Speeding up Bipartite Graph Visualization Method

Takayasu Fushimi[1], Yamato Kubota[2], Kazumi Saito[1,2], Masahiro Kimura[3], Kouzou Ohara[4], and Hiroshi Motoda[5]

[1] Graduate School of Management and Information of Innovation, University of Shizuoka
52-1 Yada, Suruga-ku, Shizuoka 422-8526, Japan
{j11507,k-saito}@u-shizuoka-ken.ac.jp
[2] School of Management and Information, University of Shizuoka
52-1 Yada, Suruga-ku, Shizuoka 422-8526, Japan
{b08038, k-saito}@u-shizuoka-ken.ac.jp
[3] Department of Electronics and Informatics, Ryukoku University
Otsu, Shiga 520-2194, Japan
kimura@rins.ryukoku.ac.jp
[4] Department of Integrated Information Technology, Aoyama Gakuin University
Kanagawa 229-8558, Japan
ohara@it.aoyama.ac.jp
[5] Institute of Scientific and Industrial Research, Osaka University
Osaka 567-0047, Japan
motoda@ar.sanken.osaka-u.ac.jp

Abstract. We address the problem of visualizing structure of bipartite graphs such as relations between pairs of objects and their multi-labeled categories. For this task, the existing spherical embedding method, as well as the other standard graph embedding methods, can be used. However, these existing methods either produce poor visualization results or require extremely large computation time to obtain the final results. In order to overcome these shortcomings, we propose a new spherical embedding method based on a power iteration, which additionally performs two operations on the position vectors: double-centering and normalizing operations. Moreover, we theoretically prove that the proposed method always converges. In our experiments using bipartite graphs constructed from the Japanese sites of Yahoo!Movies and Yahoo!Answers, we show that the proposed method works much faster than these existing methods and still the visualization results are comparable to the best available so far.

1 Introduction

Visualization by embedding graphs into a low dimensional Euclidean space plays an important role to intuitively understand the essential structure of graphs (networks). To this end, various graph embedding methods have been proposed in the past that include multi-dimensional scaling [6], spectral embedding [1], spring force embedding [2], cross-entropy embedding [7]. Each method has its own advantages and disadvantages.

In this paper, we address the problem of visualizing structure of bipartite graphs such as relations between pairs of objects and their multi-labeled categories. More specifically, relations of this kind include pairs of movies and their associated genres,

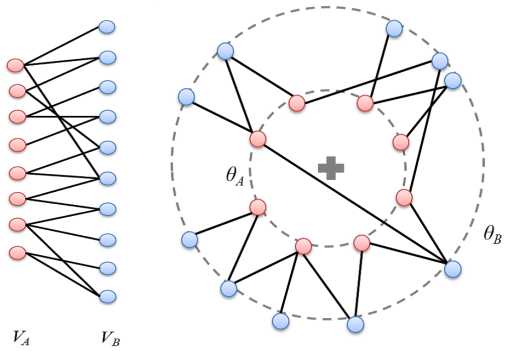
pairs of persons and their interested genres, pairs of researchers and their coauthoring papers, pairs of words and their appearing documents, and many more. Clearly, we can straightforwardly apply any one of the above-mentioned embedding methods for the visualization. However, we note that these standard methods have an intrinsic limitation because they cannot make much use of the essential structure of bipartite graphs. Indeed, the existing spherical embedding method has been proposed for the purpose of visualizing bipartite graphs [5]. In this method, the position vectors are embedded on two concentric spheres (circles) with different radii. We consider that such a spherical embedding can be a natural representation for bipartite graphs. However, the biggest problem with the existing method is that it often requires an extremely large computation time to obtain the final visualization results.

In this paper, to overcome these shortcomings, we propose a new spherical embedding method based on a power iteration, which adopts two operations to iteratively adjust the positioning vectors: double-centering and normalizing operations. We further show theoretically that the convergence of the proposed algorithm is always guaranteed. In our experiments that use bipartite graphs constructed from the Japanese sites of Yahoo!Movies and Yahoo!Answers, we show that the proposed method works much faster than these existing methods, and yet the visualization results are comparable to the best available so far.

2 Problem Framework

We describe the problem framework of embedding the bipartite graph $G = (V, E)$ into a K-dimensional Euclidean space, where $V = V_A \cup V_B$, $V_A \cap V_B = \emptyset$, and $E \subset V_A \times V_B$. For the sake of technical convenience, we identify each set of the nodes, V_A and V_B, by two different series of positive integers, i.e., $V_A = \{1, \cdots, m, \cdots, M\}$ and $V_B = \{1, \cdots, n, \cdots, N\}$. Here M and N are the numbers of the nodes in V_A and V_B , i.e., $|V_A| = M$ and $|V_B| = N$, respectively. Then, we can define the $M \times N$ adjacency matrix $\mathbf{A} = \{a_{m,n}\}$ by setting $a_{m,n} = 1$ if $(m, n) \in E$; $a_{m,n} = 0$ otherwise. We denote the K-dimensional embedding position vectors by $\mathbf{x}_m$ for the node $m \in V_A$ and $\mathbf{y}_n$ for the node $n \in V_B$. Then we can construct $M \times K$ and $N \times K$ matrices consisting of these position vectors, i.e., $\mathbf{X} = (\mathbf{x}_1, \cdots \mathbf{x}_M)^T$ and $\mathbf{Y} = (\mathbf{y}_1, \cdots \mathbf{y}_N)^T$. Here $\mathbf{X}^T$ stands for the transposition of $\mathbf{X}$.

According to the work on the existing spherical embedding method [5], we explain the framework of spherical embedding of bipartite graph. In Fig. 1, we show an example in a two-dimensional Euclidean space, i.e., unlike the standard visualization scheme shown in Fig. 1a, we consider locating the position vectors on two concentric spheres (circles) as shown in Fig. 1b. We believe that this kind of spherical embedding is natural to represent bipartite graphs, and its usefulness has been reported [5]. Hereafter, we assume that nodes in subset V_A are located on the inner circle θ_A with radius $r_A = 1$, while nodes in V_B are located on the outer circle θ_B with radius $r_B = 2$. Note that $\|\mathbf{x}_m\| = 1$, $\|\mathbf{y}_n\| = 2$. Then, our aim is to locate the position vectors of the nodes having similar connection patterns closely to each other.



(a) Bipartite Graph (b) Spherical Embedding

Fig. 1: Spherical Embedding for Bipartite Graph

3 Proposed Method

3.1 Proposed Algorithm

The new spherical embedding method is based on a power iteration. It has two operations on the positioning vectors which we call double-centering operation and normalizing operation. In order to describe our algorithm, we need to introduce the centering matrices and normalizing operations. The centering (Young-Householder transformation) matrices are defined as $\mathbf{H}_M = \mathbf{I}_M - \frac{1}{M}\mathbf{1}_M\mathbf{1}_M^T$, $\mathbf{H}_N = \mathbf{I}_N - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T$ where $\mathbf{I}_M$ and $\mathbf{I}_N$ stands for $M \times M$ and $N \times N$ identity matrices, respectively, and $\mathbf{1}_M$ and $\mathbf{1}_N$ are M- and N-dimensional vectors whose elements are all one. Clearly, the mean vector of the resulting position vectors becomes $\mathbf{0}$ by the operations $\mathbf{H}_M\mathbf{X}$ and $\mathbf{H}_N\mathbf{Y}$. On the other hand, the normalizing operations are defined as $\mathbf{\Lambda}_M(\mathbf{X}) = r_A\mathrm{diag}(\mathbf{X}\mathbf{X}^T)^{-1/2}\mathbf{X}$, $\mathbf{\Lambda}_N(\mathbf{Y}) = r_B\mathrm{diag}(\mathbf{Y}\mathbf{Y}^T)^{-1/2}\mathbf{Y}$, where $\mathrm{diag}(\cdot)$ is an operation to set all the non-diagonal elements to zero, i.e., $\mathrm{diag}(\mathbf{X}\mathbf{X}^T)$ is a diagonal matrix whose m-th element is $\mathbf{x}_m^T\mathbf{x}_m$.

Intuitively, the basic procedure of our proposed algorithm is that the position vector $\mathbf{x}_m$ is repeatedly moved to the position calculated by adding the position vectors $\{\mathbf{y}_n\}$ that are connected to $\mathbf{x}_m$. Of course, we need to perform a normalizing operation so as to satisfy the spherical constraints. Below we describe our proposed algorithm.

1. Initialize the matrix $\mathbf{X}$ and $\mathbf{Y}$.
2. Update the matrix $\mathbf{X} \leftarrow \mathbf{\Lambda}_M(\mathbf{H}_M\mathbf{A}\mathbf{H}_N\mathbf{Y})$.
3. Update the matrix $\mathbf{Y} \leftarrow \mathbf{\Lambda}_N(\mathbf{H}_N\mathbf{A}^T\mathbf{H}_M\mathbf{X})$.
4. Terminate if the changes for the position vectors $\mathbf{X}$ and $\mathbf{Y}$ are small.
5. Return to the step 2.

As the basic framework, our proposed algorithm employs a power iteration, just like the HITS algorithm [3], which utilizes $\mathbf{A}$ and $\mathbf{A}^T$, does. However, the main differences are

use of the double-centering operations by $\mathbf{H}_M$ and $\mathbf{H}_N$ and the normalizing operations by $\mathbf{\Lambda}_M(\cdot)$ and $\mathbf{\Lambda}_N(\cdot)$. Here note that the double-centering operation is also employed in the standard multidimensional scaling method [6].

Now we briefly mention the computational complexity of our algorithm. Clearly, the main computational complexity of one-iteration comes from the multiplication by the matrix $\mathbf{A}$ (or $\mathbf{A}^T$) which is the most intensive part and is proportional to the number of links in the bipartite graph. Thus, the proposed algorithm is expected to work much faster especially for a sparse bipartite graph, compared with the existing spherical embedding algorithm that require a nonlinear optimization just like a spring force embedding [2] does. In fact, it has been well known that the PageRank algorithm based on a power iteration works very fast for a large and sparse network [4].

3.2 Convergence Proof

We prove the convergence property of the algorithm. To do this, we first introduce the double-centered matrix $\mathbf{B} = \{b_{m,n}\}$ that is calculated from the adjacency matrix $\mathbf{A}$ i.e., $\mathbf{B} = \mathbf{H}_M\mathbf{A}\mathbf{H}_N$. Then, by using the matrix $\mathbf{B}$, we can consider the following objective function with respect to the position vectors $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_M)^T$ and $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_N)^T$.

$$J(\mathbf{X}, \mathbf{Y}) = \sum_{m=1}^{M}\sum_{n=1}^{N} b_{m,n}\frac{\mathbf{x}_m^T}{r_A}\frac{\mathbf{y}_n}{r_B} + \frac{1}{2}\sum_{m=1}^{M}\lambda_m(r_A^2 - \mathbf{x}_m^T\mathbf{x}_m) + \frac{1}{2}\sum_{n=1}^{N}\mu_n(r_B^2 - \mathbf{y}_n^T\mathbf{y}_n), \quad (1)$$

where $\{\lambda_m \mid m = 1, \cdots, M\}$ and $\{\mu_n \mid n = 1, \cdots, N\}$ correspond to Lagrange multipliers for the spherical constraints, i.e., $\mathbf{x}_m^T\mathbf{x}_m = r_A^2$ and $\mathbf{y}_n^T\mathbf{y}_n = r_B^2$ for $1 \le m \le M$ and $1 \le n \le N$.

Now we consider maximizing $J(\mathbf{X}, \mathbf{Y})$ defined in Equation (1) by use of a coordinate strategy. Note that maximizing $J(\mathbf{X}, \mathbf{Y})$ pushes the pairs $\mathbf{x}_m$ and $\mathbf{y}_n$ to the same direction if they are connected and pushes them to the opposite direction if they are unconnected, and realizes the intended visualization. We repeat the following two steps: maximizing $J(\mathbf{X}, \mathbf{Y})$ with respect to $\mathbf{X}$ by fixing the matrix $\mathbf{Y}$ first, and maximizing $J(\mathbf{X}, \mathbf{Y})$ with respect to $\mathbf{Y}$ by fixing the matrix $\mathbf{X}$ next. If the maximization of these steps are achieved by the above algorithm's step 2 and 3, respectively, we can guarantee the convergence of our proposed algorithm.

In order to confirm these facts, we consider the following gradient vector of the objective function $J(\mathbf{X}, \mathbf{Y})$ with respect to $\mathbf{x}_m$.

$$\frac{\partial J(\mathbf{X}, \mathbf{Y})}{\partial \mathbf{x}_m} = \frac{1}{r_A r_B}\sum_{n=1}^{N} b_{m,n}\mathbf{y}_n - \lambda_m\mathbf{x}_m. \quad (2)$$

Thus, for a fixed matrix $\mathbf{Y}$, we obtain the optimal position vector $\mathbf{x}_m$ which maximizes the objective function $J(\mathbf{X}, \mathbf{Y})$ as $\mathbf{x}_m = \frac{r_A}{\|\tilde{\mathbf{x}}_m\|}\tilde{\mathbf{x}}_m$, where $\tilde{\mathbf{x}}_m = \sum_{n=1}^{N} b_{m,n}\mathbf{y}_n$. Here note that the optimal vector $\mathbf{x}_m$ is calculated by using the matrix $\mathbf{Y}$ only. Thus, for $m = 1, \cdots, M$, by using the normalizing operation $\mathbf{\Lambda}_M(\cdot)$ whose diagonal elements become $r_A/\|\tilde{\mathbf{x}}_1\|, \cdots, r_A/\|\tilde{\mathbf{x}}_M\|$, we can obtain the solution in the matrix representation, i.e.,

$$\mathbf{X} = \mathbf{\Lambda}_M(\mathbf{B}\mathbf{Y}) = \mathbf{\Lambda}_M(\mathbf{H}_M\mathbf{A}\mathbf{H}_N\mathbf{Y}). \quad (3)$$

1 Science Fiction/Fantasy	red circle	9 Documentary	olive triangle−right
2 Action/Adventure	black square	10 Drama	lime cross
3 Animation	green diamond	11 Family	darkgold plus
4 Comedy	blue star	12 Horror	darkcyan asterisk
5 Suspense	maroon hexagon	13 Musical	magenta circle
6 Teen	orange triangle−up	14 Romance	cyan square
7 Western	purple triangle−down	15 Special Effects	yellow diamond
8 War	navy triangle−left	16 Others	gray star

Fig. 2: category names in Japanese Yahoo!Movies site

Recall that Equation (3) performs centering the matrix $\mathbf{Y}$ by the matrix $\mathbf{H}_N$, multiplies the adjacency matrix $\mathbf{A}$, performs re-centering the matrix by multiplying the matrix $\mathbf{H}_M$, and normalizes so as to guarantee spherical constraints. By this formula, we can obtain the optimal solution of position vectors $\mathbf{X}$ by fixing the matrix $\mathbf{Y}$.

Similarly, we can also obtain the following optimal solution of position vector $\mathbf{y}_n$ by fixing the matrix $\mathbf{X}$ as $\mathbf{y}_n = \frac{r_B}{\|\tilde{\mathbf{y}}_n\|}\tilde{\mathbf{y}}_n$, where $\tilde{\mathbf{y}}_n = \sum_{m=1}^{M} b_{m,n}\mathbf{x}_m$. Thus, for $n = 1, \cdots, N$, by using the normalizing operation $\mathbf{\Lambda}_N(\cdot)$ whose diagonal elements become $r_B/\|\tilde{\mathbf{y}}_1\|, \cdots, r_B/\|\tilde{\mathbf{y}}_N\|$, we can obtain the solution in the matrix representation, i.e.,

$$\mathbf{Y} = \mathbf{\Lambda}_N(\mathbf{B}^T\mathbf{X}) = \mathbf{\Lambda}_N(\mathbf{H}_N\mathbf{A}^T\mathbf{H}_M\mathbf{X}). \tag{4}$$

Therefore, since the finite objective function $J(\mathbf{X}, \mathbf{Y})$ defined in Equation (1) has the analytical optimal solution under the condition that either $\mathbf{X}$ or $\mathbf{Y}$ is fixed, and is always maximized by performing the step 2 and 3 of the algorithm, we can guarantee that the algorithm always converges.

4 Evaluation by Experiments

4.1 Network Data

We constructed the bipartite graphs from the Japanese sites of Yahoo!Movies and Yahoo!Answers, and experimentally evaluated the proposed method by comparing it with the existing embedding methods in terms of both the efficiency of the algorithms and ease of interpretability of the visualization results.

We regard the movies as nodes in V_B, and their genres as nodes in V_A for the Japanese Yahoo!Movies site [2]. Note that each movie is associated with more than or equal to one genre. In Fig. 2, we show their genre names used in our experiments, and for our visual analyses purpose, we assign an individual marker with a different color to each genre as shown in this figure. In order to evaluate our proposed method by using a set of different bipartite graphs, we classify these movies into 7 groups according to their release dates(1950-59, 1960-69, 1970-79, 1980-89, 1990-99, 2000-04 and 2005-09). Here the number of genres is $|V_A| = 16$ for all the periods, the numbers of movies $|V_B|$ are 594, 1079, 1314, 1805, 2659, 2948 and 3264, and the numbers of links $|E|$ are 899, 1617, 2071, 2994, 4424, 6057 and 6564 for each period.

[2] http://movies.yahoo.co.jp/

We regard the users who answered questions as nodes in V_B, and the genres of these questions as nodes in V_A for the Japanese Yahoo!Answers site [3]. Note that although each question belongs to only one genre, the same user frequently answers several questions belonging to a wide variety of genres. Thus we can obtain bipartite graphs between the pairs of the users and the genres they answered. In our experiments, we utilized a set of data from April, 2004, to October, 2005. Again, in order to evaluate our proposed method by using a set of different bipartite graphs, we classify these questions into 6 groups according to their submission dates(2004-2nd, 3rd, 4th, 2005-1st, 2nd and 3rd). Here the number of genres is $|V_A| = 10$ for all the periods, the numbers of users $|V_B|$ are 11871, 27446, 35907, 39451, 42884 and 46834, and the numbers of links $|E|$ are 30849, 80664, 96926, 95714, 102086 and 112548 for each period.

4.2 Brief Description of Other Visualization Methods used for Comparison

We first explain the existing spherical embedding method as our primal comparison method, whose problem framework is the same to ours. In this method the following objective function is directly minimized with respect to the position vectors $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_M)^T$ and $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_N)^T$ under the constraints that $\mathbf{x}_m^T \mathbf{x}_m = r_A^2$ and $\mathbf{y}_n^T \mathbf{y}_n = r_B^2$ for $1 \leq m \leq M$ and $1 \leq n \leq N$. The objective function is defined as $\mathcal{J}(\mathbf{X}, \mathbf{Y}) = \frac{1}{2} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(c_{m,n} r_A r_B - \mathbf{x}_m^T \mathbf{y}_n \right)^2$, where $c_{m,n} = 2a_{m,n} - 1$, i.e., $c_{m,n} = 1$ if $(m, n) \in E$; $c_{m,n} = -1$ otherwise. In order to obtain the solution vectors, this method repeatedly moves each position vector by using the Newton method in a framework of nonlinear optimization, i.e., it repeats the following two steps: First, minimizing $\mathcal{J}(\mathbf{X}, \mathbf{Y})$ for $\mathbf{x}_m$ by fixing $\{\mathbf{x}_1, \cdots \mathbf{x}_M\} \setminus \mathbf{x}_m$ and $\{\mathbf{y}_1, \cdots \mathbf{y}_N\}$, and next minimizing $\mathcal{J}(\mathbf{X}, \mathbf{Y})$ for $\mathbf{y}_n$ by fixing $\{\mathbf{x}_1, \cdots \mathbf{x}_M\}$ and $\{\mathbf{y}_1, \cdots \mathbf{y}_N\} \setminus \mathbf{y}_n$. Thus this method requires an extremely large computation time to obtain the final results.

We have further compared the proposed method with the four well known embedding methods: multi-dimensional scaling [6], spectral embedding [1], spring force embedding [2], and cross-entropy embedding [7]. Here the former two perform a power iteration with respect to either a double-centered distance matrix or a graph Laplacian matrix which is calculated from a given graph, just like our proposed spherical embedding method does, while the latter two repeatedly move each position vector by using the Newton method in a framework of nonlinear optimization, just like the existing spherical embedding method does. Note that these four methods are not designed for embedding bipartite graphs, but as mentioned earlier, we can straightforwardly apply them for our purpose because a bipartite graph is regarded as an instance of general undirected graph.

In what follows in this subsection, we regard a bipartite graph as an undirected graph $G = (V, E)$ to describe the basic ideas of these standard embedding methods, and then consider a framework of embedding it into a K-dimensional Euclidean space. In this framework, we identify the set of the nodes by a positive integer, i.e., $V = \{1, \cdots, l, \cdots, L\}$, $|V| = L$ and $L = M + N$. Then, we can define the $L \times L$ adjacency matrix $\mathbf{A} = \{a_{m,n}\}$ by setting $a_{m,n} = 1$ if $(m, n) \in E$; $a_{m,n} = 0$ otherwise. We denote the K-dimensional embedding position vectors by $\mathbf{x}_m$ for the node $m \in V$, and then

[3] http://chiebukuro.yahoo.co.jp/

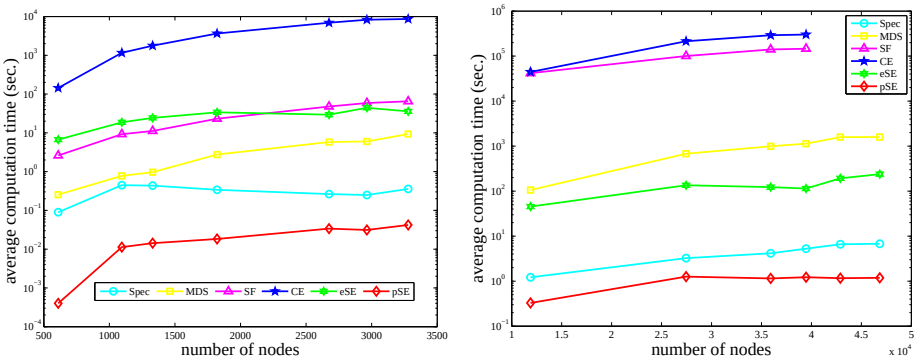
construct an $L \times K$ matrix consisting of these position vectors, i.e., $\mathbf{X} = (\mathbf{x}_1, \cdots \mathbf{x}_L)^T$. We also denote the graph distance matrix by $\mathbf{G} = \{g_{m,n}\}$, each element of which is the minimum path length between node m and node n.

Multi-dimensional scaling method [6] first calculates the distance matrix $\mathbf{G}$, and performs the double centering operation ($\mathbf{H}_L = \mathbf{I}_L - \frac{1}{L}\mathbf{1}_L\mathbf{1}_L^T$) to the distance matrix. Mathematically it is formulated as minimizing $\mathcal{M}(\mathbf{X}) = \frac{1}{2}\sum_{k=1}^{K}\mathbf{z}_k^T(\mathbf{H}_L\mathbf{G}\mathbf{H}_L)\mathbf{z}_k$, where $\mathbf{z}_k = (x_{1,k}, \cdots, x_{L,k})^T$, and $\{\mathbf{z}_1, \cdots, \mathbf{z}_K\}$ need to be orthonormal vectors, i.e., $\mathbf{z}_k^T\mathbf{z}_k = 1$ and $\mathbf{z}_k^T\mathbf{z}_{k'} = 0$ if $k \neq k'$. Spectral embedding method [1] tries to directly minimize distances between position vectors of connecting nodes. Mathematically it is formulated as minimizing $\mathcal{S}(\mathbf{X}) = \sum_{k=1}^{K}\mathbf{z}_k^T(\mathbf{D} - \mathbf{A})\mathbf{z}_k$, where $\mathbf{D}$ is a diagonal matrix each element of which is the degree of node (number of links). Note that $(\mathbf{D}-\mathbf{A})$ is referred to as a graph Laplacian matrix. Again, we set $\mathbf{z}_k = (x_{1,k}, \cdots, x_{L,k})^T$, and $\{\mathbf{z}_1, \cdots, \mathbf{z}_K\}$ need to be orthonormal vectors, which excludes the trivial vector expressed as $\mathbf{z} \propto \mathbf{1}_L$. Spring force embedding method [2] assumes that there is a hypothetical spring between each connected node pair and locates nodes such that the distance of each node pair is closest to its minimum path length at equilibrium. Mathematically it is formulated as minimizing $\mathcal{K}(\mathbf{X}) = \sum_{m=1}^{L-1}\sum_{n=m+1}^{L}\alpha_{m,n}(g_{m,n} - \|\mathbf{x}_m - \mathbf{x}_n\|)^2$, where $\alpha_{m,n}$ is a spring constant which is normally set to $1/(2g_{u,v}^2)$. Cross-entropy embedding method [7] first defines a similarity $\rho(\mathbf{x}_m, \mathbf{x}_n)$ between the embedding positions $\mathbf{x}_m$ and $\mathbf{x}_n$ and uses the corresponding element $a_{m,n}$ of the adjacency matrix as a measure of distance between the node pair, and tries to minimize the total cross entropy between these two. Mathematically it is formulated as minimizing $C(\mathbf{X}) = -\sum_{m=1}^{L-1}\sum_{n=m+1}^{L}(a_{m,n}\log\rho(\mathbf{x}_m, \mathbf{x}_n) + (1 - a_{m,n})\log(1 - \rho(\mathbf{x}_m, \mathbf{x}_n)))$. Here, note that we used the function $\rho(\mathbf{x}_m, \mathbf{x}_n) = \exp(-\frac{1}{2}\|\mathbf{x}_m - \mathbf{x}_n\|^2)$ in our experiments.

4.3 Experimental Results

We first evaluated the efficiency of our proposed method in comparison with the existing methods. We show our experimental results in Fig. 3, where Spec, MDS, SF, CE, eSE and pSE stand for the spectral embedding, multi-dimensional scaling, spring force embedding, cross-entropy embedding, existing spherical embedding and proposed spherical embedding methods, respectively (machine used is Intel(R) Xeon(R) CPU X5472 @3.0GHz with 64GB memory). Here Figs. 3a and 3b correspond to the results by using the bipartite graphs constructed from the Yahoo!Movies and Yahoo!Answers sites, respectively. In these figures, we plotted the average processing time (sec.) over 10 trials by changing the initial position vectors, where the horizontal and vertical axes stand for the number of nodes in V_B and the processing times, respectively. Here recall that the number of nodes in V_B is different for each bipartite graph as mentioned above.

As expected, these figures show that our proposed spherical embedding (pSE) method works much faster than all the existing methods we compared. More specifically, the spectral embedding (Spec) method works comparable to our method. This is because these methods perform a power iteration on a sparse adjacency matrix. In fact, the multi-dimensional scaling (MDS) method requires a substantially large computation time because it needs to perform a power iteration on a full distance matrix. All the other methods including the existing spherical embedding (eSE) method, which repeatedly move each position vector by using the Newton method, generally require an extremely large computation time before the final results are obtained. Especially, both the spring



(a) Yahoo!Movies (b) Yahoo!Answers

Fig. 3: Comparison of processing times.

force embedding (SF) and cross-entropy embedding (SE) methods require more than three days to obtain the final results even for one trial when the numbers of nodes for the Yahoo!Answers graphs become more than 40,000; thus we omitted these results in Fig. 3b. Here we should emphasize that the scale of the vertical axis of these figures is logarithmic.

Next we evaluated the visualization results of our proposed method in comparison with the existing methods. Due to a space limitation, we only show our experimental results obtained for a bipartite graph constructed from the Japanese Yahoo!Movies sites in Fig. 4. Here recall that the genre information has been shown in Fig. 2. In Fig. 4a, we show the visualization result by our proposed method, which we consider intuitively natural. Actually, we can see that the genre nodes of Action/Adventure (black square) and Suspense (maroon hexagon) are located in near positions at the right-side of the inner circle (θ_A), while at the opposite left-side of this circle, the genre nodes of Teen (orange triangle_up) and Romance (cyan square) are located in near positions. Overall, we can observe that the similar genres are located closely on the inner circle (θ_A).

Now we compare the above results with the five existing methods. The first one is the visualization result by the existing spherical embedding method shown in Fig. 4b. We see that there are several minor differences but we consider this result comparable to the result by our method. However, this one is very slow and inefficient. Our method is much faster. The second one is the visualization result by the multidimensional scaling method shown in Fig. 4c. We can observe some clusters of genres. Although this result might indicate some intrinsic property, we feel that the spherical embedding scheme is a more natural representation of bipartite graphs. The third one is the visualization result by the spectral embedding method shown in Fig. 4d. This one is relatively poor in our own experiments. In fact, the two genres of Drama (lime cross) at the bottom-right and Documentary (Olive triangle_right) at the top-left are too much isolated, although this method works reasonably fast among the existing methods.The fourth and the fifth ones are the visualization results by the spring force embedding method and the cross-entropy embedding method shown in Figs. 4e and 4f. We can observe a similar tendency

between these two, *e.g.*, we can easily see that the genre node of Drama (lime cross) is much isolated in both. The main difference in these methods is that we can observe that some genre nodes are clustered for the spring force embedding method, but there are no such clusters and all the genres are scattered for the cross-entropy embedding method. Overall, although each embedding method might have its own characteristics that are both advantageous and disadvantageous, we believe that our proposed spherical embedding method is most effective for visualizing bipartite graphs in terms of efficiency and interpretability.

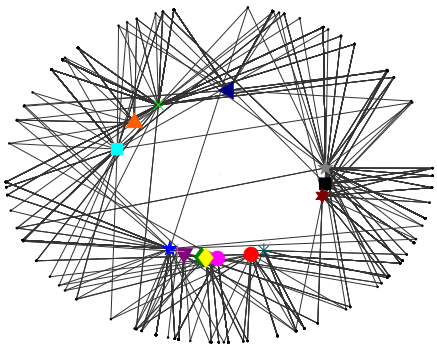
Last but not least, we evaluated our proposed method only in the case of two-dimensional embedding for our visualization purpose, but this does not mean that it is limited to two-dimensional embedding. It is quite easy to extend it to the general K-dimension embedding. We plan to evaluate our method as a powerful technique for both dimensional reduction and clustering as a future work.

5 Conclusion

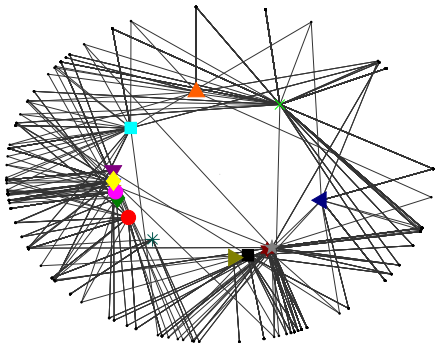
In this paper, we addressed the problem of visualizing structure of bipartite graphs such as relations between pairs of objects and their multi-labeled categories, and proposed a new spherical embedding method that is based on a power iteration. The key features of this method is that it employs two operations on the positioning vectors, one called double-centering operation and the other called normalizing operation. This enables the iterative approach to be equivalent to maximizing an objective function which is guaranteed to converge. Thus, our algorithm is theoretically guaranteed to converge. We applied our method to a set of bipartite graphs with different sizes and connections, and compared the results with five existing visualization methods. The results showed that the proposed method works much faster than all the five existing methods, and the visualization results are intuitively understandable and comparable to the best available so far known. In future, we plan to apply the new method to evaluate its performance and robustness for a wide variety of bipartite graphs.

References

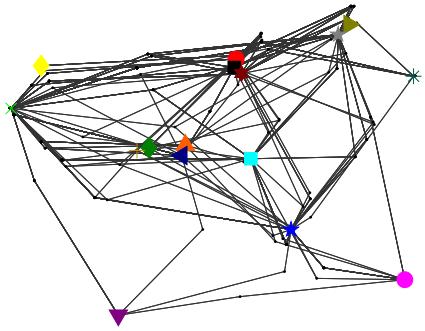
1. Chung, F.R.K.: Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92). American Mathematical Society (Feb 1997)
2. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. Inf. Process. Lett. 31, 7–15 (April 1989)
3. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. J. ACM 46, 604–632 (September 1999)
4. Langville, A.N., Meyer, C.D.: Deeper inside pagerank. Internet Mathematics 1, 2004 (2004)
5. Naud, A.P., Usui, S., Ueda, N., Taniguchi, T.: Visualization of documents and concepts in neuroinformatics with the 3d-se viewer. Frontiers in Neuroinformatics 1, Article 7 (2007)
6. Torgerson, W.S.: Theory and methods of scaling. John Wiley & Sons Inc. (1958)
7. Yamada, T., Saito, K., Ueda, N.: Cross-entropy directed embedding of network data. In: Proceedings of the 20th International Conference on Machine Learning (ICML03). pp. 832–839 (2003)



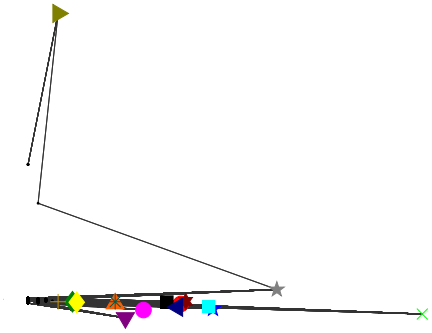
(a) proposed spherical embedding (pSE)



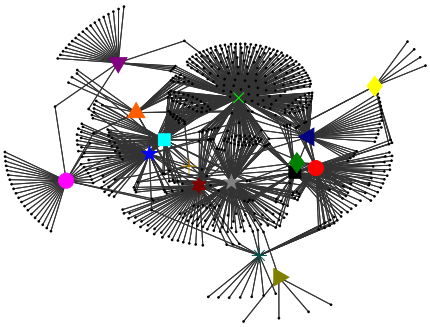
(b) existing spherical embedding (eSE)



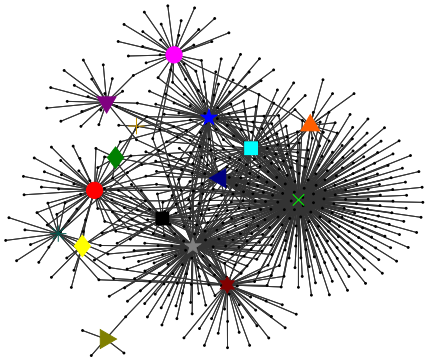
(c) multi-dimensional scaling (MDS)



(d) spectral embedding (Spec)



(e) spring force embedding (SF)



(f) cross-entropy embedding (CE)

Fig. 4: experimental results obtained for a bipartite graph constructed from the Japanese Yahoo!Movies sites(1950 - 1959)