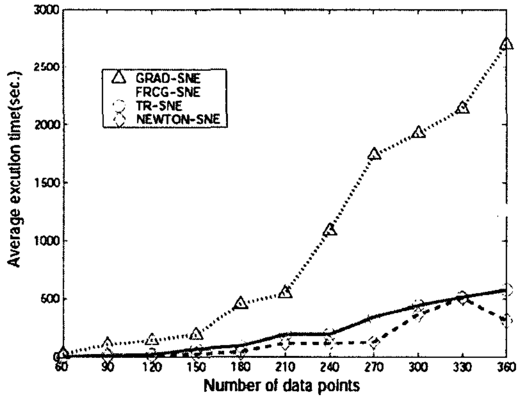

Fig. 2. The comparison of execution times

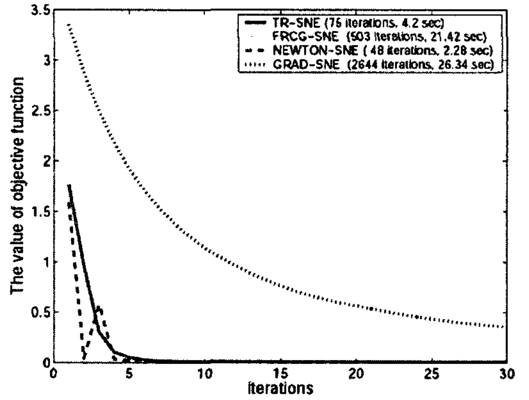

Fig. 3. The comparison of number of iterations

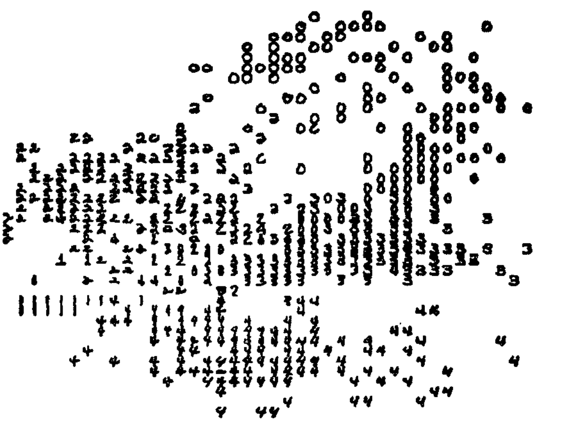

Fig.4. The embedding of USPS data (digit 0~4)

The experiment was carried out with USPS digit data. To avoid very similar pairs like 3 to 5 or 7 to 9, we only consider five classes (0,1,2,3, and 4) are randomly selected. The comparison of the average execution time as the number of data points grows, is drawn in Fig. 2. Also it shows plots of the value of the objective function versus the number of iterations in Fig. 3. We can see the convergence rate of TR-SNE is superior to either the original SNE or CG-FR-SNE. Both the convergence rate and execution time of the modified Newton's SNE seem to be slightly better than those of the TR-SNE, however, modified Newton's SNE tend to be unstable as an initial state. As a result, TR-SNE is about 2--6 time as fast as FRCG-SNE and GRAD-SNE. Fig. 4. presents the embedding result for USPS data using TR-SNE.

8. Conclusion

The original SNE algorithm based on the steepest descent method suffered from its slow convergence. Trust-region methods guarantee the globally linear and locally quadratic convergence rate. We have presented a fast SNE algorithm, TR-SNE, which employed a trust-region method. Moreover, we described CG-FR-SNE, and Modified Newton SNE, comparing TR-SNE with both two method in experiment results we conclude that TR-SNE is superior to these algorithms considering its convergence rate and stability. Also, it is plain that TR-SNE has the high performance and fast convergence compare with SNE based on steepest descent method through several numerical experiments and basic theorems.

REFERENCES

[1] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, pp. 1373-1396, 2003.

[2] T. Cox and M. Cox, *Multidimensional Scaling, 2nd Edition.* Chapman & Hall, 2001.

[3] G. Hinton and S. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems*, vol. 15. MIT Press, 2003.

[4] J. Nocedal and S. J. Wright, *Numerical Optimization.* Springer, 1999.

[5] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323-2326, 2000.

[6] Y. W. Teh and S. Roweis, "Automatic allignment of local representations," in *Advances in Neural Information Processing Systems*, vol. 15. MIT Press, 2003.

[7] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319-2323, 2000.