Cross-Entropy Directed Embedding of Network Data

Takeshi Yamada YAMADA@CSLAB.KECL.NTT.CO.JP
Kazumi Saito SAITO@CSLAB.KECL.NTT.CO.JP
Naonori Ueda UEDA@CSLAB.KECL.NTT.CO.JP
NTT Communication Science Laboratories, 2-4 Hikaridai Seika-cho Soraku-gun Kyoto, 619-0237 JAPAN

Abstract

We present a novel approach to embedding data represented by a network into a low-dimensional Euclidean space. Unlike existing methods, the proposed method attempts to minimize an energy function based on the cross-entropy between desirable and embedded node configurations without directly utilizing pairwise distances between nodes. We also propose a natural criterion to effectively evaluate an embedded network layout in terms of how well node connectivities are preserved. Experimental results show that the proposed method provides better layouts than those produced by some of the well-known embedding methods in terms of the proposed criterion. We believe that our method produces a natural embedding of a large-scale network suitable for analyzing by manual browsing in a two- or three-dimensional Euclidean space.

1. Introduction

In many scientific and engineering domains, complicated relational data structures are frequently represented by networks or, equivalently, graphs. For example, WWW (World Wide Web) sites are often represented by *hyperlink networks*, with pages as nodes and hyperlinks between pages as edges, the interactions between genes, proteins, metabolites and other small molecules in an organism are represented by *gene regulatory networks*, and the relationships between people and other social entities are characterized by *social networks*. This is because network representations often provide important insights for researchers to understand the intrinsic data structure with the help of some mathematical tools such as graph theory, as well as by examining an embedded layout in a low-dimensional Euclidean space.

However, when the size of the network grows large and complicated, it becomes extremely difficult to obtain relevant embedding of networks. Therefore, developing a network embedding algorithm that encourages researchers to make scientific discoveries about underlying knowledge or principles from network data is a quite challenging and important task in the field of machine learning.

One of the most fundamental methods to study a network and intuitively understand its inherent structures is *browsing* over a network layout embedded in a low-dimensional Euclidean space; to examine nodes manually one by one by following their connections and by comparing their connectivities with other nodes. Our goal is to develop an algorithm that embeds a network into a low-dimensional Euclidean space in a manner that is suitable for browsing.

It is difficult to evaluate whether a given network layout is suitable for browsing or not. *Aesthetically pleasing* measures have been used in the literature, but they depend on subjective concepts. In this paper, we start from the following simple and basic principle:

Principle A: connectivity preserving principle
Each node attempts to place its adjacent (i.e., directly connected) nodes relatively more closely than non-adjacent ones.

We propose an algorithm that fulfills this principle by only using connectivity information between nodes as a direct criterion, based on the cross-entropy directed energy function for minimizing. On the other hand, a large body of exiting work assumes pairwise distances between nodes. For example, the spring method proposed by Kamada and Kawai (1989) (hereafter referred to as the KK spring method) first calculates graph-theoretic distances for each pair of nodes. The graph-theoretic distance can be calculated using the shortest path algorithm on a graph, such as the Floyd's algorithm (Floyd, 1962). It then embeds nodes into

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

the low-dimensional Euclidean space such that these graph-theoretic distances are most preserved. In other words, the KK spring method attempts to fulfill the following principle:

Principle B: distance preserving principle
Each pair of nodes attempts to place each other such that their Euclidean distance restores their graph-theoretic distance.

It is clear that the complete fulfillment of principle **B** implies principle **A**, however the converse is not true. We will show that an attempt to fulfill the stronger principle **B** may often fail and end up with a biased embedding that also fails to fulfill principle **A**, and is not suitable for browsing, especially when the network size is large.

When browsing a large and complex network, we often want to focus our attention on some restricted portion of the network by removing nodes and connections that are out of our focus, and by re-optimizing that portion at full scale for more detailed browsing. When this re-optimization does not change the basic structure of the layout drastically, we describe that the ebmedding algorithm has good *clipping stability*.

Removing a certain amount of nodes may change the graph-theoretic distances between the remaining nodes drastically, and therefore, completely change the embedding results produced by the KK spring method that is based on principle **B**. We will see that the proposed approach performs quite well in this sense.

2. Cross-Entropy Approach

2.1. Objective Function

Consider a network (graph) with N nodes (vertices), where its adjacency matrix is denoted by $\mathbf{A} = (a_{i,j})$. In this paper, we focus on undirected graphs, i.e., $a_{i,j} \in \{0, 1\}$, $a_{i,i} = 1$, and $a_{i,j} = a_{j,i}$, but our approach can be easily extended to directed ones. For a given network, our objective is to compute a K-dimensional embedding of the N nodes such that principle **A** stated in Section 1 is fulfilled in terms of the K-dimensional Euclidean norm.

Let $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ be the positions of the N nodes in a K dimensional space. As usual, the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as follows:

$$d_{i,j} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 = \sum_{k=1}^{K} (x_{i,k} - x_{j,k})^2. \qquad (1)$$

We now introduce a monotonic decreasing function $\rho(u) \in [0, 1]$ with respect to $u \geq 0$, where $\rho(0) = 1$ and $\rho(\infty) = 0$. Here $\rho(d_{i,j})$ can be regarded as a continuous similarity function between $\mathbf{x}_i$ and $\mathbf{x}_j$. The basic idea is to consider a modified version of principle **A** using the concept of similarity; each node attempts to place other nodes such that the similarities between the node and its adjacent nodes are higher than those between the node and its non-adjacent nodes. This again can be reformulated in a slightly relaxed form in terms of $a_{i,j}$ and $\rho(d_{i,j})$; each node attempts to place other nodes such that the continuous similarity function $\rho(d_{i,j})$ becomes the closest approximation of the discrete similarity measure $a_{i,j}$ as possible.

To approximate $a_{i,j}$ by $\rho(d_{i,j})$, a cross-entropy function between $a_{i,j}$ and $\rho(d_{i,j})$ is introduced as follows:

$$E_{i,j} = -a_{i,j} \ln \rho(d_{i,j}) - (1 - a_{i,j}) \ln(1 - \rho(d_{i,j})). \quad (2)$$

Equation (2) attains its minimum when $\rho(d_{i,j}) = a_{i,j}$ with respect to $\mathbf{x}_i$ and $\mathbf{x}_j$; i.e., when the discrete similarity measure and the continuous measure become identical. Keeping the symmetric nature of $E_{i,j}$ in mind, we consider the following total energy function to be minimized with respect to $\mathbf{x}_1, \cdots, \mathbf{x}_N$:

$$E = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} E_{i,j}. \qquad (3)$$

When node i is fixed, $a_{i,j}$ can be viewed as a binary class label of node j; i.e., if $a_{i,j} = 1$, then node j has label 1, and, because $a_{i,i} = 1$, this means that node j belongs to the same class with node i. Otherwise, if $a_{i,j} = 0$, then node j has label 0 and belongs to a different class from node i. The problem can be viewed as a set of N binary classification tasks, and Equation (3) as a standard cost function of classification problems with parameters $\mathbf{x}_1, \cdots, \mathbf{x}_N$. Namely, the main characteristic of our approach is to solve the embedding problem the same way we would train a classification function.

In this paper, we use $\rho(u) = \exp(-u/2)$ as the similarity function. Note that our approach is not limited to this particular type of $\rho(u)$. The energy function corresponding to Equation (2) can be reformulated as follows:

$$\begin{aligned} E_{i,j} \;\; = \;\; & a_{i,j} \, \tfrac{1}{2} \|\mathbf{x}_i - \mathbf{x}_j\|^2 \\ & -(1 - a_{i,j}) \ln(1 - \exp(-\tfrac{1}{2} \|\mathbf{x}_i - \mathbf{x}_j\|^2)). \end{aligned} \qquad (4)$$

Finally, we define the following objective function with a regularization (weight-decay) term that controls the size of the resultant embeddings:

$$J = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} E_{i,j} + \frac{\mu}{2} \sum_{i=1}^{N} \|\mathbf{x}_i\|^2, \qquad (5)$$

$N \times N$ adjacency matrix $\mathbf{A} = (a_{i,j})$, embedding dimensionality K, and the convergence precision ϵ are given as inputs.

1. Set $t = 1$, and initialize points $\mathbf{x}_1, \cdots, \mathbf{x}_N$ randomly.
2. Calculate gradient vectors $J^{(1)}_{\mathbf{x}_1}, \cdots, J^{(1)}_{\mathbf{x}_N}$.
3. Select $\mathbf{x}_i$ such that $i = \arg\max_j \{\|J^{(t)}_{\mathbf{x}_j}\|^2\}$.
4. If $\|J^{(t)}_{\mathbf{x}_i}\|^2 < \epsilon$, output $\mathbf{x}_1, \cdots, \mathbf{x}_N$ and terminate.
5. Calculate modification vector $\mathbf{\Delta x}_i$.
6. Update gradient vectors $J^{(t+1)}_{\mathbf{x}_1}, \cdots, J^{(t+1)}_{\mathbf{x}_N}$ from $J^{(t)}_{\mathbf{x}_1}, \cdots, J^{(t)}_{\mathbf{x}_N}$.
7. Update $\mathbf{x}_i$ by $\mathbf{x}_i = \mathbf{x}_i + \mathbf{\Delta x}_i$.
8. Set $t = t + 1$ and go to Step 3.

Figure 1. CE Learning Algorithm for Network Embedding

where μ is a predetermined constant. Hereafter, our method is referred to as the CE (Cross-Entropy) method.

2.2. Learning Algorithm

The basic structure of our CE learning algorithm inherits the fundamental idea of the KK spring algorithm. More specifically, instead of all the N points being moved simultaneously, a point $\mathbf{x}_i$ having the maximum gradient vector norm is selected and is moved using the Newton-Raphson method, while the other points are fixed. This learning scheme, together with the objective function, can be interpreted as on-line learning of a classification problem using cross-entropy with a weight-decay term, which is well-known in the field of artificial neural networks.

The CE algorithm can be summarized as shown in Figure 1. Here, $J_{\mathbf{x}_i}$ is the gradient vector of the objective function J with respect to $\mathbf{x}_i$ calculated as follows:

$$J_{\mathbf{x}_i} = \frac{\partial J}{\partial \mathbf{x}_i} = \sum_{j \neq i} \frac{\partial E_{i,j}}{\partial \mathbf{x}_i} + \mu \mathbf{x}_i, \tag{6}$$

where the derivative of $E_{i,j}$ is

$$\frac{\partial E_{i,j}}{\partial \mathbf{x}_i} = \frac{a_{i,j} - \exp(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2)}{1 - \exp(-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2)}(\mathbf{x}_i - \mathbf{x}_j). \tag{7}$$

Step 2 initially calculates the gradient vectors for all points by using Equation (6). Assuming that dimensionality K is much smaller than N, the computational complexity for this calculation is $O(N^2)$. However, when updating the gradient vectors in Step 6, only the gradient vector for the selected point $\mathbf{x}_i$ must be calculated from scratch, and those for the other points can be efficiently updated by only calculating the differences as follows:

$$\begin{aligned} J^{(t+1)}_{\mathbf{x}_j} &= J^{(t)}_{\mathbf{x}_j} - \frac{\partial E^{(t)}_{i,j}}{\partial \mathbf{x}_j} + \frac{\partial E^{(t+1)}_{i,j}}{\partial \mathbf{x}_j} \\ &= J^{(t)}_{\mathbf{x}_j} + \frac{\partial E^{(t)}_{i,j}}{\partial \mathbf{x}_i} - \frac{\partial E^{(t+1)}_{i,j}}{\partial \mathbf{x}_i}. \end{aligned} \tag{8}$$

This is because of the following equalities:

$$\frac{\partial E^{(t)}_{j,i}}{\partial \mathbf{x}_j} = -\frac{\partial E^{(t)}_{i,j}}{\partial \mathbf{x}_i} \text{ and } \frac{\partial E^{(t+1)}_{j,h}}{\partial \mathbf{x}_j} = \frac{\partial E^{(t)}_{j,h}}{\partial \mathbf{x}_j}, \tag{9}$$

where h denotes an index such that $h \neq i, j$. Thus, Step 6 can be performed in $O(N)$.

In Step 5, according to the Newton-Raphson method, $\mathbf{\Delta x}_i$ is calculated using the Hessian matrix $\mathbf{H}$ as follows:

$$\mathbf{\Delta x}_i = -\mathbf{H}^{-1} J^{(t)}_{\mathbf{x}_i}, \quad \text{where } \mathbf{H} = \frac{\partial^2 J^{(t)}}{\partial \mathbf{x}_i \partial \mathbf{x}'_i}. \tag{10}$$

Here, $\mathbf{x}'$ denotes a transposed vector of $\mathbf{x}$. The KK spring method also uses Equation 10. However, because the Hessian matrix $\mathbf{H}$ is not always positive definite, either in the KK spring method whose energy function is described later in Equation 16 or in the CE method, we cannot guarantee that $J^{(t+1)} < J^{(t)}$ for all t. Therefore, in the CE method, if J increases after Equation 10 is applied, we undo it and instead resort to Equation 11 as follows:

$$\mathbf{\Delta x}_i = \lambda J^{(t)}_{\mathbf{x}_i}, \tag{11}$$

where the step-length λ is chosen so that $J^{(t+1)} < J^{(t)}$. Note that since $J^{(t)}_{\mathbf{x}_i}$ is the gradient direction, J always decreases as long as a sufficiently small λ is chosen and thus, the algorithm is guaranteed to converge. J can be updated in $O(N)$ by only calculating the differences as follows:

$$\begin{aligned} J^{(t+1)} - J^{(t)} = &\textstyle\sum_{j \neq i} (E^{(t+1)}_{i,j} - E^{(t)}_{i,j}) \\ &+ \tfrac{\mu}{2}\{\|\mathbf{x}_i + \mathbf{\Delta x}_i\|^2 - \|\mathbf{x}_i\|^2\}. \end{aligned} \tag{12}$$

In our experiments, the regularization term added in Equation 10 is observed to encourage $\mathbf{H}$ being positive definite as much as possible, and resorting to Equation 11 happens only occasionally. In such cases, each iteration of the above algorithm can be performed in approximately $O(N)$.

3. Evaluations by Experiments

3.1. Embedding Methods

In our experiments, we compared our method with three representative conventional methods: the classical Multidimensional Scaling (MDS) developed by

Torgerson (1958), the spectral clustering method (Ng et al., 2002), and the KK spring method. In the following, we briefly review each method.

Let $\mathbf{G} = (g_{i,j})$ be a pairwise graph-theoretic distance matrix, and $O = (o_{i,j})$ be a matrix defined by $o_{i,j} = g_{i,j}^2$. Let $\mathbf{X} = [\mathbf{x}_1, \cdots \mathbf{x}_N]'$ be the $N \times K$ matrix of coordinates in a K dimensional Euclidean space. The classical MDS has the following objective function to be minimized:

$$J_{\mathrm{CMDS}} = \mathrm{trace}\{(-\frac{1}{2}\mathbf{YOY} - \mathbf{XX}')^2\}. \quad (13)$$

Here, $\mathbf{Y}$ denotes the N-dimensional Young-Householder transformation matrix.

Let $\mathbf{B} = (b_{i,j})$ be an affinity matrix defined by $b_{i,j} = \exp(-g_{i,j}/2)$ if $i \neq j$ and $b_{i,i} = 0$. The spectral clustering method has the following objective function to be minimized:

$$J_{\mathrm{SC}} = \mathrm{trace}\{(\mathbf{D}^{-1/2}\mathbf{BD}^{-1/2} - \mathbf{XX}')^2\}. \quad (14)$$

Here, $\mathbf{D}$ denotes a diagonal matrix whose (i,i)-element is the sum of the i-th row of $\mathbf{B}$. In this method, the final embedded points are obtained by re-normalizing each row of $\mathbf{X}$ to have unit length, i.e.,

$$\widehat{x}_{i,j} = \frac{x_{i,j}}{\sqrt{\sum_{j=1}^{N} x_{i,j}}}. \quad (15)$$

The KK spring method has the following objective function to be minimized:

$$J_{\mathrm{KK}} = \frac{1}{2}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{(g_{i,j} - \|\mathbf{x}_i - \mathbf{x}_j\|)^2}{g_{i,j}^2}. \quad (16)$$

3.2. Experimental Data

Three different types of networks assembled from real data are used to evaluate the proposed method. The first network data, denoted as **E.Coli**, is biology originated. **E.Coli** is the gene regulatory network of the bacterium *Escherichia Coli* as described by Shen-Orr et al. (2002). The second data, denoted as **NIPS**, is human relation data generated by assembling a co-authorship relations in the conference papers appeared in NIPS (Neural Information Processing Systems) volumes 1 to 12, obtained from the web site of Roweis (2002), in which two persons who have at least one joint paper are directly linked. The third data, denoted as **NTT**, is a WWW hyperlink network. **NTT** is generated by collecting all WWW pages that are located in NTT (Nippon Telegraph and Telephone Corporation) domain and have "`www.ntt.co.jp`" in common in their URL (Universal Resource Locator).

Table 1. Summary statistics of networks.

NAME	N	L	$\bar{L}$	L°	$\bar{G}$	G°	C	N_2
E.Coli	328	456	2.78	72	4.83	13	29	12
NIPS	1061	2080	3.92	45	7.23	17	235	37
NTT	3870	9337	4.83	279	6.41	17	76	54

In our experiments, we first transform these networks into undirected ones, extract the maximally connected components, and then apply embedding methods to them. If we need to embed the whole network, then we can treat each connected component separately.

Table 1 shows the statistics of the extracted connected networks. In the table, N denotes the number of nodes, and L, $\bar{L}$ and L° denote total, average and maximum number of links respectively. Let L_i be the number of links connected with i. They are then calculated as follows:

$$L = \frac{1}{2}\sum_{i=1}^{N} L_i,\ \bar{L} = \frac{1}{N}\sum_{i=1}^{N} L_i \text{ and } L^\circ = \max_i\{L_i\}. \quad (17)$$

$\bar{G}$ and G° denote the average and maximum value of pairwise graph-theoretic distances in each graph respectively. Let $g_{i,j}$ be the graph-theoretic distances between nodes i and j. They are then calculated as follows:

$$\bar{G} = \frac{2}{N(N-1)}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} g_{i,j} \text{ and } G^\circ = \max_{i,j}\{g_{i,j}\}. \quad (18)$$

Here, we emphasize that these networks are all sparse in terms of the adjacency matrices, but have different statistics to some degree. C denotes total number of connected components in the original networks and N_2 denotes the number of nodes in the second largest component, which is much smaller than N.

3.3. Evaluation Measure

In Section 1, we claimed that network embedding should fulfill principle **A** and proposed an embedding algorithm based on cross-entropy and local connectivities in Section 2. In this section, we attempt to evaluate the embedded results in a strictly quantitative fashion.

Assume we have an embedding of a network with N nodes, with $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ as their corresponding embedded positions in a K-dimensional space. Consider a K-dimensional ball $B_i(r_i)$ with its center $\mathbf{x}_i$ and radius r_i. From principle **A**, it is expected that in an ideal

embedding, each node i can maintain $B_i(r_i)$ with an appropriately chosen radius r_i such that it includes all the points corresponding to the adjacent nodes (i.e., nodes directly connected with i) and excludes all the non-adjacent ones. In reality, however, especially when K is small, there may not exist such r_i for each i. However, it may be still possible to find an *optimal* r_i in terms of some relevant criterion.

In a sparse network, where there are many more non-adjacent nodes than adjacent ones, the standard accuracy measure that counts the number of correctly included points and correctly excluded points is not appropriate to determine the optimal radius. This is because high accuracy can be achieved even if it excludes all the points regardless of their connectivities. Instead, we adopt the idea of the F-measure that is widely used in the field of information retrieval.

The F-measure is defined as the weighted harmonic average of *precision* and *recall*. Let $\#X$ be the number of elements in set X. The precision $P_i(r_i)$ for the i-th ball $B_i(r_i)$ corresponding to $\mathbf{x}_i$ and r_i is defined as follows:

$$P_i(r_i) = \frac{\#\{j|\mathbf{x}_j \in B_i(r_i), a_{i,j} = 1, j \neq i\}}{\#\{j|\mathbf{x}_j \in B_i(r_i), j \neq i\}}, \qquad (19)$$

while the recall $R_i(r_i)$ is defined as:

$$R_i(r_i) = \frac{\#\{j|\mathbf{x}_j \in B_i(r_i), a_{i,j} = 1, j \neq i\}}{\#\{j|a_{ij} = 1, j \neq i\}}. \qquad (20)$$

Roughly speaking, high precision favors a small r_i and high recall favors a large r_i; the optimal r_i should be found in between. We choose the optimal radius $\widehat{r}_i$ for each i that maximizes the following F-measure with weight α. ($\alpha = 1/2$ is used throughout our experiments.)

$$F_i(r_i) = 1/\left\{\alpha\frac{1}{P_i(r_i)} + (1-\alpha)\frac{1}{R_i(r_i)}\right\}. \qquad (21)$$

The proposed measure, denoted as the connectivity F-measure, to evaluate an embedded network layout is defined as the average over all N points as follows:

$$F = \sum_{i=1}^{N} \frac{F_i(\widehat{r}_i)}{N}. \qquad (22)$$

It may be arguable to straightforwardly apply our criterion to the embeddings produced by the other methods based on principle **B**, because their objective functions do not directly incorporate this F-measure. However, the complete fulfillment of principle **B** implies principle **A**. In fact, when principle **B** is fulfilled, $F = 1$ is achieved by setting the optimal radius $\widehat{r}_i = 1$ for each i. Therefore, our criterion can be thought of as a reasonable measure for evaluating any embedding method.

3.4. Comparisons Using the Connectivity F-measure

We applied our method together with the three conventional methods, the classical MDS, the KK spring method, and the spectral clustering method, to the three different kinds of networks mentioned above, and obtained embedding results in K-dimensional spaces. We then quantitatively evaluated these results according to the proposed connectivity F-measure. In the experiments, we changed K from 2 to 25 for **E.Coli** and from 2 to 9 for **NIPS** and **NTT** respectively.

Figure 2 summerizes the experimental results. The axis of ordinates shows the values of the connectivity F-measure and the axis of abscissas shows the dimensionalities of the embedded space. For the CE and the KK spring methods, the results of five trials with different random initializations are plotted to see if these methods suffer from the local optimality problem. One can see that variances are very small for all networks, and therefore the local optimality problem is almost negligible in our experiments. The performances of all methods are monotonically improved as the dimensionality K, and thus the degree of freedom, increases.

As expected, the performance of the CE method is better than those of the other methods, especially in the lower dimensions, where embeddings are more difficult. In particular, the experiments using **NTT**, which is the largest in size and the most complicated, most impressively demonstrate that the CE method significantly outperforms the others. Figure 2 only shows the F-measure, but more detailed analysis reveals that, in low dimensions, all the methods show high recall values, more or less 90%, but poor precision values, except for the CE method.

3.5. Two Dimensional Visualizations

Each of the pictures on the left of Figure 3 shows a two-dimensional embedding result obtained by the classical MDS, the KK spring method, and the CE method for **NTT**, the WWW network data. The results of the spectral clustering method are not included here because it embeds on the surface of a sphere, as described in Equation 15.

As discussed earlier, it would be difficult to evaluate these results in the sense of the *aesthetically pleasing*

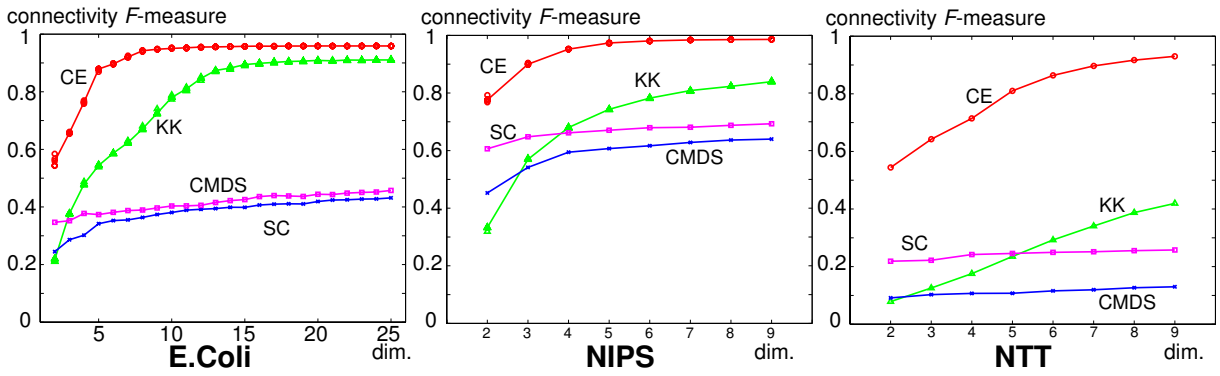

Figure 2. Experimental comparisons of the connectivity F-measure between the classical MDS (labeled CMDS), the spectral clustering method (labeled SC), the KK spring method (labeled KK) and the proposed CE method (labeld CE) using the **E.Coli**, **NIPS** and **NTT** networks.

measure. Nevertheless, some characteristic features of each method can be observed below. The result of the classical MDS shown in Figure 3(a) is problematic for visualization: many nodes are collapsed to a single point, which is not desirable for a browsing purpose. This seems to be a limitation of the linear projection methods.

In contrast, the result of KK spring method shown in Figure 3(b) and the result of the proposed CE method shown in Figure 3(c) do not suffer from the node collapse problem. Looking at Figures 3(b) and 3(c) carefully, one can observe that in the former picture, many nodes tend to be arranged densely in semi-circles, which is characterized as a *dandelion* effect (Buja et al., in press), while in the latter picture, nodes are laid more uniformly in a radial manner. The CE method appears to exploit the space more efficiently than the KK spring method. This agrees with the quantitative evaluation results using the connectivity F-measure, and also agrees with the fact that the KK spring method tries to preserve the pairwise graph-theoretic distances that take only discrete values. Besides, it appears that it is harder for the former to follow links, especially in a relatively congested area, than for the latter.

As mentioned earlier, we may want to look into some portions of the network in detail. For this purpose, the sub network shown in the dotted circle in the left of each picture in Figure 3 is *clipped out* by removing the nodes and connections outside this circle, and is re-embedded. Note that each clipped region corresponds to the same sub network. The re-embedded results are shown on the right of Figure 3. It can be seen that the embedded network layouts before and after the clipping are quite different from each other in the case of the classical MDS and KK spring methods, although the layouts in the dotted circles shown on the left and right of each picture correspond to the same sub network. In contrast, the CE method is much more stable with the clipping than the classical MDS and the KK spring method, and thus has better clipping stability. We believe that the clipping stability is an important property for browsing nodes and discovering knowledge from given networks.

4. Related Works and Discussion

Several other embedding methods based on the eigenvector analysis other than the classical MDS and the spectral clustering methods exist, including Isomap (Tenenbaum et al., 2000) and LLE (Roweis & Saul, 2000). These methods assume high dimensional coordinates or at least pairwise distances between nodes. In contrast, our method only assumes the adjacency matrix. Stochastic embedding, recently proposed by Hinton and Roweis (in press), also assumes pairwise distances.

Eades (1984) first proposed a spring-directed graph embedding algorithm. In his algorithm, each pair of adjacent nodes is linked by a spring of length one, and each non-adjacent pair by a spring of infinite length. He claims that the linear spring that obeys Hooke's law is too strong and that a logarithmic version should be used. Eades' algorithm was the basis for the subsequent spring methods. The KK spring method has extended Eades' idea and proposed an algorithm in which each pair of nodes is linked by a spring of length equal to the graph-theoretic distance between those nodes. This method is much more popular than its ancestor and is widely refered to as *the* spring method.

The well-known state-of-the-art graph drawing program called "NEATO" in the graph-viz software package (North, 1992) includes an almost literal implementation of the KK spring method.

Many of the existing algorithms, including the classical MDS and KK spring methods utilize the pairwise distances between nodes. Because the distance matrix is not sparse (even if the network is sparse), they need to deal with a huge non-sparse matrix, especially for a large network. In such cases, it is possible to modify the matrix sparse by thresholding large distances to infinity (or, equivalently, small affinities to zero), but still necessary to find an appropriate threshold. Our CE approach only deals with the sparse adjacency matrix when the network is sparse, and needs no such thresholding.

5. Directions for Future Research

Although we have been encouraged by our results to date, there remain a number of directions in which we must extend our approach before it can become a useful tool for scientific discovery. One promising direction might be to combine our method with existing discovery systems such as the IPM (Inductive Process Modeler) proposed by Langley et al. (2002). For instance, we can regard a process model as relational data, with processes as nodes and shared variables between them as links. Our method can be directly incorporated as an interface to support the browsing of large scale process models obtained by the IPM using empirically observed data.

The WWW is a complex system that changes over time. It would be more difficult than any other dynamic system, but still highly expected, to understand its inherent structures in scientific terms and to reveal regularities in its complicated behavior. Thus, another direction would be to develop a discovery system for the WWW. To this end, we need to extend our method by incorporating a number of important characteristics of the Web networks, such as network motifs (Shen-Orr et al., 2002). As the first step, we are planing to evaluate our method by using a wider variety of the Web networks.

In this paper, we have newly proposed the connectivity F-measure based on the connectivity preserving principle. However, it seems quite difficult to develop an algorithm for obtaining results by direct optimization on this measure due to its non-smoothness. On the other hand, our method based on the cross-entropy energy function works reasonably well on this measure, as shown in our experiments. From these results, we believe that the cross-entropy approach is suitable for preserving the principle. However, we need to perform further experiments to confirm this claim.

References

Buja, A., Swayne, D. F., Littman, M., Dean, N., & Hofmann, H. (in press). Xgvis: Interactive data visualization with multidimensional scaling. *Journal of Computational and Graphical Statistics*.

Eades, P. (1984). A heuristic for graph drawing. *Congressus Numerantium*, *42*, 149–160.

Floyd, R. W. (1962). Algorithm 97: shortest path. *Communications of the ACM*, *5*, 345.

Hinton, G., & Roweis, S. T. (in press). Stochastic neighbor embedding. *Advances in Neural Information Processing Systems 15*. Cambridge, MA.: MIT Press.

Kamada, T., & Kawai, S. (1989). An algorithm for drawing general undirected graph. *Information Processing Letters*, *32*, 7–15.

Langley, P., Sanchez, J., Todorovski, L., & Dzeroski, S. (2002). Inducing process models from continuous data. *Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 347–354). San Francisco: Morgan Kaufmann.

Ng, A., Jordan, M., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems 14*. Cambridge, MA.: MIT Press.

North, S. C. (1992). *NEATO user's guide*. AT&A Bell Laboratories.

Roweis, S. T. (2002). Data for MATLAB hackers: NIPS conference papers Vols 0–12. http://www.cs.toronto.edu/~roweis/data.html.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Shen-Orr, S. S., Milo, R., Mangan, S., & Alon, U. (2002). Network motifs in the transcriptional regulation network of escherichia coli. *Nature Genetics*, *31(1)*, 64–68.

Tenenbaum, J. B., de Silva, V., & Langford, J. C. (2000). global geometric framework for nonlinear dimensionality reduction. *Science*, *290*, 2319–2323.

Torgerson, W. S. (1958). *Theory and methods of scaling*. New York: Wiley.

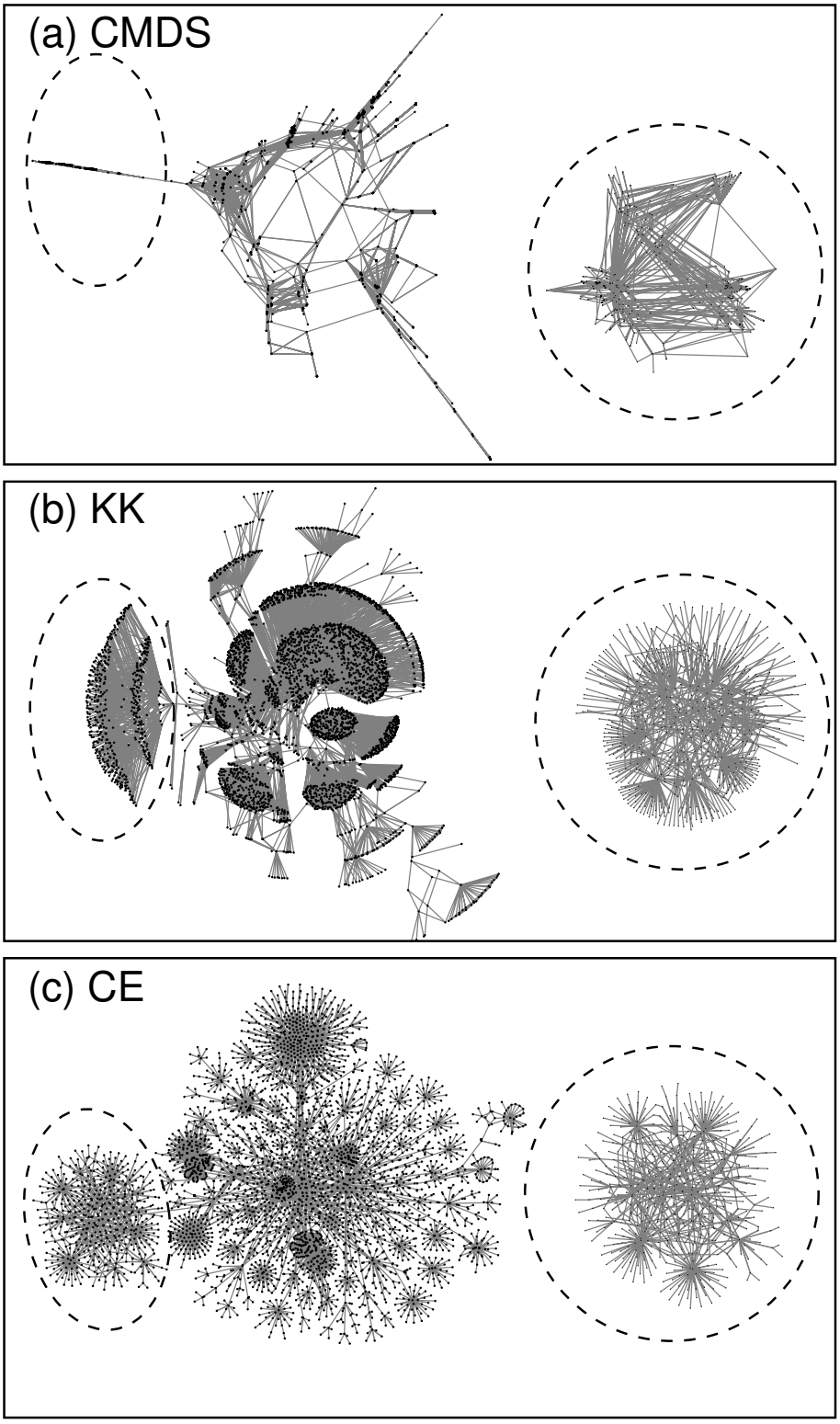

Figure 3. 2-dimensional layouts of the **NTT** Web network data produced by the classical MDS (a), the KK spring method (b) and our CE method (c). Each picture on the left shows the whole network and the sub network in the dotted circle is re-embedded and shown on the right.