Neurocomputing 124 (2014) 97–104

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Data visualization for asymmetric relations

Heeyoul Choi *

Samsung Advanced Institute of Technology, Samsung Electronics, 97, Samsung2-ro, Giheung-gu, Yongin-si 446-712, Gyeonggi-do, Republic of Korea

ARTICLE INFO

Article history:
Received 28 January 2013
Received in revised form
25 April 2013
Accepted 24 July 2013
Communicated by S. Choi
Available online 2 September 2013

Keywords:
Asymmetric proximity
Data visualization
Multidimensional scaling

ABSTRACT

Most 2D visualization methods based on multidimensional scaling (MDS) and self-organizing maps (SOMs) use a symmetric distance matrix to represent and visualize object relationships in a data set. In many real-world applications, however, raw data such as a world-trade data are best captured as an asymmetric proximity matrix. Such asymmetric matrices cannot be perfectly represented by most previous methods. To handle such an intrinsic limitation, in this paper, we propose a dynamic learning for metric representations of asymmetric proximity data to better understand the data. The proposed learning generates two representations (maps) with the row vectors (sending or exporting) and column vectors (receiving or importing) of the matrix, respectively. To better present the patterns, we supplement the maps with two analysis tools: cluster analysis and distance analysis, which connect and compare the different patterns from the different maps. Experiment results using three real world data sets confirm that the proposed learning method is useful to understand asymmetric proximity data.

1. Introduction

Visualization is a procedure that helps represent the complex data (usually in a high-dimensional space) in an effective way (usually in a low-dimensional space). Dimensionality reduction algorithms have been used for this purpose. Multidimensional scaling (MDS) [1] and principal component analysis (PCA) [2] are popular linear methods for dimensionality reduction. Manifold learning is a non-linear dimensionality reduction approach, which induces a smooth nonlinear low-dimensional manifold from a set of data points drawn from the manifold. Recently, various dimensionality reduction methods (for example see [3–6]) have been developed in machine learning community drawing an attention in pattern recognition and signal processing.

In data analysis, many times, information is provided as a similarity (or dissimilarity) matrix, whose elements could be the distances between the data points. For visualization of such data, most methods based on a dimensionality reduction approach such as MDS or self-organizing maps (SOMs) [7] rely on a symmetric matrix which represents the object relationships [3]. Although sometimes those methods seem to be applied to an asymmetric matrix, they convert the asymmetric matrix into a symmetric one, and actually work on the symmetric matrix [8–10]. However, a symmetric matrix cannot represent any directional relationship between data points, while asymmetric matrix can. It means potential information loss if we use only symmetric relationship.

In many real-world applications [11], raw data can be best captured as an asymmetric proximity matrix. For example, a world-trade dataset can be represented as an asymmetric matrix with each column and each row corresponding to one country, and each cell indicating the money amount transferred from one country to another. Journal citation data (or bibliometric data) is another example for asymmetric similarity (or dissimilarity), where the numbers of citation between two journals are usually different [12]. In pattern classification, a confusion matrix as in Morse code [13] is asymmetric. Also, in social network service, the following relationship as in Twitter (www.twitter.com) can be represented as an asymmetric matrix, while the friendship as in Facebook (www.facebook.com) is symmetric.

To visualize such asymmetric proximity data, some methods have been proposed. However, most previous methods first decompose it into a symmetric and a skew-symmetric component, and then put much focus on the symmetric component using MDS-like methods [14–17]. Those methods do not present the data as 2D points with their asymmetric properties, and fail to visualize asymmetric properties geometrically on a metric space and also fail to discover regularities in such data (e.g. differences between importing and exporting within and across multiple countries). Furthermore, separating the representations of the symmetric and skew-symmetric components makes it very hard to understand nature structures and properties of the data set intuitively.

In fact, asymmetric relationships cannot be represented in any single metric space including 2D space, keeping the asymmetric properties perfectly. Once a data set is represented as points in a metric space, their geometric relations become symmetric. MDS-like methods implicitly assume that the proximity matrix was obtained

* Tel.: +82 10 9623 6578.
E-mail address: heeyoul@gmail.com

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.07.030

from the data points in a metric space, which makes the methods not capable of handling asymmetric matrices. Therefore, such 2D representations cannot present the asymmetric properties. But still visual representation, if possible, is very attractive, so we propose a dynamic learning algorithm of asymmetric relations to represent the asymmetric data on metric spaces. We plot the data onto two different maps separately, focusing on one direction in the asymmetric matrix at a time: one based on the column vectors (e.g. importing) and one based on the row vectors (e.g. exporting).

In addition, the two maps are supposed to be considered together, since they are derived from the same matrix. To enable users to better understand the data, we supplement the two maps with two analysis tools on the maps: cluster analysis and distance analysis, allowing users to connect and compare various patterns within each map and across two maps. These tools provide quantitative measures of the difference between before and after learning, or the difference between the sending map and the receiving maps. Although the proposed method does not provide a perfect view of asymmetric properties, it increases users' understanding of the asymmetric data.

Several case studies conducted with real data sets reveal the effectiveness of our approach. The cola-brand-switching data is presented to explain the maps and analysis functions, and the other data sets follow after that to confirm that the proposed method is useful to understand asymmetric proximity data.

2. Background

2.1. Asymmetric proximity data

Given N objects with a similarity matrix $\boldsymbol{S}$, where S_{ij} is the proximity between the ith and jth objects. Asymmetric relation is defined when $S_{ij} \neq S_{ji}$. If this inequality is from noise or error, we can symmetrize it by $(\boldsymbol{S}+\boldsymbol{S}^T)/2$. Otherwise, we have to deal with the asymmetric property differently from the previous methods that are based on the symmetry assumption.

In this paper, to present how our method works, we use three data sets: cola-brand switching data between 15 cola brands [14], threat display behaviours data [18], and 113 countries' trading data in 2009 from International Monetary Fund (IMF) web page, www.imf.org. These data can be summarized as an asymmetric matrix as shown in the left of Fig. 1. Note that for the world-trade data, we use logarithmic values of the trade amount since the absolute amount of trade is too much dominated by a few countries. The directional flow of the trade among the countries can be presented as in the right of Fig. 1. However, it is hard to understand the underlying pattern from the matrix or the directional flow arrows.

2.2. Previous work

Although multidimensional scaling (MDS) is not perfect for asymmetric data, it can approximately present the data points on a 2D space after the symmetrization procedure. Many previous methods are based on MDS and our proposed learning method takes MDS as an initial step to present the asymmetric properties of data.

2.2.1. Multidimensional scaling

To simply connect the asymmetric matrix to classical MDS, we need to transform the asymmetric matrix to a distance matrix. Given an asymmetric matrix, $\boldsymbol{M}$ for N objects, we make a symmetric matrix

$$\boldsymbol{M}^s = \frac{\boldsymbol{M}+\boldsymbol{M}^T}{2}. \tag{1}$$

Then, as in [1], a distance matrix, $\boldsymbol{D}$, can be given by

$$D_{ij} = 1 - M^s_{ij}. \tag{2}$$

Note that $\boldsymbol{M}$ is a normalized matrix by the maximum value of the elements of $\boldsymbol{M}$. Next, the constant adding technique is applied to make sure that each distance is the length of two points on a metric space, as in kernel Isomap [19]. In classical MDS, given a distance matrix, $\boldsymbol{D}$, a popular cost function is defined by

$$J = \sum_{ij} (d(\boldsymbol{x}_i, \boldsymbol{x}_j) - D_{ij})^2, \tag{3}$$

where $d(\cdot,\cdot)$ is a binary function to calculate the Euclidean distance between two points on a low dimensional space, and $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]$ is a representation on the low-dimensional space. Let $\boldsymbol{B}$ be the inner product matrix, where

$$\boldsymbol{B} = \boldsymbol{X}^\top \boldsymbol{X}. \tag{4}$$

Considering $\boldsymbol{B}$ as a kernel matrix as described in [6], $\boldsymbol{B}$ can be given by

$$\boldsymbol{B} = -\tfrac{1}{2}\boldsymbol{H}\boldsymbol{D}^2\boldsymbol{H}, \tag{5}$$

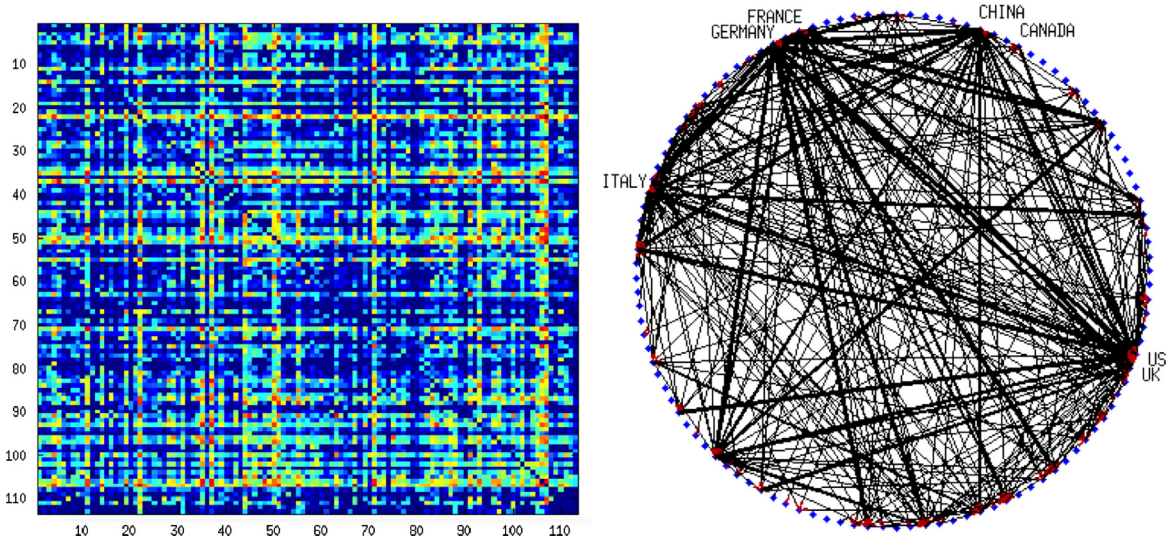

Fig. 1. Asymmetric world-trade data. (Left) 113 countries' trade data matrix, (Right) the directional flow of the trade which shows only the flows with higher values than twice the standard deviation of the matrix elements from the mean. The G7 countries' names are shown.

where $\boldsymbol{D}^2$ means element-wise square of $\boldsymbol{D}$, $\boldsymbol{H}$ is the centering matrix, given by

$$\boldsymbol{H} = \boldsymbol{I} - \frac{1}{N}\boldsymbol{e}\boldsymbol{e}^\top, \tag{6}$$

and $\boldsymbol{e} = [1, \ldots, 1]^\top \in \mathbb{R}^N$. Finally, connecting the two formulations, Eqs. (4) and (5), the solution of the cost function is given by

$$\boldsymbol{X} = \boldsymbol{\Lambda}^{1/2}\boldsymbol{V}^\top, \tag{7}$$

where $\boldsymbol{V}$ and $\boldsymbol{\Lambda}$ are the eigenvector and the eigenvalue matrix of $\boldsymbol{B}$, respectively. See [6] for more details.

2.2.2. Other methods for asymmetric proximities

To geometrically understand asymmetric proximity data on a 2D space, several approaches have been applied. First, the simplest way would be to symmetrize the asymmetric matrix in different ways, and then apply MDS-like methods [15,16]. Second, some methods decompose the asymmetric matrix into a symmetric and a skew-symmetric component, and plot the two matrices onto two separate spaces using MDS-like methods [17] or use only the symmetric matrix [15]. Some others put much focus on the symmetric component using MDS-like methods, and the skew-symmetric component is added to the symmetric part as in Gower diagram, drift vectors, or unfolding method [14].

Those approaches fail to visualize and discover asymmetric properties geometrically on a 2D space and regularities in such data (e.g. differences between importing and exporting within and across multiple countries). It is partly due to the intrinsic property that any single 2D representation cannot preserve perfectly asymmetric relationship between the points. So, we propose to produce two separate representations with the column vectors (receiving) and the row vectors (sending) in an asymmetric matrix, respectively. This method reveals the hidden asymmetric dynamics. Also, since the method generates two maps, we need some analysis tools to connect and compare various patterns within each map and across the two maps.

3. Dynamic learning method

In this section, we propose a dynamic learning method to represent geometrically the hidden dynamics of the two directions in the asymmetric matrix: sending or receiving. The directional information can be interpreted as how objects are attracted to other objects, and we can move all the points based on the amount of the attraction. As some previous methods, we apply MDS to the symmetrized matrix to have an initial representation. From the MDS representation, the proposed method moves all the points (representation of objects) iteratively, until they converge to equilibrium statuses, which may reveal the dynamics in the two directions.

For example, Fig. 2 shows the MDS representation of the symmetrized matrix of the cola-brand switching data, whose elements mean how many consumers have switched the cola brand from one to another. Three clusters are presented: diet type cola, original type cola, and the rest type cola including 'decaf'. There are more transitions within each cluster than across clusters. However there is no asymmetric relation in this figure. From this representation, the proposed method moves all the points to generate two maps: a sending (or exporting) and a receiving (or importing) maps.

3.1. Learning algorithm

Without loss of generality, in this section, we use sending information. Let the points on the low-dimensional space be $\boldsymbol{X}$. For the ith point $\boldsymbol{x}_i$, the sending information is given by M_{ij}, for $j = 1, 2, \ldots, N$. Now, we move $\boldsymbol{x}_i$ based on the sending information. The goal is to move $\boldsymbol{x}_i$ as much as needed, which is the difference between the

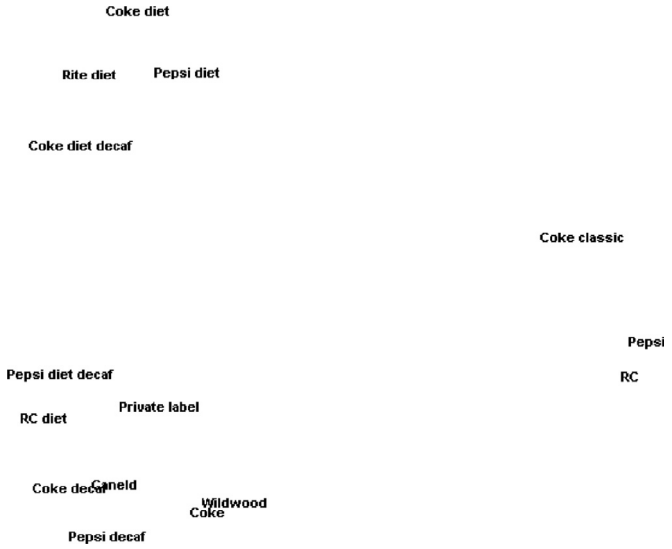

Fig. 2. MDS result on the symmetrized matrix from the cola-brand-switching data. The closer the two brands are, the more the transitions occur between them. Note that the real point of the brand on the coordinate is located on the left of the string.

symmetric matrix $\boldsymbol{M}^s$ and the asymmetric matrix $\boldsymbol{M}$. That is, $M_{ij} - M^s_{ij}$ means the amount of movement that $\boldsymbol{x}_i$ is supposed to take toward $\boldsymbol{x}_j$. After normalization, the amount is given by

$$P_{ij} = \frac{M_{ij} - M^s_{ij}}{M^s_{ij}}. \tag{8}$$

However, since M^s_{ij} and M_{ij} are not updated, we need to know when they should stop moving. So, we can measure how much they have moved by

$$Q_{ij} = \frac{D_{ij} - d_{ij}}{D_{ij}}, \tag{9}$$

where d_{ij} is the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ on the current iteration. Here, since Q_{ij} is the amount of relative distance that the point $\boldsymbol{x}_i$ has moved toward $\boldsymbol{x}_j$, at the current iteration, a point $\boldsymbol{x}_i$ needs to move toward $\boldsymbol{x}_j$ at the amount of $P_{ij} - Q_{ij}$. Considering all the points, finally the update rule of $\boldsymbol{x}_i$ at time t is given by

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \alpha \sum_j (P_{ij} - Q_{ij})(\boldsymbol{x}_j^t - \boldsymbol{x}_i^t), \tag{10}$$

where α is a learning rate, set to around 0.1 in our experiments.

For the receiving information, the update rule for $\boldsymbol{x}_i$ is the same as the one for the sending information, except using $(P_{ji} - Q_{ji})$ instead of $(P_{ij} - Q_{ij})$. Note that this approach can work with massive data sets since the update rule in Eq. (10) can be executed in parallel. Also, $\boldsymbol{x}_j$ in Eq. (10) can be selected from only the neighbors of $\boldsymbol{x}_i$ assuming that the non-neighbors do not affect much on $\boldsymbol{x}_i$.

To stop the iteration using the sending information, we check the change in the update defined by

$$E = \frac{1}{N}\sum_i \| \boldsymbol{x}_i^{t+1} - \boldsymbol{x}_i^t \|. \tag{11}$$

If E is lower than a threshold, 10^{-5} in our experiments, we consider that the iteration has converged. E for the receiving map is defined in the same way.

With this method, one can see the hidden dynamics in the asymmetric relationship. In Fig. 3, note that *Pepsi decaf* is different in the two maps. In the sending map, *Pepsi decaf* is out of the cluster which means people have not much changed from that to the other brands. However, in the receiving map, it has moved into the center of the cluster which means people have replaced the other brands by *Pepsi decaf*.

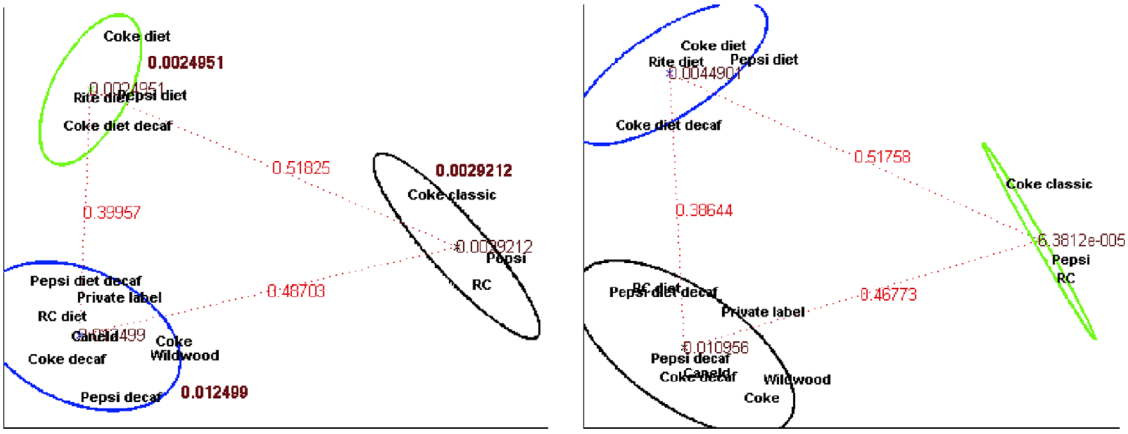

Fig. 3. Cluster analysis on each map of the cola data after dynamic learning. (Left) Sending map, and (Right) receiving map. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

This method can be understood in analogy of the gravity model. In the gravity, both objects have the same force to move, but the amount of movement is different due to the different masses. For example, an apple falls down to the earth, but the earth does not to the apple. Likewise, in the sending map and receiving map, we can interpret that the same object has different masses on the different maps. The final representations are the equilibrium statuses of the dynamics on the maps, as the objects converge to the equilibrium status of the gravity model.

Note that even at the equilibrium status, the points might be following other points and make circles as an ant mill where army ants follow one another in circle to death. Also, an MDS representation has rotation ambiguity. So, after updating all the points at each iteration, we need to rotate the updated representation to make it as similar to the previous one as possible.

3.2. Rotating a representation

We provide a method to rotate the updated representation to minimize the difference to the previous one. Let $\boldsymbol{X}=[\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_N]$ and $\boldsymbol{Y}=[\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_N]$ be the two representations of the points before and after moving the points, respectively. The goal is to rotate $\boldsymbol{Y}$ so that $\boldsymbol{Y}$ is as similar to $\boldsymbol{X}$ as possible. A cost function for this goal can be formulated by

$$J_r=\sum_i(\boldsymbol{x}_i-\boldsymbol{R}\boldsymbol{y}_i)^2, \tag{12}$$

where $\boldsymbol{R}$ is a rotation matrix defined by

$$\boldsymbol{R}=\begin{bmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{bmatrix}, \tag{13}$$

where θ is the angle that $\boldsymbol{Y}$ is supposed to rotate. Note that if $\det(\boldsymbol{R})$ is negative, it means reflection should be considered, so before finding the rotation matrix, the second dimension of $\boldsymbol{Y}$ is negated.

Then, using the least square method, the cost function has a unique algebraic solution which is given by

$$\boldsymbol{R}=(\boldsymbol{X}\boldsymbol{Y}^T)(\boldsymbol{Y}\boldsymbol{Y}^T)^{-1}. \tag{14}$$

However, since $\boldsymbol{Y}$ is not perfectly generated by rotating $\boldsymbol{X}$, $\boldsymbol{R}$ is not a pure rotation matrix and we need to extract just the rotation part from $\boldsymbol{R}$. To find the angle, θ, for the rotation, we use a simple trick. Let $\boldsymbol{v}=[1,0]^T$ and $\boldsymbol{u}=\boldsymbol{R}\boldsymbol{v}$. Then, the angle, θ, is obtained by

$$\theta=\mathrm{atan}(u_y,u_x), \tag{15}$$

where $\boldsymbol{u}=[u_x,u_y]^T$.

If the determinant of $\boldsymbol{Y}\boldsymbol{Y}^T$ is close to zero, which is the case when $\boldsymbol{Y}$ has almost all the points on the same line, then the estimation of $\boldsymbol{R}$ is not stable. In that case, we take the two points which has the longest distance in $\boldsymbol{Y}$, and directly calculate the angles between the line in $\boldsymbol{Y}$ and the corresponding line in $\boldsymbol{X}$. Then, as stated above, the rotation matrix is obtained by Eq. (13).

4. Visual analysis tools

After plotting the representations on two maps, to make the representations more comprehensible, we apply two analysis tools on the two maps: *cluster analysis* and *distance analysis*, allowing users to connect and compare various patterns within each map and across the two maps.

4.1. Cluster analysis

After dynamic learning of the representations, the points are distributed differently on the sending and receiving maps. Cluster analysis can show the distribution of the points, which helps users better understand how the points are distributed on each map.

Our cluster analysis is based on the expectation minimization (EM) algorithm with a mixture of Gaussian model for the density estimation of the points on 2D spaces [20]. In the EM algorithm, generally, the number of mixture of Gaussians should be given. Instead, we set the maximum number of Gaussians and execute EM algorithm with different mixture numbers. Then, we find the best fitting number using model selection methods such as normalized entropy criterion (NEC), Bayesian inference criterion (BIC) and integrated completed likelihood (ICL) [3]. Note that some other clustering methods such as *k*-means can be adopted too.

Fig. 3 shows the cluster analysis of the cola-brand-switching data after the dynamic learning. The EM algorithm was used on each map with ICL. On such a clearly separable data, the number of clusters is the same on the both maps. The clusters are presented by ellipses from the covariance matrix. After clustering, the distances between the clusters' centers are calculated in red as in Fig. 3. Also, the determinant of each cluster's covariance matrix is calculated in brown to show relatively the area of the corresponding cluster. The bigger the determinant is, the less the transition happens in the cluster compared to the transition in the other clusters.

4.2. Distance analysis

Distance analysis calculates the difference between the distances of point to other points on two maps. Basically these

comparisons are conducted on 3 pairs of maps: (1) the sending map and the initial map, (2) the receiving map and the initial map, and (3) the sending map and the receiving map. This analysis presents how individual variables' geometric relationship with other variables change due to the dynamic learning. This analysis allows the users to compare quantitatively the representations of individual variables. While cluster analysis gives somehow big picture of how the data represented on the two maps, distance analysis compares the distances of individual variables before and after the learning, and compares the distances between the sending and receiving maps.

In Fig. 4 the top and the middle graphs are for either the sending map or the receiving map. On the other hand, the bottom graph compares the sending map and the receiving map directly to show some interesting pattern. From this analysis, users can see which variables get closer to which variables by the dynamic learning. Also, users can measure how close or far the variables are to the other variables in the sending and receiving maps. Note that the highlighted variables (in this figure, numbers 1 and 4) are compared to all other variables. The numbers on the horizontal axes are brand indices of the cola data. The red bars are for *Pepsi decaf* and the blue bars for *Coke decaf*. For example, the red bars in the first row mean the differences between the distances on the sending map and the initial map of *Pepsi decaf* to all brands.

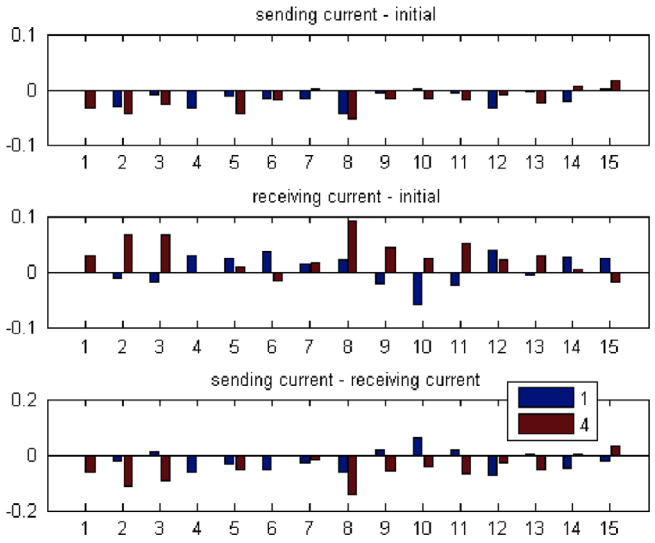

Fig. 4. Distance analysis of the cola data after the dynamic learning, which made the current maps. The before maps were the initial map by MDS. The variable number 1 is *Coke decaf* and 4 is *Pepsi decaf*. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

Basically, we can interpret the maps after learning as follows. If a variable 'A' becomes closer to a specific variable 'B' in a sending (or receiving) map, it means that 'A' sends much information to (or receives from) 'B'. However, if 'A' becomes closer to most variables in a sending (or receiving) map, then the other variables send much information to (or receives from) 'A'. Also, their inversions are possible. For example, if 'A' becomes far way from 'B' in a sending map, then it means that 'A' sends less information to 'B'.

For example, from Figs. 3 and 4, one can see that after the learning, *Pepsi decaf* and *Coke decaf* went closer to the other brands in the sending map. This means that many people have changed from the other brands to the two decaf brands. Meanwhile, in the receiving map, *Pepsi decaf* went further away from almost all the other brands, while *Coke decaf* moved away from some variables and toward the others, especially number 10 (*RC diet*). This means that many people keep enjoying *Pepsi decaf*, while some people have replaced their *RC diet* by *Coke decaf*. The third row says that *Pepsi decaf* is closer to other variables in the sending map than the receiving map, which means more likely people have moved from the other brands to *Pepsi decaf*.

5. Case studies

For case studies, we applied the proposed method to two more data sets: (1) threat display behaviors in a species of bird, great tit, and (2) world trade data.

5.1. Threat display behaviors

In [18], Jones measured preceding–following contingencies for certain types of threat display behaviors in the great tit. Fig. 5 shows the numbers corresponding to the proportion of times that the behavior in column *j* follows the behavior in row *i*, which is definitely asymmetric. For example, *Feeding* (or 6) follows *Fluffing* (or 11) with the probability of 3%, and after *Hopping around* (or 8), *Incomplete feeding* (or 7) follows with the probability of 46%. Fig. 6 shows the initial plot by MDS with the symmetrized matrix from the preceding–following contingencies matrix.

Figs. 7 and 8 show the two different maps with cluster analysis after the dynamic learning based on the sending and the receiving information, respectively. We can see some different patterns on the two maps. First of all, the number of clusters is different on the two maps. In the sending map, there are three clusters, while there is only one cluster in the receiving map. This means that in the sending map, the threat display behaviors precede the other behaviors in a clustered way. That is, a certain behavior is followed more probably by a small number of behaviors rather than the

Behavior types	1	2	3	4	5	6	7	8	9	10	11	12	13
1 Attack	4	17	16	11	10	13	11	0	6	0	0	9	4
2 Head down	26	0	5	14	4	13	2	8	5	0	0	5	18
3 Horizontal	25	3	0	12	13	11	3	2	10	8	0	4	9
4 Head up	5	9	8	8	14	15	5	4	13	0	2	5	12
5 Wings out	22	13	10	5	2	10	2	7	7	0	0	2	19
6 Feeding	2	5	18	13	11	3	3	5	13	8	1	16	1
7 Incomplete feeding	4	10	15	4	4	13	7	22	0	0	12	8	0
8 Hopping around	1	10	0	4	2	4	46	0	3	6	11	11	3
9 Hopping away	0	4	6	9	5	1	8	4	1	6	31	15	10
10 Crest raising	0	0	0	6	7	3	0	11	17	1	30	13	12
11 Fluffing	0	4	5	6	3	3	0	23	13	35	0	6	3
12 Looking around	5	0	5	0	3	6	12	12	11	30	8	0	9
13 Hopping towards	5	25	12	8	21	4	2	2	2	7	5	6	0

Fig. 5. Preceding–following contingencies matrix.

Fig. 6. Initial plot of the threat display behavioral data by MDS on the symmetrized matrix.

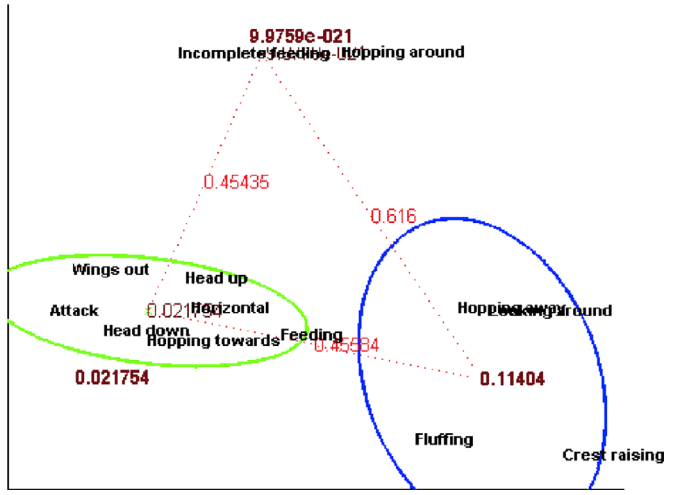

Fig. 7. Cluster analysis on the sending map of the threat display behavioral data. Three clusters are presented.

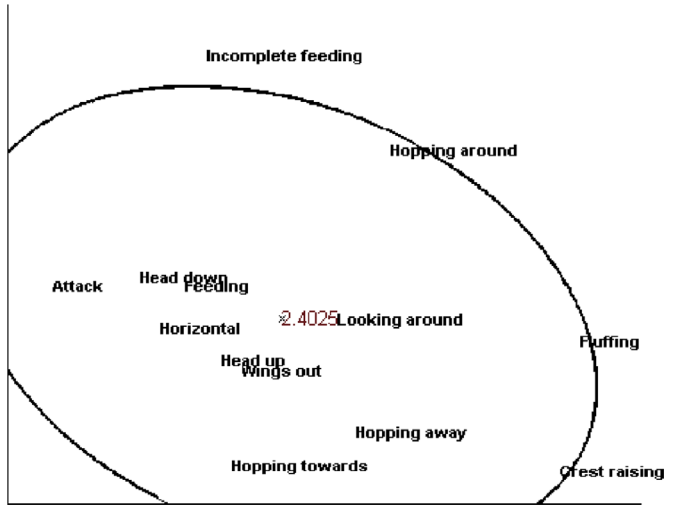

Fig. 8. Cluster analysis on the receiving map of the threat display behavioral data. There is only one cluster.

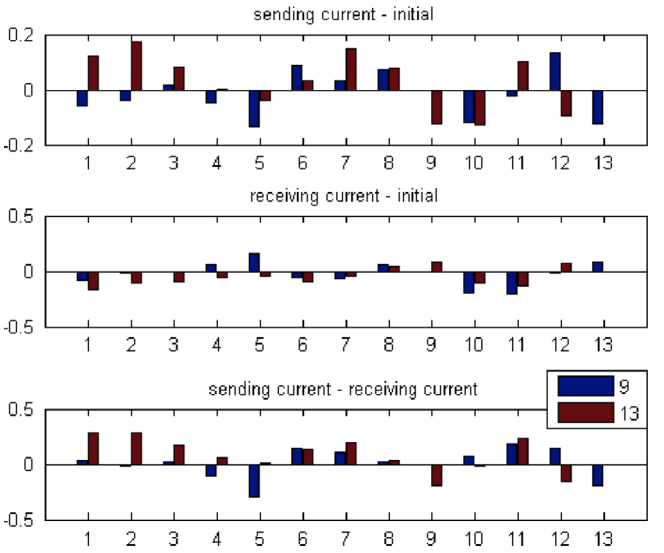

Fig. 9. Distance analysis between sending map and receiving map. *Hopping away* (9) and *Hopping towards* (13) have strongly asymmetric relationship with the other behaviors.

others. On the other hand, in the receiving map, the behaviors follow other behaviors without any patterns.

In addition to the global pattern, there are some different patterns for the individual behaviors. For example, *Wings out* is on the boundary of the cluster in the sending map and at the center of the cluster in the receiving map. From the sending map, we can see that *Wings out* precedes *Attack* more likely than the other behaviors, which is consistent with Fig. 5, where *Wings out* precedes *Attack* with the probability of 22% which is the most dominant probability. But, from the receiving map, we can tell that *Wings out* does not have any preference to follow any other behaviors, which is also consistent with Fig. 5. Based on the results, we can tell that the sending and receiving maps faithfully represent the properties of the asymmetric relations in metric spaces.

To check how much the distance has changed, we can apply distance analysis as shown in Fig. 9, where as an example *Hopping away* (9) and *Hopping towards* (13) are compared within and across the two maps. As we have seen on the maps, we can see that, compared to the initial plot, they became very close in the sending map (negative values for 9 and 13 in the top figure), and they became far from each other (positive in the middle figure). This means that they precede each other more likely than they do the other behaviors, but they follow each other less likely than they do others. Also, the comparison across the two maps shows directly that they are closer to each other on the sending map than on the receiving map (negative on the bottom figure), which can be interpreted in the same way as above.

5.2. World-trade data

In this section, we applied the dynamic learning algorithm to the world-trade data set. We extracted the 2009 direction of international trade (DOTS) data including 113 countries with the largest amount of trade. Note that we converted the empty cells in the table like *Afghanistan-Angola* into zero. As shown in Fig. 1, 113 countries' trade data can be summarized up to a 113×113 asymmetric matrix. The initial plot by MDS on the symmetrized matrix is shown in Fig. 10, where the clusters roughly keep the geographical proximities.

After moving all points iteratively based on the dynamic learning method, cluster analysis shows how the countries are distributed in

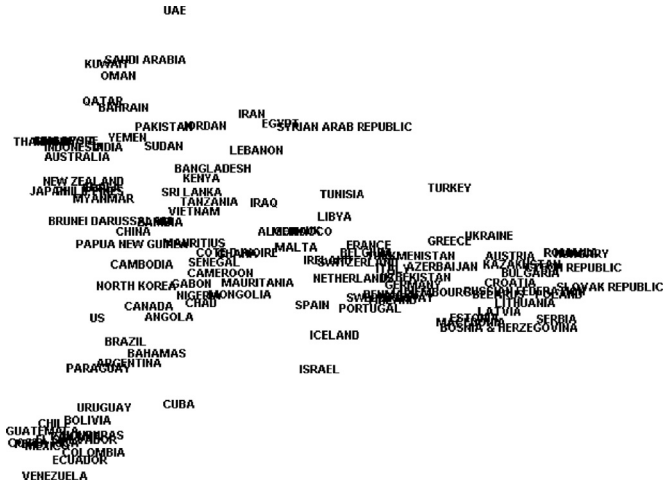

Fig. 10. Initial plot of the world trade data by MDS on the symmetrized matrix. The clusters roughly keep the geographical proximities.

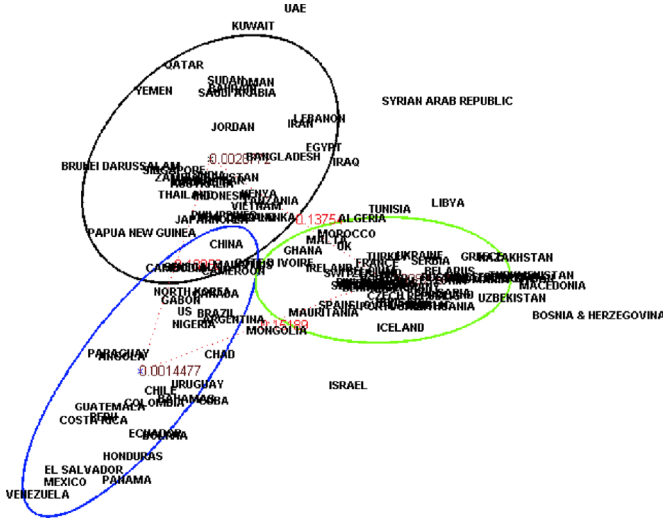

Fig. 11. Cluster analysis after dynamic learning on the exporting map of the world-trade data. Three clusters are presented.

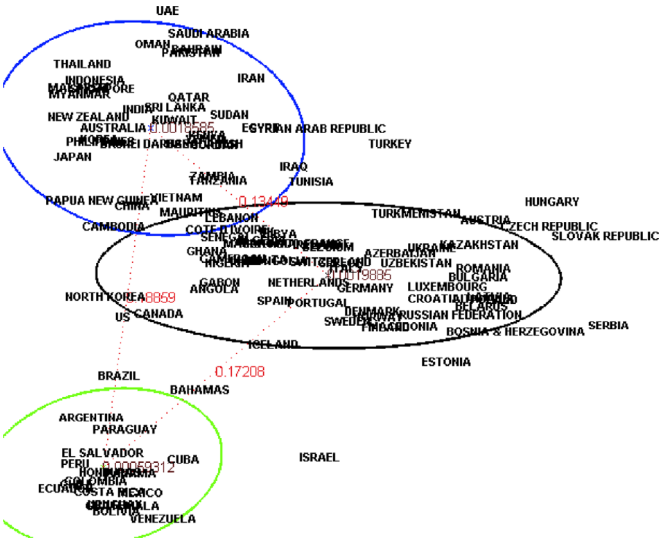

Fig. 12. Cluster analysis after dynamic learning on the importing map of the world-trade data. Three clusters are presented. The South American countries at the bottom are more separated from the other clusters compared to the exporting map.

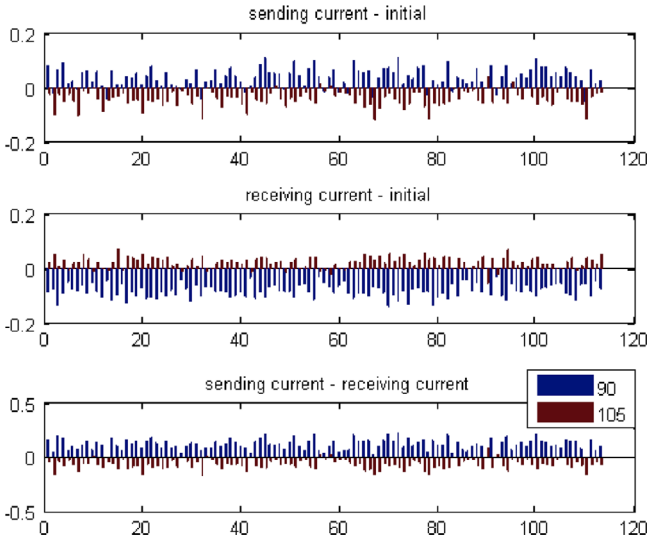

Fig. 13. Distance analysis between the sending map and the receiving map of the world-trade data. *Serbia* (90) and *UAE* (105) have strongly asymmetric relationship with the other countries.

Figs. 11 and 12. As above, we can see some patterns from the maps. For example, in the maps, the South American countries are in the same cluster. However, in the exporting (sending) map, the area of the ellipse measured by the determinant of the covariance matrix is larger than the one in the importing (receiving) map. Also it is closer to the other clusters in the exporting map. From these patterns, we can see that they import more from their geographical neighbors compared to export. Note that the geometric relationships on the maps are relatively the same as the flow amounts from the original asymmetric matrix, which means that the 2D representations reflect faithfully the asymmetric properties given by an asymmetric matrix.

Fig. 13 shows the distance changes within and across the maps. *Serbia* (90) is closer to the other countries in the importing (receiving) map than exporting (sending) map, and *UAE* (105) has the opposite relationship with the other countries. From this figure, we can see that for *Serbia*, importing is greater than exporting, while for *UAE*, exporting is greater than importing.

Although the 2-dimensional representations are not enough to provide the full understanding of the world-trade data set, we could increase our understanding of the data set.

6. Conclusion

In this paper, we proposed a dynamic learning method for metric representations of an asymmetric proximity data set to better understand it. The proposed learning method generates two representations (maps) with the column vectors (importing) and row vectors (exporting) of the matrix, respectively. In addition, by adding visual analysis tools to the two maps, we have created an insightful way of building intuition about the nature structure and asymmetric properties of such kinds of data.

Experimental results using three real-world data sets confirmed that the proposed dynamic learning method is useful to understand asymmetric proximity data. We can apply this method to other data sets such as clinical data to have better understanding on causes and effects. As future work, we can add more analysis tools on the maps.

Acknowledgments

This work is based on our previous conference presentation [21].

References

[1] T. Cox, M. Cox, Multidimensional Scaling, 2nd ed., Chapman & Hall, London, 2001.
[2] I.T. Jolliffe, Principal Component Analysis, Springer-Verlag, New York, NY, 1986.
[3] J.B. Tenenbaum, V. de Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.
[4] L. Saul, S.T. Roweis, Think globally, fit locally: Unsupervised learning of low dimensional manifolds, Journal of Machine Learning Research 4 (2003) 119–155.
[5] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Computation 15 (2003) 1373–1396.
[6] H. Choi, S. Choi, Robust kernel Isomap, Pattern Recognition 40 (3) (2007) 853–862.
[7] T. Kohonen, Self-Organizing Maps, Springer, New York, NY, 1995.
[8] W. Cui, H. Zhou, H. Qu, P.C. Wong, X. Li, Geometry-based edge clustering for graph visualization, in: Proceedings of the IEEE Symposium on Information Visualization (InfoVis), 2008.
[9] S. Ingram, T. Munzner, V. Irvine, M. Tory, S. Bergner, T. Moller, DimStiller: workflows for dimensional analysis and reduction, in: Proceedings of IEEE Symposium on Visual Analytics Science and Technology (VAST), 2010.
[10] V. Roth, J. Laub, M. Kawanabe, J.M. Buhmann, Optimal cluster preserving embedding of nonmetric proximity data, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (12) (2003) 1540–1551.
[11] B. Zielman, W.J. Heiser, Models for asymmetric proximities, British Journal of Mathematical and Statistical Psychology 49 (1) (1996) 127–146.
[12] N.J. van Eck, L. Waltman, R. Dekker, J. van den Berg, A comparison of two techniques for bibliometric mapping: multidimensional scaling and VOS, Journal of the American Society for Information Science and Technology 61 (12) (2010) 2405–2416.
[13] E.Z. Rothkopf, A measure of stimulus similarity and errors in some paired-associate learning tasks, Journal of Experimental Psychology 53 (1957) 94–101.
[14] I. Borg, P.J.F. Groenen, Modern Multidimensional Scaling: Theory and Applications, 2nd ed., Springer, 2010.
[15] R.A. Harshman, P.E. Green, Y. Wind, M.E. Lundy, A model for the analysis of asymmetric data in marketing research, Marketing Science 1 (2) (1982) 205–242.
[16] M. Martin-Merino, A. Munoz, Visualizing asymmetric proximities with SOM and MDS models, Neurocomputing 63 (2005) 171–192.
[17] D.G. Weeks, P.M. Bentler, Restricted multidimensional scaling models for asymmetric proximities, Psychometrika 47 (2) (1982) 201–208.
[18] B. Jones, Observations and experiments on causation of threat displays of the great tit (Parus major), Animal Behavior Monographs 1 (1968) 75–158.
[19] H. Choi, S. Choi, Kernel Isomap, Electronics Letters 40 (25) (2004) 1612–1613.
[20] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society. Series B (Methodological) 39 (1) (1977) 1–38.
[21] H. Choi, Dynamic learning for visual representation of asymmetric proximity, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Seoul, Korea, 2012.

Heeyoul Choi received his B.S. and M.S. degrees in Computer Science and Engineering from Pohang University of Science and Technology, Korea, in 2002 and 2005, respectively, and the Ph.D. degree in Computer Science and Engineering from Texas A&M University, Texas, in 2010. He was a post-doctoral researcher in Psychological and Brain Science at Indiana University, Indiana, from 2010 to 2011. He is currently a research staff member in Samsung Advanced Institute of Technology, Korea. His research interests cover machine learning, pattern recognition, Big Data analysis, information theory, cognitive science, and computational neuroscience.