II

Publication II

Timo Similä (2005). Self-organizing map learning nonlinearly embedded manifolds, *Information Visualization* **4**(1):22–31.
© 2005 Palgrave Macmillan. Reproduced with permission of Palgrave Macmillan.

Information Visualization (2005) 4, 22–31
© 2005 Palgrave Macmillan Ltd. All rights reserved 1473-8716 $30.00
www.palgrave-journals.com/ivs

Self-organizing map learning nonlinearly embedded manifolds

Timo Similä[1]

[1]*Neural Networks Research Centre, Helsinki University of Technology, PO Box 5400, FI-02015 HUT, Finland*

Correspondence: Timo Similä, Neural Networks Research Centre, Helsinki University of Technology, E-mail: timo.simila@hut.fi

Abstract

One of the main tasks in exploratory data analysis is to create an appropriate representation for complex data. In this paper, the problem of creating a representation for observations lying on a low-dimensional manifold embedded in high-dimensional coordinates is considered. We propose a modification of the Self-organizing map (SOM) algorithm that is able to learn the manifold structure in the high-dimensional observation coordinates. Any manifold learning algorithm may be incorporated to the proposed training strategy to guide the map onto the manifold surface instead of becoming trapped in local minima. In this paper, the Locally linear embedding algorithm is adopted. We use the proposed method successfully on several data sets with manifold geometry including an illustrative example of a surface as well as image data. We also show with other experiments that the advantage of the method over the basic SOM is restricted to this specific type of data.
Information Visualization (2005) **4,** 22–31. doi:10.1057/palgrave.ivs.9500088

Keywords: Self-organizing map; manifold learning; dimensionality reduction; visualization

Introduction

Exploratory data analysis aims to analyze high-dimensional observations of an unknown process by representations that are both understandable and preserve essential properties of the process. In the absence of detailed prior knowledge, such representations must be discovered automatically. The Self-organizing map (SOM)[1] is an algorithm that is able to unsupervisedly learn and visualize characteristic features or other abstractions from the data. It produces nonlinear dimensionality reduction by fitting a low-dimensional topologically ordered grid of nodes in high-dimensional observation coordinates. This paper presents a modification of the basic SOM algorithm, named the M-SOM. The prefix M denotes a manifold and promotes the applicability of the method to a certain class of data sets.

A manifold is by definition a topological space that is locally Euclidean. To illustrate the idea, consider the surface of the Earth, which looks on the small scales flat. However, unlike the ancient belief, we know that it is globally round shaped. Manifold learning methods study intrinsically low-dimensional data lying in high-dimensional observation coordinates aiming to discover the low-dimensional representation. For linearly embedded manifolds, Principal component analysis (PCA)[2] is guaranteed to discover the low-dimensional representation. Recently, there has been increasing interest in learning nonlinearly embedded manifolds, and algorithms such as the Isomap,[3] Locally linear embedding (LLE),[4] Hessian LLE,[5] Laplacian eigenmaps[6] and Curvilinear distance analysis[7] have been

Received: 20 August 2004
Revised: 19 January 2005
Accepted: 24 January 2005
Online publication date: 3 March 2005

presented. They form a neighborhood-preserving projection that projects a point $\boldsymbol{x}$ from the high-dimensional observation coordinates onto the point $\boldsymbol{y}$ in the internal coordinates on the manifold.

Although the SOM and nonlinear manifold learning methods have a similar purpose, the SOM fails in some cases to model a visually obvious structure in the data, as addressed by Tenenbaum.[8] The iterative optimization may get stuck at locally optimal solutions, when nonlinear structure in the data cannot simply be regarded as a perturbation from a linear approximation. Figure 1 illustrates a situation, where three-dimensional data (Figure 1B) are sampled from an intrinsically two-dimensional S-curve manifold (Figure 1A). Depending on the initial conditions, but very likely, the SOM does not model the data appropriately (Figure 1F). A satisfactory representation is obtained by using the proposed M-SOM algorithm, which may be summarized as follows.

1. Run the LLE algorithm to learn the internal coordinates on the manifold (Figure 1C).
2. Train a map in such a way that similarities are defined in the internal coordinates, but the nodes are adapted in both the internal and observation coordinates (Figure 1D and E).
3. (Optional) Continue the training in the observation coordinates using the basic SOM algorithm with a narrow neighborhood function to fine-tune the positions of the nodes.

We have adopted the LLE algorithm in this study, but, in general, any manifold learning method may be used in place of it.

Visualization is the purpose of this paper. Dimensionality reduction is one of the key aspects of this purpose, but it is not enough for providing a complete picture. Manifold learning methods may provide a successful low-dimensional representation, but they do not summarize the data in any other way. Besides the need to comprehend a possibly large data set, the observations may be highly overlapping in the low-dimensional coordinates, which hampers the analysis. The Relational perspective map[9] is one of the few methods that take the problem of overlapping projections into account at some extent. On the other hand, the M-SOM shares the same advantages that the SOM has in summarizing massive data sets. The grid, which takes place in the high-dimensional observation coordinates, can be visualized by a low-dimensional uniform lattice with a coloring of the nodes. The nodes provide a clustering, which solves both the problem of overlapping and high volume of the data.

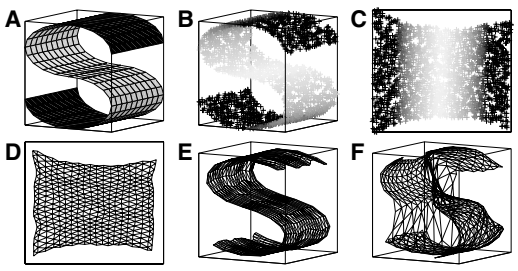

Figure 1 The problem of manifold learning for three-dimensional data (B) sampled from an intrinsically two-dimensional S-curve manifold (A). The LLE algorithm discovers the projection onto the internal coordinates on the manifold (C). The map trained using the proposed M-SOM algorithm learns similarities in the internal coordinates (D) and, thus, allows a successful representation of the manifold in the observation coordinates (E). The basic SOM algorithm fails in this task (F).

In practice, the M-SOM becomes suitable whenever the data are known to form a manifold in the high-dimensional observation coordinates. Applications may be found from many areas, such as computer vision and monitoring of dynamical processes. Consider, for example, grayscale images of an object taken under fixed lighting conditions with a moving camera. The intrinsic dimensionality of a set of these images equals the number of degrees of freedom of the camera instead of the number of pixels in an image.[10] The M-SOM produces a clustering of such images, where the neighborhood relations and prototype images for the clusters are visible. As an example, Figure 2B shows a clustering of images of a toy object taken at varying views of angle. Another example is a dynamic process that traverses certain trajectories between the states forming a manifold in the process variable space.

The M-SOM may also be applied in information retrieval. The 'curse of dimensionality' states that more features do not necessarily imply better retrieval accuracy.[11] Sometimes the use of only a few visually salient features of a large set of them implies more accurate results and makes the query process simple and fast.[12] Similarities in the M-SOM algorithm are defined in the internal coordinates. When a query has been made to the M-SOM, the retrieval will be performed in these coordinates and their dimensionality may be substantially lower than the dimensionality of the observation coordinates.

LLE algorithm

Suppose that hidden data are sampled randomly in a domain $Y \subset \mathbb{R}^d$ and then smoothly embedded by f: $Y \to \mathbb{R}^D$ for some $D > d$, that is, a hidden data point $\boldsymbol{y}$ becomes an observed data point by mapping $\boldsymbol{x} = f(\boldsymbol{y})$. Manifold learning can be formulated as the problem of inverting the unknown mapping f for a given set of observations. Without any restrictions of f the problem is ill-posed and usually f is assumed to be a linear, isometric or conformal mapping.[13] In the starting point of the M-SOM algorithm the manifold learning problem is assumed to be solved somehow and a data point may be written in the paired form $(\boldsymbol{y}, \boldsymbol{x})$.

In this study, the manifold learning problem is chosen to be solved by using the LLE algorithm.[4] This choice is justified by its computational efficiency and applicability to a rather broad range of manifolds.[13] Local approach to the problem is also tractable in the sense that it retains the cluster structure of the data,[6] which is commonly one of the properties of interest. On the other hand, solving the manifold problem itself is not the main contribution of this paper. The aim is to provide an accurate visualization of a set of data points lying on or near a manifold in the high-dimensional observation coordinates.

The LLE algorithm consists of three steps: nearest neighbor search, computation of the weights that best reconstruct each observation from its neighbors, and computation of the embedding of the observations reconstructed using the weights.

1. Find K nearest neighbors of each D-dimensional vector $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ according to the Euclidean distance.
2. Compute weights w_{ij} such that the reconstruction error $\sum_i \|\boldsymbol{x}_i - \sum_j w_{ij}\boldsymbol{x}_j\|^2$ is minimized subject to two constraints. The sparseness constraint forces $w_{ij}=0$ if $\boldsymbol{x}_j$ does not belong to the K nearest neighbors of $\boldsymbol{x}_i$. The invariance constraint forces $\sum_j w_{ij}=1$.
3. Compute a representation of the data by finding d-dimensional vectors $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N$, which minimize the embedding cost $\sum_i \|\boldsymbol{y}_i - \sum_j w_{ij}\boldsymbol{y}_j\|^2$ subject to two constraints. The zero-mean constraint forces $\sum_i \boldsymbol{y}_i = 0$ and the unit covariance constraint forces $\sum_i \boldsymbol{y}_i \boldsymbol{y}_i^{\mathrm{T}} = N\boldsymbol{I}$, where $\boldsymbol{I}$ is the $d \times d$ identity matrix.

The optimization is not prone to local minima and may be efficiently implemented using sparse matrix computations. The final step is the most computationally expensive. It leads to the computation of bottom $d+1$ eigenvectors of an $N \times N$ sparse, symmetric matrix. The computing scales as $\mathcal{O}(dN^2)$. However, there exist specialized methods for sparse, symmetric problems that reduce the computation. The only free parameters of the algorithm are the size of the neighborhood K and the dimensionality of the manifold d on which the data are expected to lie. Several heuristics have been presented for fixing the values of these parameters.[14]

M-SOM algorithm

An M-SOM consists of nodes, which form a topologically ordered grid and have positions in three different spaces. The location of a node in the space that is traditionally in SOM studies called the input space is decomposed into the position in the internal coordinates $\boldsymbol{m}_j^1$ and another position in the observation coordinates $\boldsymbol{m}_j^2$. The subscript j denotes the index of the node in the uniform lattice in the output space. The output space reveals the adjacency of the nodes in the grid. This discrete structure in the output space remains the same once fixed, whereas the reference vectors $\boldsymbol{m}_j^1$ and $\boldsymbol{m}_j^2$ vary during the training.

The decomposition of the input space into two parts and the following algorithm is similar to the supervised variant of the SOM algorithm previously used in prediction and regression analysis.[15] That study is not about manifold learning, but it aims to model the probability of bankruptcy of an enterprise given past values of a set of financial indicators. We take advantage of that approach by modeling the manifold structure in the data in the observation coordinates given a projection of the same data onto the internal coordinates on the manifold.

The M-SOM algorithm iterates two steps that are the winner node selection and adaptation. At each iteration t, the winner node c for a data point $(\boldsymbol{y}, \boldsymbol{x})$ is the one lying nearest in the internal coordinates according to the Euclidean distance

$$c = \underset{j}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{m}_j^1(t)\|. \tag{1}$$

As the mapping f preserves the local configurations of nearest neighbors, the winner selection rule equals to choosing the node that has the shortest distance between $\boldsymbol{x}$ and $f(\boldsymbol{m}_j^1(t))$ along the manifold $f(Y)$.

The reference vectors of the winner node, as well as the vectors of all other nodes in the neighborhood of it, is modified during the adaptation step according to the update rule

$$\begin{bmatrix} \boldsymbol{m}_j^1(t+1) \\ \boldsymbol{m}_j^2(t+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{m}_j^1(t) \\ \boldsymbol{m}_j^2(t) \end{bmatrix} + h_{cj}(t) \begin{bmatrix} \boldsymbol{y} - \boldsymbol{m}_j^1(t) \\ \boldsymbol{x} - \boldsymbol{m}_j^2(t) \end{bmatrix}. \tag{2}$$

The vectors are updated in the direction where the Euclidean distance to $(\boldsymbol{y}, \boldsymbol{x})$ decreases most rapidly both in the internal and observation coordinates. The neighborhood function $h_{cj}(t)$ controls the adaptation. It is a decreasing function of the distance between the node c and another node j in the lattice. We use a smooth neighborhood function by defining the Gaussian kernel

$$h_{cj}(t) = a(t) \exp\left(-\frac{\|\boldsymbol{p}_c - \boldsymbol{p}_j\|^2}{2s^2(t)} \right), \tag{3}$$

where $\boldsymbol{p}$ denotes the position of a node in the lattice, $a(t)$ is a positive scalar-valued learning-rate factor and $s(t)$ defines the width of the kernel. Both $a(t)$ and $s(t)$ are some monotonically decreasing functions of iteration. Values for these training parameters can be selected in the same way as for the basic SOM algorithm. Details of an optimal selecting have been reviewed by Kohonen.[1]

The operation of the M-SOM algorithm may be described by observing that the map consists of two distinct layers. If the mapping f was known explicitly, the first layer, characterized by $\boldsymbol{m}_j^1$, could already in the beginning of training process be mapped onto the manifold $f(Y)$. However, it would not represent the cluster stucture of the manifold or be in an ordered fashion. The second layer, characterized by $\boldsymbol{m}_j^2$, takes a random place in the observation coordinates. In the convergence $f(\boldsymbol{m}_j^1$ and $\boldsymbol{m}_j^2)$ approach each other. The nodes in the first layer of the M-SOM represent the data in the internal coordinates. This representation is similar to the one obtained by the basic SOM algorithm. The

second layer shows the embedding of the first layer onto the observation coordinates.

The computational complexity of one epoch of training scales linearly with the number of observations N, number of nodes M, dimensionality of the observations D, and dimensionality of the manifold d. The saving in computation in the M-SOM algorithm with respect to the SOM algorithm is $3NM(D-2d)$ floating point operations per an epoch of training. This is due to the decrease of $D-d$ dimensions in the winner node selection and the increase of d dimensions in the adaptation. Memory requirements are low, because basically just the reference vectors of the nodes and the current training data point need to be available. The M-SOM algorithm shares with the SOM algorithm the same principles of convergence and ordering. Readers interested in those issues should refer to the book of Kohonen.[1]

Another approach to learn nonlinearly embedded manifolds would be a variant of the SOM, which operates solely in the observation coordinates, but selects the winner node, for example, based on geodesic distances. Two problems are related to this approach. Firstly, every time the reference vectors of the nodes are updated, geodesic distances to the current training data point need to be calculated. These distances are estimated as shortest paths through the training data on the manifold. We can take advantage of a pre-calculated geodesic distance matrix of the training data and find the nearest neighbors of the nodes. However, this requires calculation of MN Euclidean distances in the D-dimensional space, whereas Eq. (1) requires M Euclidean distances in the d-dimensional space. Secondly, the nodes are not all the time during the training on the manifold, but somewhere in the observation coordinates. A simple example is random initialization of the grid. Geodesic distances are only meaningful along the manifold surface. Note that the M-SOM algorithm can operate effectively with geodesic distances if the Isomap[3] or Curvilinear distance analysis[7] is used in producing the projection onto the internal coordinates, which has Euclidean metric.

Measuring neighborhood preservation

Visualization of high-dimensional observations is one of the most salient purposes of dimensionality reduction methods. The methods differ in what properties of the observations they try to preserve, but as a common aim they try to project similar observations nearby onto the low-dimensional visual display. For instance, in the SOM nearby nodes in the grid represent points in the observation coordinates that are nearby as well. Given two different projection methods, it is beneficial to quantitatively compare, which one is better in the task of preserving the neighborhoods.

In this study, the SOM and M-SOM algorithms are compared using a simple nonparametric measure of whether a set of observations projected nearby onto the grid are actually nearby in the observation coordinates and *vice versa*. Two kinds of errors may occur in the projection. Either the projection introduces new observations to the neighborhood, or observations that are originally neighbors become projected further away. The former kind of error reduces the trustworthiness of the visualization: observations found to be similar in the grid cannot be trusted to be proximate in the observation coordinates. The latter kind of error results from discontinuities in the projection: not all proximities existing in the observations are visible in the grid.

More precisely, trustworthiness of the visualized neighborhoods is defined by[16]

$$M_1(k) = 1 - A(k) \sum_{i=1}^{N} \sum_{\boldsymbol{x}_j \in U_k(\boldsymbol{x}_i)} (r(\boldsymbol{x}_i, \boldsymbol{x}_j) - k), \qquad (4)$$

where $U_k(\boldsymbol{x}_i)$ is the set of observations that belong to the k nearest neighbors of $\boldsymbol{x}_i$ in the grid but not in the observation coordinates. $r(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the rank of $\boldsymbol{x}_j$ in the ascending ordering according to the Euclidean distance from $\boldsymbol{x}_i$ in the observation coordinates. $A(k) = 2/(Nk(2N-3k-1))$ scales the values of $M_1(k)$ between zero and one.

In the same way, preservation of the original neighborhoods is defined by[16]

$$M_2(k) = 1 - A(k) \sum_{i=1}^{N} \sum_{\boldsymbol{x}_j \in V_k(\boldsymbol{x}_i)} (\hat{r}(\boldsymbol{x}_i, \boldsymbol{x}_j) - k), \qquad (5)$$

where $V_k(\boldsymbol{x}_i)$ is the set of observations that belong to the k nearest neighbors of $\boldsymbol{x}_i$ in the observation coordinates but not in the grid. $\hat{r}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the rank of $\boldsymbol{x}_j$ in the ascending ordering according to the distance from $\boldsymbol{x}_i$ in the grid. Again, $A(k)$ scales the values of $M_2(k)$ between zero and one.

Distances between the nodes along the grid may be defined in several ways. One possibility is to measure the shortest paths of the nodes in the grid, where internodal lengths equal to the Euclidean distances between the nodes in the observation coordinates.[17,18] This corresponds to the visual distances on the U-matrix display[19] and takes into account the density of the nodes in the observation coordinates. In the case of the M-SOM, distances between the nodes could be measured as Euclidean distances in the internal coordinates on the manifold, because that is how the dissimilarities are defined during the training process. In this paper, for the sake of simplicity, distances between the nodes along the grid are measured in the same way as in the neighborhood function $h_{cj}(t)$ using the positions denoted by $\boldsymbol{p}$. This choice corresponds to the uniform map lattice traditionally used in SOM studies.

Regardless of the choice how to measure the distances between the nodes, a problem occurs, when several observations are projected onto the same node. In the case of ties in the rank ordering of the distances, the best and worst orderings are calculated and the value is

average of the error measures. In order to avoid too many ties, a grid with a relatively large number of nodes is used.

In many cases, both types of errors cannot be avoided. Then, it is crucial to decide which one of the errors is more harmful. In the earlier studies, trustworthiness has been considered more important and the SOM has been found to outperform many other methods in terms of it.[16–18] On the other hand, the SOM in Figure 1F allows a more trustworthy projection than the M-SOM in Figure 1E, but the M-SOM preserves the original neighborhoods clearly better (not shown). This is due to the fact that no observations are projected onto the SOM nodes that lie away from the manifold and, thus, trustworthiness of the projected observations remains high. The aim in this paper is to produce an accurate visualization of a manifold structure in the data instead of modeling the density of a data cloud. We require a projection to be continuous and consider preservation of the original neighborhoods more important than trustworthiness.

It is worth noting that, in the comparisons, we assign each data point to the nearest node in the observation coordinates. Both of the measures estimates, how accurately the data in the observation coordinates become visualized in the grid. This may not be fair for the M-SOM, which learns similarities in the internal coordinates. Moreover, any difference between the SOM and M-SOM is caused by the adopted manifold learning algorithm.

Experiments with image data sets

The ability of the M-SOM algorithm was tested using two data sets of images. Each observation consists of a single image, which is represented as a point in a vector space, whose dimensionality equals to the number of pixels in the image. The problem of manifold learning is to represent the images in terms of their modes of variability. The data set for the first experiment was created by moving a camera over the half-sphere centered at a toy object,[20] and the modes equal to the pose parameters. The second experiment studies handwritten digits 4, 7 and 9 from the MNIST test data set. This is a more complex case, which does not have so clear manifold geometry. The problem of manifold learning was solved using the LLE algorithm.

The results for the toy object data are presented in Figure 2, where the M-SOM grid is showed in internal and observation coordinates. The projection onto the internal coordinates is structured based on pose configurations of the object, but it is not clear that the LLE components match with the true pose parameters, rotation and elevation. The M-SOM nodes have been arranged along the density of the data in the internal coordinates and the map has learned the embedding. The nodes form meaningful images of the object in the high-dimensional observation coordinates and neighboring nodes represent the object in similar poses. The sharpness of the images

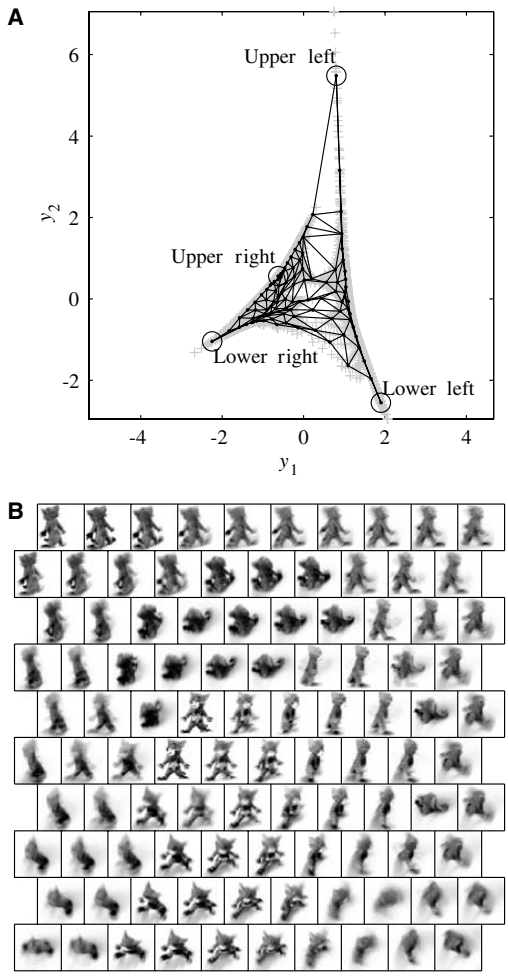

Figure 2 The grid of M-SOM nodes in the internal coordinates (A) and observation coordinates (B). The corner nodes have been marked to allow comparison between the coordinates. The data set consists of total 2500 images of a toy object at 50×50 resolution. The LLE produced the projection using four nearest neighbors.

varies across the lattice indicating that the accuracy of the fitting to the manifold varies by pose configurations.

Similar results for the digits data are presented in Figure 3. The projection onto the internal coordinates produces a more uniform distribution. Only two LLE components are used for visual purposes, but it is clear that at least different digits are projected separately. The M-SOM grid in the observation coordinates shows that the nodes form a smooth representation of the images within a digit and between the digits.

The previous two experiments provided visual evidence that the M-SOM algorithm truly works with the data sets.

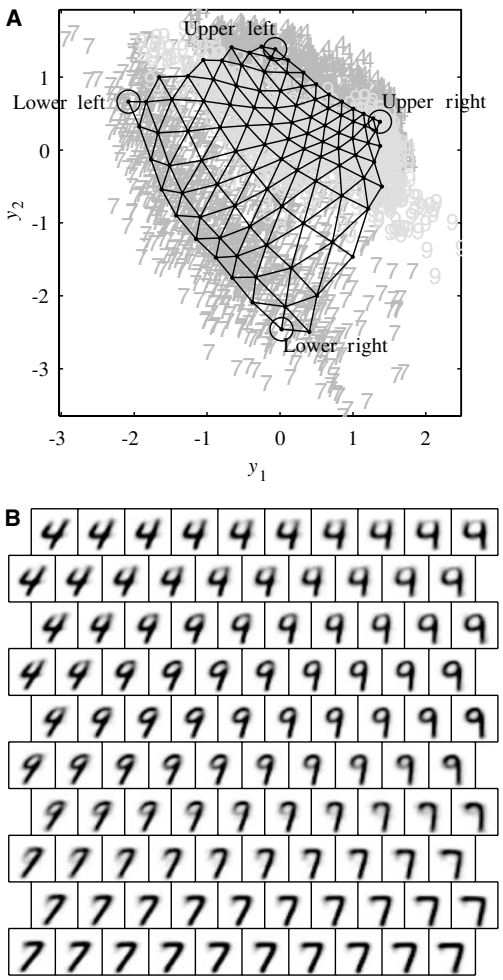

Figure 3 The grid of M-SOM nodes in the internal coordinates (A) and observation coordinates (B). The corner nodes have been marked to allow comparison between the coordinates. The data set consists of total 3019 images representing handwritten digits 4, 7 and 9 at 28×28 resolution. The LLE produced the projection using 18 nearest neighbors.

In order to be convinced that the method competes favorably against the basic SOM algorithm in quantitative terms, the nonparametric measures of the capability to preserve the neighborhoods were calculated. The experiments were repeated using the M-SOM and SOM algorithms with a larger map size than the one shown in Figures 2 and 3, so that the reliability of the error measures would be ensured. Figure 4 shows the measures as a function of the neighborhood size for both data sets.

The SOM algorithm provides consistently a more trustworthy representation of the data in the observation coordinates. This is true for both data sets. On the other hand, the M-SOM preserves the original neighborhoods of all sizes clearly better than the SOM for the handwritten digits data. In the case of the toy object data, this happens only for neighborhoods smaller than about 200 data points. This may be due to the fact that the LLE focuses on preserving rather small neighborhoods of four data points. However, in a visual inspection of the results, the smaller neighborhoods may be argued to be more important than the larger ones.[18]

At least two notices can be made based on the comparison. Firstly, both algorithms acquit themselves well in the task. Especially, the SOM algorithm produces accurate results without any signs of becoming trapped in local minima, as was the situation for the S-curve data shown in Figure 1. Secondly, the algorithms behave oppositely in the dualism of being trustworthy or preserving the original similarities. This helps the analyst to choose between the algorithms depending on which one of the properties he/she considers more important.

Besides differences between the results of the two methods, the time required for the computation varies also. In the SOM algorithm the winner node for a data point is searched in the observation coordinates, whose dimensionality for the toy object data is 2500 and for the digits data 784. In the M-SOM algorithm, this operation is performed in the two-dimensional internal coordinates in both cases. The increase in adaptation cost consists only of these two dimensions. This makes the M-SOM faster than the SOM. The LLE algorithm must, of course, be run before the training process of the M-SOM. The SOM required 1225 s, LLE 19 s and M-SOM 443 s of CPU time for the toy object data on a 2.2 GHz AMD Opteron processor. Running times for the digits data were: SOM 471 s, LLE 281 s and M-SOM 172 s. The training parameters were the same for the SOM and M-SOM algorithms. These running times are only suggestive since our code is not maximally optimized.

Based on the weak trustworthiness of the LLE, a question arose that shortcomings in the LLE may prevent the M-SOM being as trustworthy as the SOM is, see Figure 4. To test this hypothesis, the training process of the M-SOM was continued in the observation coordinates using the basic SOM algorithm with a relatively narrow neighborhood function. This fine-tune training increases indeed the trustworthiness of the map. For neighborhoods smaller than about 200 data points the extended M-SOM and SOM are practically equal in terms of trustworthiness. On the other hand, the capability to preserve small original neighborhoods remains better than in the SOM. This result suggests a new kind of use for the M-SOM algorithm: the accuracy of the SOM algorithm may be increased by initializing the map using the M-SOM algorithm.

Experiments with other data sets

The data sets that were analyzed in the previous section are somewhat similar, namely large sets of grayscale images with very high dimensionality and only a few

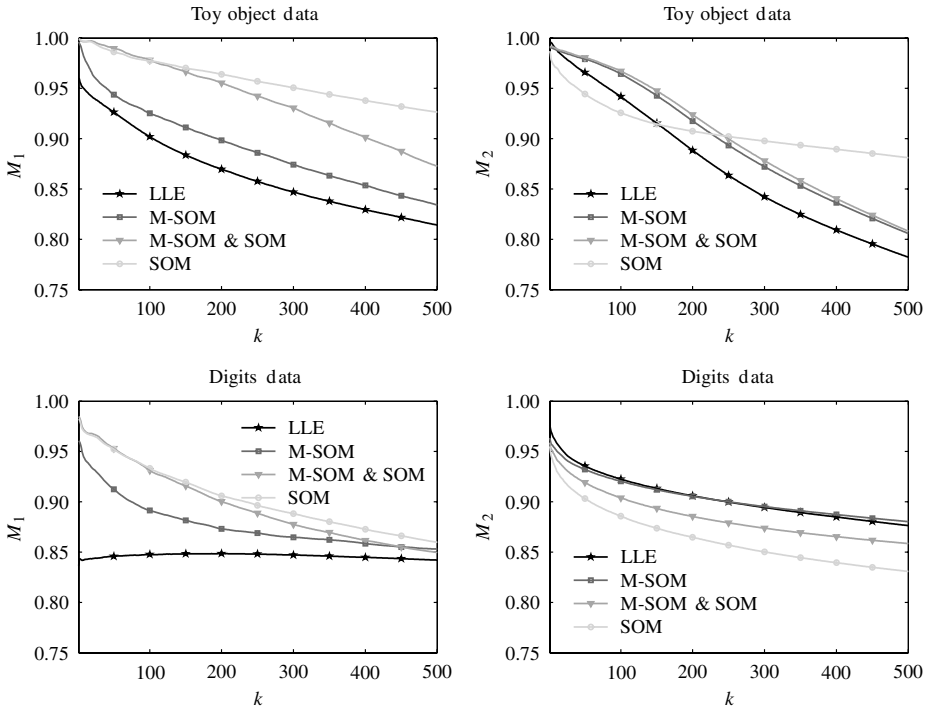

Figure 4 Trustworthiness of the visualized neighborhoods M_1 and capability of the visualizations to preserve the original neighborhoods M_2 for the toy object data and handwritten digits data as a function of the neighborhood size k. The results are for the LLE and 25×25 maps with hexagonal topologies. The worst case with random neighborhoods has approximately $M_1 = 0.5$ and $M_2 = 0.5$.

degrees of freedom. The M-SOM algorithm is designed for this type of data. However, in order to test how the M-SOM performs in terms of M_1 and M_2 with fairly different patterns, we acquired three other data sets. The first set is Glass Identification Database (from UCI Machine Learning Repository) having 214 instances in nine dimensions. The second set is Ionosphere Database (from UCI Machine Learning Repository) having 351 instances in 34 dimensions. The third set includes example data from speech signal and is named as Phonetic data (from LVQ_PAK) having 1962 instances in 20 dimensions. Venna and Kaski[16] have used the measures M_1 and M_2 to study the performance of a group of other projection methods for the same data sets.

We used the LLE to produce projections onto two dimensions using 25, 35 and six neighbors for Glass, Ionosphere and Phonetic data sets, respectively. Then, the M-SOM, SOM and extended M-SOM algorithms were run. Figure 5 shows trustworthiness of the visualized neighborhoods and capability to preserve the original neighborhoods for all three data sets as a function of the neighborhood size.

The methods behave similarly in all cases in terms of trustworthiness. The SOM provides the most trustworthy visualization. The M-SOM is better than the LLE at least for the smaller neighborhoods, but it is still worse than the SOM. Again, the fine-tune training of the M-SOM by the SOM algorithm with a narrow neighborhood function increases the trustworthiness. Hence, the trustworthiness results for these three data sets agree highly with the results for the image data sets in Figure 4.

Capability to preserve the original neighborhoods shows a backside of the issue. It seems that the M-SOM has no consistent advantage over the basic SOM algorithm. The M-SOM preserves the original neighborhoods better than the SOM for the Ionosphere and Phonetic data, but the situation is opposite for the Glass data. However, the differences are so minor that it is hard to make significant conclusions. The results suggest that with these data sets a loss in trustworthiness does not compensate for an increase in preservation of the original neighborhoods as was the case for the image data sets, see Figure 4. The reason is that there is no such a manifold structure in the data that the LLE would capture better than the basic SOM algorithm. The use of the M-SOM algorithm should therefore be limited to the cases in which the data forms a manifold structure.

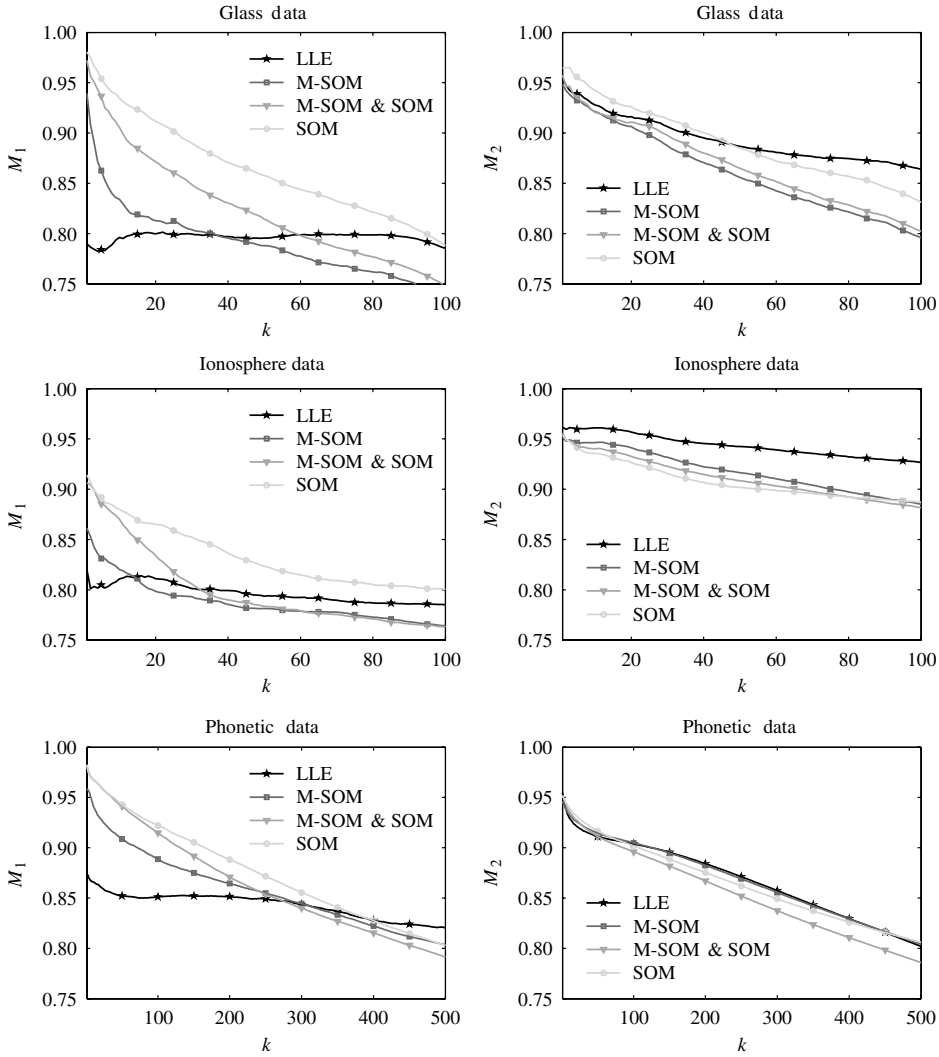

Figure 5 Trustworthiness of the visualized neighborhoods M_1 and capability of the visualizations to preserve the original neighborhoods M_2 for three data sets as a function of the neighborhood size k. The data sets do not have clear manifold geometries. The results are for the LLE and 25×25 maps with hexagonal topologies. The worst case with random neighborhoods has approximately $M_1 = 0.5$ and $M_2 = 0.5$.

Discussion

Visualization is the purpose of developing the M-SOM algorithm. The LLE algorithm, for instance, is able to discover the low-dimensional representation of a manifold, but the reduction of the dimensionality alone is not enough for the visualization purpose. For an efficient visualization, the representation must be summarized somehow. We studied in this paper data sets of images with a manifold structure in which a data point is easily illustrated in the observation coordinates. Instead, if the data set consists of measurements of a complex industrial process forming a manifold in the process variable space, it is not clear that a single observation or a pattern of them becomes illustrated in a satisfactory manner in the low-dimensional internal coordinates. There may be too many, possibly overlapping, observations, and the internal coordinates do not necessarily have any specific interpretation. The M-SOM shares the same advantages that the SOM has in process monitoring using a component plane or some other representation.[21]

The M-SOM does not assume a manifold to be low-dimensional. However, sometimes the observations in the data set are effectively parameterized by a small enough number of continuous variables, say less than four. This allows visualizing the grid in the metric space of the M-SOM revealing the cluster structure of the data on the manifold. For example, the M-SOM grid in the internal coordinates in Figure 1D suggests that data on the S-curve manifold is uniformly distributed. This property comes as a by-product of the M-SOM algorithm, while the cluster structure of the basic SOM has earlier been illustrated using distinct techniques. The visual inspection may have been based on distance matrix techniques, displaying the number of hits in each map node, the use of a generic vector projection or the contraction method.[22,23]

The results of this study suggest that the M-SOM provides a less trustworthy visualization than the SOM. On the other side of the coin, the M-SOM preserves small original neighborhoods better for image data. Depending on the application, one of the properties may be considered more important than the other. When aiming to produce an accurate visualization of a manifold structure in the data, preservation of the original neighborhoods is essential, and the choice of the algorithm favors the M-SOM.

Very promising results were obtained by continuing the training process of the M-SOM in the observation coordinates using the basic SOM algorithm. This improves the trustworthiness of the M-SOM. In the case of image data, small original neighborhoods are still preserved better than in the SOM. The M-SOM algorithm may be thought to guide the map onto the manifold surface without becoming trapped in local minima. Then, the SOM algorithm fine-tunes the locations of the nodes optimally along the density of the data on the manifold. The two stage training strategy may be useful, especially, if there are doubts about the accuracy of the projection of the data points between the two coordinates, in our case, the accuracy of the LLE algorithm. The two stage training strategy, of course, loses the information of the locations of the nodes in the internal coordinates on the manifold. This is a disadvantage, if the manifold is low-dimensional and could be easily inspected.

Experiments with several data sets without clear manifold geometries showed the limits of the M-SOM. Any difference between the SOM and M-SOM is caused by the adopted manifold learning algorithm. In the cases of these data sets the LLE could not provide any additional information. Instead, the trustworthiness of the M-SOM is worse than the SOM visualization and there is no significant difference in preservation of the original neighborhoods. These findings naturally lead to questions, which remain open for future research. Sometimes, the dimensionality and topology of the manifold may suggest using a different kind of a lattice than the one with hexagonal neighborhoods on a two-dimensional sheet, which suffers from the border effect.[24] The LLE may not be the best method to guide the map in all cases. Actually, the idea of supervising the training by another projection could be extended beyond manifold learning to different kinds of applications.

Conclusion

We have presented a modification of the SOM algorithm, named the M-SOM. The proposed algorithm is able to learn nonlinearly embedded manifolds in high-dimensional observation coordinates. The position of an M-SOM node is decomposed into a point in the internal coordinates on the manifold and another point in the high-dimensional observation coordinates. The algorithm iterates the winner node search in the internal coordinates and adaptation in both of the coordinates. In the convergence, the map has learned the projection of the nodes between the coordinates. The M-SOM algorithm requires the projection of the observations onto the internal coordinates to be available before the training process. In this paper, we used the Locally linear embedding algorithm, but, in general, any manifold learning method may be incorporated to the M-SOM algorithm. The proposed method extends the applicability of the SOM algorithm to a class of data sets in which it has previously lacked accuracy.

Acknowledgments

I am thankful for the possibility of using the databases. Especially, Gabriele Peters shared the toy object images recorded at the Institute for Neural Computation, Ruhr-University Bochum, Germany. Yann LeCun shared the MNIST database.

References

1 Kohonen T. *Self-Organizing Maps*. Springer Series in Information Sciences. Springer-Verlag: Berlin, 2001; 521pp.
2 Jolliffe IT. *Principal Component Analysis*. Springer Series in Statistics. Springer-Verlag: Berlin, 2002; 502pp.
3 Tenenbaum JB, de Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. *Science* 2000; **290**: 2319–2323.
4 Roweis ST, Saul LK. Nonlinear dimensionality reduction by locally linear embedding. *Science* 2000; **290**: 2323–2326.
5 Donoho DL, Grimes C. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences of the USA* 2003; **100**: 5591–5596.
6 Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation* 2003; **15**: 1373–1396.
7 Lee JA, Lendasse A, Verleysen M. Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. *Neurocomputing* 2004; **57**: 49–76.

8 Tenenbaum JB. Mapping a manifold of perceptual observations. In: Jordan MI, Kearns MJ, Solla SA (Eds). *Advances in Neural Information Processing Systems*. Vol 10. MIT Press: Cambridge, MA, 1998; 682–688.
9 Li JX. Visualization of high-dimensional data with relational perspective map. *Information Visualization* 2004; **3**: 49–59.
10 Weinberger KQ, Saul LK. Unsupervised learning of image manifolds by semidefinite programming. *IEEE Conference on Computer Vision and Pattern Recognition 2004*. (Washington, DC, USA), Vol. 2, IEEE Computer Society Press: Silver Spring, MD, 2004; 988–995.
11 Swets DL, Weng JJ. Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1996; **18**: 831–836.
12 Kim YS, Kim WY. Content-based trademark retrieval system using visually salient features. *IEEE Conference on Computer Vision and Pattern Recognition 1997*. (San Juan, Puerto Rico), IEEE Computer Society Press: Silver Spring, MD, 1997; 307–312.
13 de Silva V, Tenenbaum JB. Global versus local methods in nonlinear dimensionality reduction. In: Becker S, Thrun S, Obermayer K (Eds). *Advances in Neural Information Processing Systems*. Vol. 15. MIT Press: Cambridge, MA, 2003; 705–712.
14 Saul LK, Roweis ST. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research* 2003; **4**: 119–155.
15 Kiviluoto K. Predicting bankruptcies with the self-organizing map. *Neurocomputing* 1998; **21**: 191–201.
16 Venna J, Kaski S. Neighborhood preservation in nonlinear projection methods: an experimental study. *International Conference on Artificial Neural Networks 2001*. (Vienna, Austria), Springer-Verlag: Berlin, 2001; 485–491.
17 Kaski S, Nikkilä J, Oja M, Venna J, Törönen P, Castrén E. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics* 2003; **4**: 48.
18 Nikkilä J, Törönen P, Kaski S, Venna J, Castrén E, Wong G. Analysis and visualization of gene expression data using self-organizing maps. *Neural Networks* 2002; **15**: 953–966.
19 Ultsch A, Siemon HP. Kohonens self organizing feature maps for exploratory data analysis. *International Neural Network Conference 1990*. (Paris, France), Kluwer, Academic Publishers: Dordrecht, 1990; 305–308.
20 Peters G, Zitova B, von der Malsburg C. How to measure the pose robustness of object views. *Image and Vision Computing* 2002; **20**: 249–256.
21 Alhoniemi E, Hollmén J, Simula O, Vesanto J. Process monitoring and modeling using the self-organizing map. *Integrated Computer-Aided Engineering* 1999; **6**: 3–14.
22 Himberg J. A SOM based cluster visualization and its application for false coloring. *International Joint Conference on Neural Networks, 2000*. (Como, Italy), Vol. 3, IEEE Computer Society Press: Silver Spring, MD, 2000; 587–592.
23 Vesanto J, Alhoniemi E. Clustering of the self-organizing map. *IEEE Transactions on Neural Networks* 2000; **11**: 586–600.
24 Wu YX, Takatsuka M. The geodesic self-organizing map and its error analysis. *Australasian Computer Science Conference 2005*. (Newcastle, Australia), Vol. 38, Conferences in Research and Practice in Information Technology, 2005; 343–352.