Nonmetric Multidimensional Scaling with Neural Networks

Michiel C. van Wezel, Walter A. Kosters, Peter van der Putten, and Joost N. Kok

Leiden Institute of Advanced Computer Science (LIACS)
Universiteit Leiden
P.O. Box 9512, 2300 RA Leiden, The Netherlands.
{michiel,kosters,putten,joost}@liacs.nl

Abstract In this paper we present a neural network for nonmetric multidimensional scaling. In our approach, the monotone transformation that is a part of every nonmetric scaling algorithm is performed by a special feedforward neural network with a modified backpropagation algorithm. Contrary to traditional methods, we thus explicitly model the monotone transformation by a special purpose neural network. The architecture of the new network and the derivation of the learning rule are given, as well as some experimental results. The experimental results are positive.

1 Introduction

This paper concerns visualization of multidimensional data using neural nonmetric multidimensional scaling. Generally stated, multidimensional scaling (abbreviated MDS) is a collection of techniques for embedding dissimilarity data in a space with a chosen dimensionality. The embedding is often used for the purpose of data visualization and exploratory data analysis. In this sense, MDS is a competitor for other data visualization techniques, such as a Kohonen neural network and principal component analysis. Traditional MDS techniques are subdivided into metric MDS, where the dissimilarities between objects are assumed to be proportional to Euclidean distances, and nonmetric MDS, where the dissimilarities are only assumed to be monotonically related to Euclidean distances. In the case where the dissimilarities represent, e.g., distances between the capitals of the European countries, the Euclidean distance assumption is realistic. However, in the case of dissimilarities between soda brands that have been reported by a panel of test persons, nonmetric MDS seems more appropriate.

Although traditional multidimensional scaling is a well established pattern recognition technique (see, e.g., [1,2,10,7]), to our knowledge this subject has not received a lot of attention from researchers in the field of neural networks. In particular, all publications concerning neural networks for multidimensional scaling that are known to us concern metric multidimensional scaling. In [9] a simple neural network is given for metric multidimensional scaling. This neural network merely performs a gradient descent on the cost function, which carries the risk of getting stuck in local minima in the error function. To prevent this, in [4,5] Klöck and Buhmann apply annealing methods from statistical mechanics to the metric MDS problem. No neural algorithms have been applied to nonmetric MDS so far.

This paper is organized as follows. First, an exact problem statement is presented for multidimensional scaling in Section 2. Next, the proposed feedforward neural network for nonmetric multidimensional scaling and the corresponding learning rule are given in Section 3. After that, the results of some experiments are given in Section 4 and finally Section 5 gives conclusions and a summary. This paper does not give an in-depth treatment of multidimensional scaling in general. The interested reader is referred to [2,3,10] for more information on the subject.

2 Multidimensional scaling: problem statement

It is impossible to give a definition of the term multidimensional scaling that covers the general use of this term and is yet formal enough to be precise. Some people use the term for a very broad class of data analysis techniques and problems including cluster analysis and factor analysis (see, e.g., [7]), while others use a more narrow definition, and refer to MDS as a class of techniques used to develop a spatial representation of proximity data. The distinction between a broad and a narrow definition is also made in a useful taxonomy introduced in a paper by Carroll and Arabie [1]. This paper is concerned with MDS in the narrow sense: finding a spatial representation of objects based on proximity data.

Using this narrow definition, there are two "kinds" of MDS: metric MDS and nonmetric MDS. Before we are able to describe these variants we need some terminology, which is often used in MDS literature and is also adopted in this paper.

Dissimilarities: In both metric and nonmetric MDS the starting point is a matrix $\boldsymbol{\delta}$ of dissimilarities, of which an element δ_{ij} denotes the dissimilarity between the objects i and j. The number of objects is denoted by n.

Embedding: An embedding of the objects in Euclidean space. The coordinates of an object i in this embedding are denoted by $\mathbf{x}_i$. The dimensionality of the embedding space is denoted by m, so $\mathbf{x}_i = (x_{i1}, \ldots, x_{im})^T$, where ^{T} denotes transpose.

Distances : The (real) Euclidean distance between objects i and j is denoted by $\hat{d}_{ij}$. The distance between the estimates for the spatial representations $\mathbf{x}_i$ and $\mathbf{x}_j$ of objects i and j is denoted by $d_{ij} = \| \mathbf{x}_i - \mathbf{x}_j \|$, where $\| \, . \, \|$ denotes the Euclidean norm. Collectively the distances are denoted by matrix $\mathbf{d}$.

Disparities: These quantities are used in nonmetric scaling. Disparities $\hat{\boldsymbol{\delta}}$ are as close as possible to distances between the corresponding coordinate estimates d but with the restriction that they are monotonically related to the original dissimilarity data $\boldsymbol{\delta}$.

Using the above concepts, the MDS problem can be accurately described. We differentiate between metric and nonmetric MDS.

Metric multidimensional scaling: As stated earlier, in metric multidimensional scaling, it is assumed that the dissimilarities are *proportional* to Euclidean distances:

$$\delta_{ij} = c\hat{d}_{ij} \tag{1}$$

One way to obtain a spatial representation of dissimilarity data under the above assumption is to minimize the following error function

$$E_{\text{metric\_mds}} = \sum_{ij,i\neq j} (\delta_{ij} - d_{ij})^2. \tag{2}$$

This minimization gives us the correct coordinates up to a scale factor, a translation and a rotation. Torgerson [8] gives a non-iterative procedure for finding these coordinates.

Nonmetric multidimensional scaling: The most fundamental difference between metric MDS and nonmetric MDS is the relaxation of the "distance assumption" (Equation 1) to a "monotonicity assumption". In nonmetric MDS it is assumed that the dissimilarities $\boldsymbol{\delta}$ are *monotonically related* to Euclidean distances:

$$\forall i, j, k, \ell : \hat{d}_{ij} < \hat{d}_{k\ell} \Rightarrow \delta_{ij} < \delta_{k\ell}\ . \tag{3}$$

One can look upon the $\boldsymbol{\delta}$ values as being monotonically transformed distance values: $\delta_{ij} = f(\hat{d}_{ij})$ where $f(.)$ is an unknown strict monotonically increasing function. Examples of such functions include some linear, power and logarithmic functions. Nonmetric MDS algorithms estimate a spatial representation for a given dissimilarity matrix in which the rank order of the distances between the embedded objects agrees with the rank order of the original dissimilarities as much as possible.

Traditionally, in nonmetric MDS one attempts to minimize the following cost function, often called stress or stress formula 1 (due to Kruskal, see [6]), in an iterative fashion:

$$E_{\text{nonmetric\_mds}} = \left[\sum_{ij,i\neq j} (\hat{\delta}_{ij} - d_{ij})^2 / \sum_{ij,i\neq j} d_{ij}^2 \right]^{1/2}. \tag{4}$$

Often, the inter-pattern distances are kept normalized ($\sum_{ij,i\neq j} d_{ij}^2 = 1$), in which case the error function reduces to

$$E^{nm} = \sum_{ij,i\neq j} (\hat{\delta}_{ij} - d_{ij})^2, \tag{5}$$

which is computationally simpler. The disparities $\hat{\boldsymbol{\delta}}$ must be re-estimated in each iteration in a so-called nonmetric phase. As stated before, the disparities are chosen to resemble the distances of the current embedding as close as possible subject to a monotonicity constraint:

$$\forall i, j, k, \ell : \delta_{ij} < \delta_{k\ell} \Rightarrow \hat{\delta}_{ij} \leq \hat{\delta}_{k\ell}. \tag{6}$$

This can be achieved by, e.g., setting the disparity $\hat{\delta}_{ij}$ equal to the m-th largest distance $d_{k\ell}$ if the dissimilarity $\delta_{k\ell}$ is the m-th largest dissimilarity. One ends up with an iterative algorithm that is schematically depicted in Figure 1.

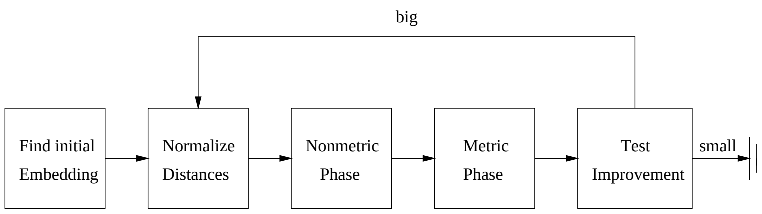

Figure 1. Schematic representation of nonmetric mds algorithm

By minimizing stress (Equation 4) a spatial representation is found in which the distances between the embedded objects resemble the corresponding disparities as close as possible, while the disparities themselves are monotonically related to the original data. The dissimilarities are thus transformed by the monotone transformation in a way that improves the final fit to the data.

3 A neural network approach to nonmetric MDS

In this section we present a neural method for nonmetric MDS. The cost function that we minimize in this section is given in Equation 5. This means that we have to normalize the distances $\mathbf{d}$ and the embedding coordinates $\mathbf{x}$ after each iteration.

Metric phase: Recall that nonmetric MDS algorithms usually perform a metric phase and a nonmetric phase in each iteration. In the metric phase, the positions $\mathbf{x}$ of the embedded objects are altered so that the distances $\mathbf{d}$ resemble the disparities $\hat{\boldsymbol{\delta}}$ as well as possible. Thus Equation 5 is minimized as a function of the vectors $\mathbf{x}$. The product rule and the chain rule are needed to compute the partial derivatives (x_{ab} denotes the b-th component of $\mathbf{x}_a$):

$$\frac{\partial E^{nm}}{\partial x_{ab}} = \sum_{ij} \frac{\partial(\hat{\delta}_{ij} - d_{ij})^2}{\partial x_{ab}}. \tag{7}$$

Now

$$\frac{\partial(\hat{\delta}_{ij} - d_{ij})^2}{\partial x_{ab}} = -2(\hat{\delta}_{ij} - d_{ij})\frac{\partial d_{ij}}{\partial x_{ab}}, \tag{8}$$

$$\frac{\partial d_{ij}}{\partial x_{ab}} = \begin{cases} (x_{ab} - x_{jb})/d_{aj} & \text{if } i = a, \\ (x_{ab} - x_{ib})/d_{ia} & \text{if } j = a, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

Nonmetric phase: In the nonmetric phase values for the disparity matrix $\hat{\boldsymbol{\delta}}$ are sought that are monotonically related to the original data (the dissimilarities $\boldsymbol{\delta}$), yet as close as possible to the distances $\mathbf{d}$. For accomplishing this task, we designed and implemented a special neural network that is only capable of modeling monotone transformations, and attempted to use this network to model the transformation $f(\boldsymbol{\delta}) = \hat{\boldsymbol{\delta}}$ explicitly. We will refer to this network type as a *mono-nn* in the remainder.

The reason for this explicit modeling using a neural network is threefold. First, explicit modeling gives the user the opportunity to visualize and examine the total monotone transformation, because the neural network performs interpolation. The second reason is that we wanted to investigate the value of neural network techniques in this problem area. The third and most important reason is that the use of interpolation in the nonmetric phase can potentially speed up the nonmetric phase considerably in the case where the dissimilarity matrix is of substantial size. The reason for this is that in a traditional nonmetric phase, one is required to perform a sorting operation of the inter-pattern distances $\mathbf{d}$. Since the fastest sorting algorithm around is the quicksort algorithm and this algorithm has a quadratic worst-case time complexity, the amount of time needed for a traditional nonmetric phase is $\mathcal{O}(n^4)$. When a mono-nn is used, the sorting step is superfluous. The mono-nn can be trained using only a subset of all available data, and it can be used as an interpolator for the remaining data.

This network takes the dissimilarities $\boldsymbol{\delta}$ as inputs and generates the disparities $\hat{\boldsymbol{\delta}}$ as outputs. It uses the distances $\mathbf{d}$ as targets. The network has one hidden layer with non-linear (hyperbolic tangent) transfer functions. The output unit uses the identity as a transfer function. It will be explained shortly how the monotonicity property is ensured. A network of this kind is depicted in Figure 2. Note that there is a bias unit present in the network. This unit is clamped to activation value 1. In the following discussion, a, b, c index units in the input,

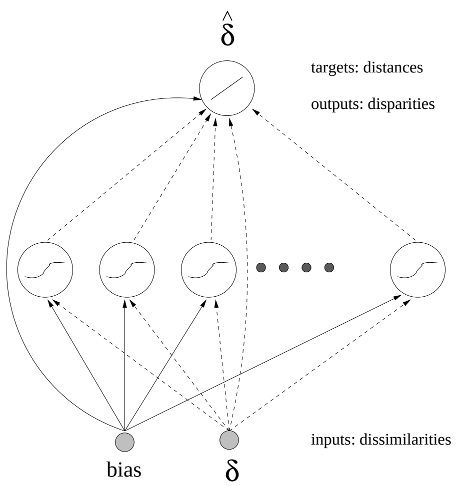

Figure 2. Monotone neural network for nonmetric phase in MDS.

hidden and output layer respectively; $y_{k|ij}$ denotes the output of unit k (input, hidden or output) when dissimilarity δ_{ij} [1] is offered at the input; $g_k(.)$ denotes the transfer function of unit k. The cost function for this network is (c denotes

[1] We standardized all inputs and targets at the beginning of the nonmetric phase in each iteration.

the only output unit):

$$E^{nn} = \sum_{ij, i \neq j} (y_{c|ij} - d_{ij})^2 = \sum_{ij, i \neq j} E^{nn|ij}. \tag{10}$$

Furthermore, $w_{k\ell}$ denotes the connection weight of the connection that runs from unit k to unit ℓ and $in_{k|ij}$ denotes the net-input to unit k, so $y_{k|ij} = g_k(in_{k|ij})$. The bias for unit k is denoted by θ_k. Since the individual transfer functions are monotonically increasing, the monotonicity constraint is always satisfied if we use squared network parameters (weights) in the computation for the weighted net-inputs for all weights except the biases:

$$in_{c|ij} = \sum_b w_{bc}^2 y_{b|ij} + w_{ac}^2 y_{a|ij} + \theta_c \qquad in_{b|ij} = \sum_a w_{ab}^2 y_{a|ij} + \theta_b. \tag{11}$$

The connections for which the *squared* weights are used in the computations are represented by the dashed arrows in Figure 2. Finally, for convenience, we define a matrix of network outputs $\mathbf{Y}$ with $Y_{ij} = y_{c|ij}$.

For a gradient based training procedure, the partial derivatives of the cost function with respect to the network weights have to be computed. Here we use k and ℓ, which can denote any unit, as indexes. We have:

$$\frac{\partial E^{nn|ij}}{\partial w_{k\ell}} = \frac{\partial E^{nn|ij}}{\partial in_{\ell|ij}} \frac{\partial in_{\ell|ij}}{\partial w_{k\ell}} \qquad \frac{\partial E^{nn|ij}}{\partial \theta_\ell} = \frac{\partial E^{nn|ij}}{\partial in_{\ell|ij}} \frac{\partial in_{\ell|ij}}{\partial \theta_\ell}. \tag{12}$$

If we denote

$$\Delta_{\ell|ij} = \frac{\partial E^{nn|ij}}{\partial in_{\ell|ij}} \tag{13}$$

and note that

$$\frac{\partial in_{\ell|ij}}{\partial w_{k\ell}} = 2 w_{k\ell} y_{k|ij} \qquad \frac{\partial in_{\ell|ij}}{\partial \theta_\ell} = 1 \tag{14}$$

we obtain

$$\frac{\partial E^{nn|ij}}{\partial w_{k\ell}} = \Delta_{\ell|ij} 2 w_{k\ell} y_{k|ij} \qquad \frac{\partial E^{nn|ij}}{\partial \theta_\ell} = \Delta_{\ell|ij}. \tag{15}$$

For the output unit c, with identity activation, the Δ values are easily computed:

$$\Delta_{c|ij} = \frac{\partial E^{nn|ij}}{\partial in_{c|ij}} = 2(a_{c|ij} - d_{ij}). \tag{16}$$

For the hidden units, with *tanh* activation, the Δ values are

$$\Delta_{b|ij} = \frac{\partial E^{nn|ij}}{\partial in_{b|ij}} = \Delta_{c|ij} w_{bc}^2 (1 - g_b(a_{b|ij})^2). \tag{17}$$

Given these formulae, the training algorithm for a monotone neural network is as follows. First, a random object pair i, j is chosen from all possible object pairs. The dissimilarity δ_{ij} for this object pair is used as input for the network. The corresponding output is computed, and the Δ values for all units are computed.

Then, the partial derivatives are computed using Equation 15. These partial derivatives are then used to update the network weights, either in batch mode or in pattern mode.

The total nonmetric MDS procedure is greatly accelerated by using a good initial embedding. In the experiments reported in the next section, we used the solution obtained by the metric MDS as the initial embedding. The complete algorithm for nonmetric MDS with a neural network is given in pseudo-code in Figure 3.

initialize the embedding coordinates $\mathbf{x}$ of all objects randomly.
initialize the disparities $\hat{\boldsymbol{\delta}}$ randomly.
(optional) **perform** metric MDS to obtain an initial embedding.
initialize weights of monotone neural network randomly.
for cycle = 1 **to** number of cycles **do**
  {
    normalize distances and embedding coordinates.
    /* nonmetric phase */
    Train the monotone neural network using M randomly selected
    pattern pairs and gradients via Equation 15.
    Set the disparities $\hat{\boldsymbol{\delta}} = \mathbf{Y}$.
    /* metric phase */
    Find new embedding coordinates using $\hat{\boldsymbol{\delta}}$ and Equation 7.
  }

Figure 3. Algorithm (in pseudo-code) for nonmetric MDS using a gradient based algorithm.

4 Experiments and results

In this section we describe the experiments we performed with our implementation and the results we obtained. In all experiments we use an embedding dimensionality of 2.

Experiments with the monotonic neural network: We started by testing our special purpose neural network. In order to do this, we created three artificial datasets containing clear non-linearities. The first dataset consisted of 105 points. The x-values were randomly distributed in the $[-2.5 : 2.5]$ interval. The corresponding y values were monotonically related to the x values. This dataset contained a general trend (resembling a tanh(.) function) and some deviations from this trend, but never violating the monotonicity property. The second dataset contained 100 random x values in the $[-2 : 2]$ interval. The y values were uniformly distributed with small standard deviation around the function $0.6 \tanh(5x + 4) + 0.9 \tanh(7x - 1)$. This dataset contains a trend that is monotonically increasing, but the noise makes the total dataset non-monotonically increasing. The third dataset was generated in a similar way around $-0.6 \tanh(5x + 4) + 0.9 \tanh(7x - 1)$. This function is not monotonically increasing. All datasets and the fits obtained by the special networks are shown in Figure 4. Observe that the major nonlinearities are fitted well by our network, but in the case of the first dataset the network was not able to model the subtle nonlinearities. The fit for the third dataset illustrates the monotonicity property of the network.

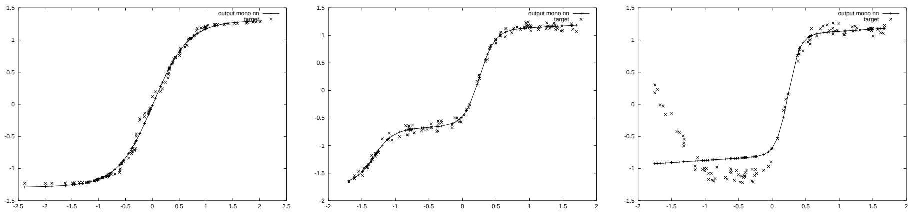

Figure 4. Curve fitted by mono-nn and original data (scattered crosses) for artificial dataset one (left), two (center) and three (right). See text for more explanation.

Experiment with MDS on artificial data Next, we applied our nonmetric scaling method to an artificial problem. For this problem, we created a dataset, the `square` dataset, consisting of 16 points arranged in a square in $[0:1]^2$. The distances between all points were computed and transformed by a nonlinear transformation: $0.5(\tanh(5(x-0.5))+1)$. This yielded the dissimilarities $\boldsymbol{\delta}$ that we used as a starting point for our algorithm.

Consider the upper three graphs of Figure 5. The original data that were used to generate the distances (which were in turn transformed to dissimilarities) are shown in the left graph of this figure. The embedding found by a metric MDS algorithm is shown in the middle graph. The right graph depicts the embedding found by our neural algorithm for nonmetric scaling. It is clearly visible that the latter embedding resembles the original data better. Now consider the lower graphs in Figure 5. The leftmost graph shows the transformation that was applied to the inter-pattern distances to create the dissimilarities. The middle graph shows the function that was learned by the mono-nn together with the target data (the inter-pattern distances) in the final iteration, the right graph shows the inverse of the mono-nn function, which clearly resembles the original transformation.

Experiment with MDS on vocational preference data In this experiment we analyzed the small "vocational preferences" dataset. We adopted this dataset from the MDS textbook of Davison [2] who, in turn, adopted it from psychometric literature. The dataset is shown in Table 1. It contains dissimilarities between six 'vocational preference inventory' scales in a sample of 1234 men. In psychological literature these six occupational types are often displayed in a hexagon (see the left side of Figure 6). It is claimed that the vocational interests of persons in occupational types adjacent to each other in the hexagon are more similar than the interests of people with occupational types more distant from each other.

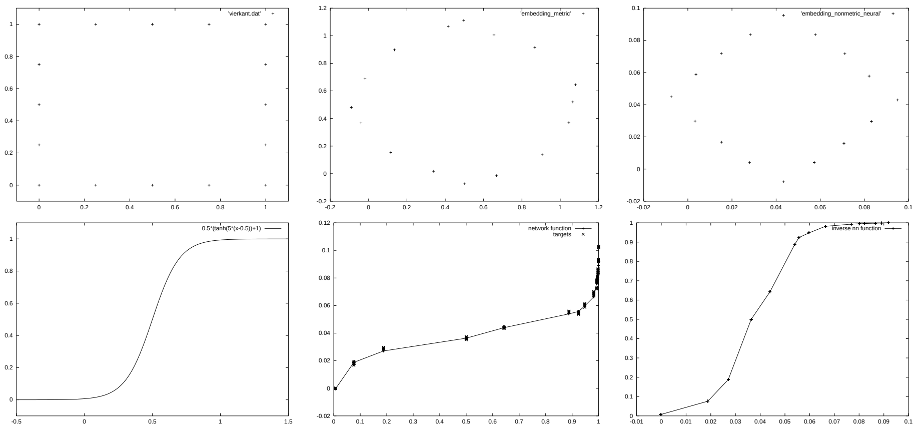

Figure 5. Upper graphs: Original data, metric MDS embedding and neural nonmetric embedding for the `square` dataset. Lower graphs: Original transformation, mono-nn function and inverse of nn-function for the `square` dataset.

In order to verify this hypothesis, nonmetric scaling was applied to the vocational preferences dataset. We applied our neural nonmetric scaling algorithm to these data and ended up with the embedding shown in the middle of Figure 6. In this embedding, the data points representing the six occupational types have indeed settled themselves in a roughly hexagonal shape.

	R	I	A	S	E	C
Realistic	0.0000	1.0392	1.2961	1.2570	1.1832	1.1314
Investigative	1.0392	0.0000	1.1489	1.1832	1.2961	1.2961
Artistic	1.2961	1.1489	0.0000	1.0770	1.1402	1.3342
Social	1.2570	1.1832	1.0770	0.0000	0.9592	1.1136
Enterprising	1.1832	1.2961	1.1402	0.9592	0.0000	0.8000
Conventional	1.1314	1.2961	1.3342	1.1136	0.8000	0.0000

Table 1. Vocational preference dissimilarity data.

5 Summary and conclusions

Multidimensional scaling is a data projection method that uses dissimilarity data as a starting point. This approach makes it different from more often used data projection methods such as principal component analysis. MDS searches for an optimal embedding given a matrix of dissimilarities. There are two types of MDS: metric MDS and nonmetric MDS. This paper focuses on the nonmetric MDS, where the original data are monotonically transformed during the embedding process.

A new neural architecture is proposed for performing the nonmetric phase that is immanent in every nonmetric scaling algorithm. The corresponding learning rule was derived and the network (called mono-nn) was shown to perform well for various test problems. The new scaling method that uses the new network type was tested on a dataset with dissimilarities that were obtained by nonlinearly transforming a matrix of distances. The mono-nn was able to correctly

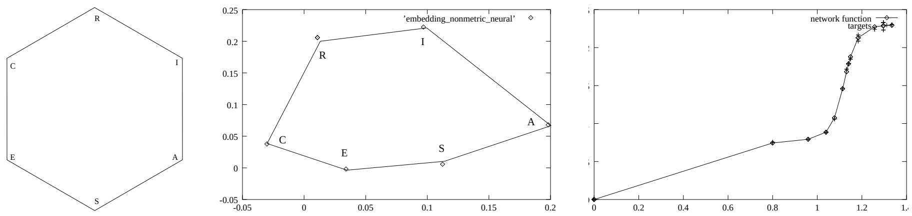

Figure 6. Theoretical hexagon for vocational preferences (left). Embedding found by neural MDS (middle). Transformation implemented by mono-nn (right).

model the inverse transformation and the scaling method yielded a good embedding. A real-life dataset from psychometrics was analyzed and the embedding confirmed a hypothesis concerning vocational preferences.

The added value of our scaling method is simple: it explicitly models the nonlinear monotone transformation of the original dissimilarity data. This explicit modeling may give the data analyst more insight into the transformation and it gives the ability to interpolate, contrary to traditional scaling methods. Interpolation gives a potential speed up for large dissimilarity matrices. It must be remarked however, that traditional nonmetric scaling methods are generally faster than our method for small data sets, because of the greater simplicity of the nonmetric phase.

References

1. J.D. Carroll and P. Arabie. Multidimensional scaling. In M.R. Rosenzweig and L.W. Porter, editors, *Annual Review of Psychology*, pages 607–649. 1980.
2. M. L. Davison. *Multidimensional Scaling.* John Wiley and Sons, New York, 1983.
3. J. de Leeuw and W. Heiser. Theory of multidimensional scaling. In P. R. Krishnaiah and L. N. Kanal, editors, *Handbook of Statistics*, volume 2, pages 285–316. North-Holland, 1982.
4. H. Klöck and J.M. Buhmann. Multidimensional scaling by deterministic annealing. In M. Pellilo and E.R. Hancock, editors, *Proceedings of the EMMCVPR'97*, pages 245–260, Venice, 1997.
5. H. Klöck and J.M. Buhmann. Data visualization by multidimensional scaling: A deterministic annealing approach. *Pattern Recognition*, 33(4):651–669, 1999.
6. J. B. Kruskal. Multidimensional scaling by optimizing goodness-of-fit to a nonmetric hypothesis. *Psychometrika*, 29:1–29, 1964.
7. B. D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, 1996.
8. W. S. Torgerson. *Theory and Methods of Scaling.* John Wiley and Sons, New York, 1958.
9. M. C. van Wezel, J. N. Kok, and W. A. Kosters. Two neural network methods for multidimensional scaling. In *European Symposium on Artificial Neural Networks (ESANN'97)*, pages 97–102, Brussels, 1997. D facto.
10. M. Wish and D. Carroll. multidimensional scaling and its applications. In P. R. Krishnaiah and L. N. Kanal, editors, *Handbook of Statistics*, volume 2, pages 317–345. North-Holland, 1982.