Expanding Self-organizing Map for Data Visualization and Cluster Analysis

Huidong Jin [a,b], Wing-Ho Shum [a], Kwong-Sak Leung [a], Man-Leung Wong [c]

[a] *Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong.*

[b] *CSIRO, Mathematical & Information Sciences, Canberra, Australia.*

[c] *Department of Computing and Decision Sciences, Lingnan University, Tuen Mun, Hong Kong.*

Abstract

The Self-Organizing Map (SOM) is a powerful tool in the exploratory phase of data mining. It is capable of projecting high-dimensional data onto a regular, usually 2-dimensional grid of neurons with good neighborhood preservation between two spaces. However, due to the dimensional conflict, the neighborhood preservation cannot always lead to perfect topology preservation. In this paper, we establish an Expanding SOM (ESOM) to preserve better topology between the two spaces. Besides the neighborhood relationship, our ESOM can detect and preserve an ordering relationship using an expanding mechanism. The computation complexity of the ESOM is comparable with that of the SOM. Our experiment results demonstrate that the ESOM constructs better mappings than the classic SOM, especially, in terms of the topological error. Furthermore, clustering results generated by the ESOM are more accurate than those obtained by the SOM.

Key words: Self-organizing map, neural networks, data mining, visualization, clustering analysis.

1 Introduction

The Self-Organizing Map (SOM) has been proven to be useful as visualization and data exploratory analysis tools [6]. It maps high-dimensional data items onto a low-dimensional grid of neurons. The regular grid can be used as a convenient visualization surface for showing different features of data such as the cluster tendencies of the data [8, 12, 16]. SOMs have been successfully applied in various engineering applications covering areas such as pattern recognition, full-text and image analysis, vector quantization, regression, financial data analysis, traveling salesman problem, and fault diagnosis [3, 4, 6, 7, 10, 14, 18].

However, because a SOM maps the data from a high-dimensional space to a low-dimensional space which is usually 2-dimensional, a dimensional conflict may occur and a perfect topology preserving mapping may not be generated [1, 5]. For example, consider the two trained SOMs depicted in Fig.1, although they preserve good neighborhood relationships, the SOM depicted in Fig.1(b) folds the neuron string onto data irregularly and loses much topology information in comparison with the SOM shown in Fig.1(a).

There are many research efforts to enhance SOMs for visualization and cluster analysis. Most of them focus on how to visualize neurons clearly and classify data [3, 14, 16]. Some work has concentrated on better topology preservation. Kirk and Zurada [5] trained their SOM to minimize the quantization error in

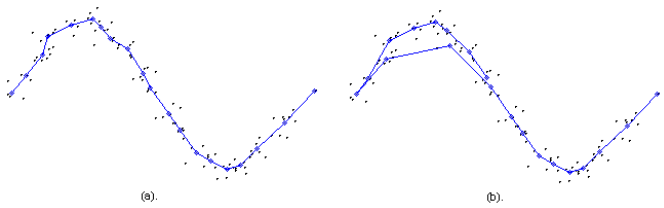

Fig. 1. Two SOMs from 2-dimensional space to 1-dimension. The connected dots indicate a string of neurons, and other dots indicate data.

the first phase and then minimize the topological error in the second phase. Su and Chang proposed a Double SOM (DSOM) that uses a dynamic grid structure instead of a static structure used in the conventional SOMs. The DSOM uses the classic SOM learning rule to learn a grid structure from input data [12].

Motivated by an irregularity problem of SOMs and our previous work of using SOM for traveling salesman problem [4], we propose a new learning rule to enhance the topology preservation. The paper is organized as follows. We outline the SOM techniques in the next section. We introduce our ESOM in Section 3, followed by its theoretic analysis. The visualization and clustering results of the ESOM are presented and compared with the SOM in Section 4. A conclusion is given in the last section.

2 Self-Organizing Map

The SOM consists of two layers of neurons. The neurons on the input layer receive data $\vec{x}_k(t) = [x_{1k}(t), x_{2k}(t), \cdots, x_{Dk}(t)]^T \in \Re^D$ $(1 \leq k \leq N)$ at time t where D is the dimensionality of the data space and N is the number of data items in the data set. The neurons on the output layer are located on a grid with certain neighborhood relationship. In this paper, the rectangular

neighborhood is used. The weight vector $\vec{w}_j(t) = [w_{1j}(t), w_{2j}(t), \cdots, w_{Dj}(t)]^T \in \Re^D$ $(1 \leq j \leq M)$ indicates the j-th output neuron's location in the data space. It moves closer to the input vector according to

$$\vec{w}_j(t+1) = \vec{w}_j(t) + \epsilon(t) h_{j,m(t)} \left[\vec{x}_k(t) - \vec{w}_j(t)\right]. \tag{1}$$

Here $m(t)$ is the winning neuron, $h_{j,m(t)}$ is *the neighborhood function*, and $\epsilon(t)$ is *the learning rate* and usually shrinks to zero. Fig.(2)(a) illustrates this learning rule. During the learning, the SOM behaves like a flexible net that folds onto the "cloud" formed by the input data. It finally constructs a neighborhood preserving map so that the neurons adjacent on the grid have similar weight vectors. The SOM usually maps a high-dimensional data set to a low-dimensional grid, so a dimensional conflict may occur in the trained SOM. Thus, the neighborhood preserving map of the SOM is a good but usually not a perfect topology preserving one.

3 Expanding SOM

Besides the neighborhood relationship in the SOM, another topology relationship can be detected and preserved during the learning process to achieve a better topology preserving mapping for data visualization. This is a linear ordering relationship based on the distance between data and their center. A neural network can detect and preserve this ordering relationship. If the distance between a data item and the center of all data items is larger, the distance between the corresponding output neuron and the center is also larger. The linear ordering relationship contains certain important topological information. A typical example may be found in Fig.1. Though two SOMs in Fig.1

(a) and (b) have similar quantization errors, the left one certainly has less topological error. In other words, the left one visualizes the data set better.

In order to overcome such irregularity as in Fig.1(b), we proposed the Expanding SOM (ESOM) to learn the linear ordering through expanding. The ESOM can construct a mapping that preserves both the neighborhood and the ordering relationships. Since this mapping preserves more topology information of the input data, better performance in visualization can be expected.

We introduce a new learning rule to learn the linear ordering relationship. Different from the SOM, the learning rule of the ESOM has an additional factor, the expanding coefficient $c_j(t)$, which is used to push neurons away from the center of all data items during the learning process. In other words, the flexible neuron network is expanding gradually in our ESOM algorithm. Moreover, the expanding force is specified according to the ordering of the data items. In general, the larger the distance between the corresponding data item and the center is, the larger the expanding coefficient $c_j(t)$ is, the larger the distance between the corresponding data item and the center, the larger the expanding coefficient $c_j(t)$. Consequently, the associated output neuron is pushed away from the center and the ordering of data items is thus preserved in the output neurons. For example, the good topological preserving map in Fig. 1(a) can be achieved.

In the following sub-sections, the ESOM algorithm will be discussed first. Theoretical analysis of the ESOM algorithm will then be described.

3.1 The ESOM algorithm

The ESOM algorithm consists of 6 steps.

(1) Linearly transform the coordinates $\vec{x}_i^{'} = [x'_{1i}, x'_{2i}, \cdots, x'_{Di}]^T$ $(i = 1, \cdots, N)$ of all given data items so that they lie within a sphere S_R centered at the origin with radius R (< 1). Here N is the number of data items, D is the dimensionality of the data set. Hereafter, $[x_{1i}, x_{2i}, \cdots, x_{Di}]^T$ denotes the new coordinate of $\vec{x}_i$. Let the center of all data items be $\vec{x}_C^{'} = \frac{1}{N}\sum_{i=1}^{N}\vec{x}_i^{'}$ and the maximum distance of data from the data center be D_{max}, then

$$\vec{x}_i = \frac{R}{D_{max}}\left(\vec{x}_i^{'} - \vec{x}_C^{'}\right) \text{ for all } i. \tag{2}$$

(2) Set $t = 0$, and the initialize weight vectors $\vec{w}_j(0)$ $(j = 1, \cdots, M)$ with random values within the above sphere S_R where M is the number of output neurons.

(3) Select a data item at random, say $\vec{x}_k(t) = [x_{1k}, x_{2k}, \cdots, x_{Dk}]^T$, and feed it to the input neurons.

(4) Find the winning output neuron, say $m(t)$, nearest to $\vec{x}_k(t)$ according to the Euclidean metric:

$$m(t) = \arg\min_j \|\vec{x}_k(t) - \vec{w}_j(t)\| . \tag{3}$$

(5) Train neuron $m(t)$ and its neighbors by using the following formula:

$$\vec{w}_j(t+1) = c_j(t)\vec{w}_j^{'}(t+1) \triangleq c_j(t)\{\vec{w}_j(t) + \alpha_j(t)\left[\vec{x}_k(t) - \vec{w}_j(t)\right]\} \tag{4}$$

The parameters include:

- *the interim neuron* $\vec{w}_j^{'}(t+1)$, which indicates the position of the excited neuron $\vec{w}_j(t)$ after moving towards the input data item $\vec{x}_k(t)$;

- *the learning parameter* $\alpha_j(t)(\in [0,1])$, which is specified by a learning rate $\epsilon(t)$ and a neighborhood function $h_{j,m(t)}(\sigma(t))$:

$$\alpha_j(t) = \epsilon(t) \times h_{j,m(t)}(\sigma(t)); \tag{5}$$

- *the expanding coefficient* $c_j(t)$, which is specified according to

$$c_j(t) = \left[1 - 2\alpha_j(t)\left(1 - \alpha_j(t)\right)\kappa_j(t)\right]^{-\frac{1}{2}}, \tag{6}$$

  where $\kappa_j(t)$ is specified by

$$\kappa_j(t) = 1 - \langle \vec{x}_k(t), \vec{w}_j(t) \rangle - \sqrt{(1 - \|\vec{x}_k(t)\|^2)(1 - \|\vec{w}_j(t)\|^2)} \tag{7}$$

(6) Update the neighbor width parameter $\sigma(t)$ and the learning parameters $\epsilon(t)$ with predetermined decreasing schemes. If the learning loop does not reach a predetermined number, go to Step 3 with $t := t + 1$.

The first step facilitates the realization of the expanding coefficient $c_j(t)$. After the transformation, we can use the norm of a data item $\|\vec{x}_k(t)\|$ to represent its distance from the center of the transformed data items since the center is the origin. Thus, the norm $\|\vec{x}_k(t)\|$ can indicate the ordering topology in the data space. This ordering will be detected and preserved in $\|\vec{w}_j(t)\|$ through the expanding process.

The learning rule defined in Eq.(4) is the key point of the proposed ESOM algorithm. Different from the SOM learning rule, it has an additional multiplication factor — the expanding coefficient $c_j(t)$. This expanding coefficient is greatly motivated by our work of applying SOM to the traveling salesman problem [4], where the expanding coefficient is defined in a 2-D space by virtue of a unit sphere. Extending the formula for the 2-D case into a D-dimensional space, we get Eq.(4).

It is worth pointing out that, although the expanding coefficient $c_j(t)$ is relevant to all data items, the calculation of $c_j(t)$ only depends on $\alpha_j(t)$, $\vec{x}_k(t)$ and $\vec{w}_j(t)$. If $c_j(t)$ is a constant 1.0, the ESOM is simplified to a conventional SOM. Since $c_j(t)$ is always greater than or equal to 1.0, the expanding force pushes the excited neuron away from the center. In other words, the inequality $\|\vec{w}_j(t+1)\| \geq \left\|\vec{w}'_j(t+1)\right\|$ is always true. Fig.2(b) illustrates the expanding functionality. After moving the excited neuron $\vec{w}_j(t)$ towards the input data item $\vec{x}_k(t)$, as indicated by $\vec{w}'_j(t+1)$, the neuron is then pushed away from the center. So, during the learning process, the flexible neuron net is expanding in the data space. More interestingly, as the expanding force is specified accordingly to the ordering relationships, distant data items are likely to be mapped to the distant neurons, while data items near the center are likely to be mapped to the neurons near the center of map, therefore, $c_j(t)$ can help us to detect and preserve the ordering relationship. The expanding coefficient $c_j(t)$ is also close to 1, which enables the ESOM to learn and preserve the neighborhood relationship as the SOM does. We will give a theoretical analysis on this point in the next subsection.

There are several variations of SOM for better topology preservation, but most of them are either computationally expensive or structurally complicated. GSOM [11] needs to expand its output neuron layer gradually during learning. However, the expansion of ESOM only happens on the output neurons' weight vectors, rather than the structure of output neuron layer. Besides finding the nearest neuron for the input data item as in ESOM/SOM, GSOM also needs to determine whether, where, and how a new output neuron has to be inserted. The two-stage SOM [5] uses both a k-means algorithm and a partition error based heuristic to improve the topography preservation. However,

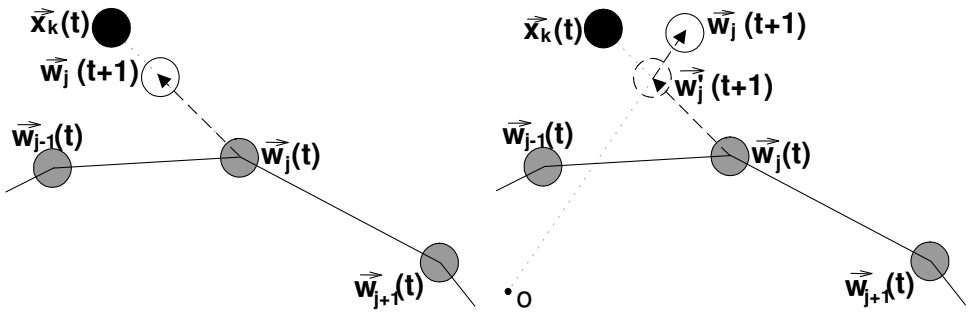

Fig. 2. A schematic view of two different learning rules. (a). The learning rule for the traditional SOM; (b). The learning rule for the ESOM. A black disc indicates a data vector; a gray disc indicates a neuron; a solid line indicates the neighbor relationship on the grid; a circle indicates the new position of a neuron; a dashed circle indicates a neuron's temporary position; a dashed arrow indicates a movement direction; and 'o' indicates the origin, i.e., the data center.

it causes a great number of minor topological discontinuities. By incorporating the fuzzy concept into the SOM, the Fuzzy SOM [13] is able to handle vague and imprecise data, but it demands a high computational cost for the membership functions. The tree-structured SOM [3] can generate a tree of SOM dynamically, It can handle many difficult data sets but it spends almost twice longer than the SOM [2].

3.2 Theoretical analysis

To show the feasibility of the ESOM, we should verify that the ESOM does generate a map that preserves both the neighborhood and the ordering relationships. In this paper, we only give a theorem on a one-step trend to support the feasibility of the ESOM because it is very difficult to prove the convergence of the SOM-like networks in higher dimensional cases. In fact, it is still one of the long-standing open research problems in neural networks [9]. We will

perform a more rigorous convergence analysis in our future work if the convergence analysis of the SOM is fulfilled. In the following theorem, we assume that all input data items are located within the sphere S_R and their center coincides with the origin because the preprocessing procedure in Step 1 has been executed.

Theorem 1 *Let S_R be the closed sphere with radius R (< 1) centered at the origin, $\{\vec{x}_k(t) \in S_R\}$ (for $k = 1, \cdots, N$) be the input data and $\{\vec{w}_j(t)\}$ (for $j = 1, \cdots, M$) be the weight vectors of the ESOM at time t. Then, for any $t \geq 0$,*

(i). *for $j \in \{1, 2, \cdots, M\}$,*

$$1 \leq c_j(t) \leq \frac{1}{\sqrt{1-R^2}}; \tag{8}$$

and $\vec{w}_j(t) \in S_R$, that is,

$$\|\vec{w}_j(t)\| \leq R. \tag{9}$$

(ii). *the expanding coefficient $c_j(t)$ increases with $\|\vec{x}_k(t)\|$ when $\|\vec{x}_k(t)\| \geq \|\vec{w}_j(t)\|$.*

Proof: **(i).** We prove Eqs.(8) and (9) together by induction. This is trivially true for $t = 0$ according to Step 2 of the ESOM algorithm. If we assume that both equations hold for certain $t(\geq 0)$, then we find

$$\begin{aligned}
1 - \kappa_j(t) &= \langle \vec{x}_k(t), \vec{w}_j(t) \rangle + \sqrt{(1 - \|\vec{x}_k(t)\|^2)}\sqrt{(1 - \|\vec{w}_j(t)\|^2)} \\
&\leq \left(\sum_{d=1}^{D} x_{dk}^2(t) + \left(\sqrt{(1 - \|\vec{x}_k(t)\|^2)} \right)^2 \right) \\
&\quad \times \left(\sum_{d=1}^{D} w_{dj}^2(t) + \left(\sqrt{(1 - \|\vec{w}_j(t)\|^2)} \right)^2 \right) \\
&= 1.
\end{aligned}$$

Similarly,

$$\begin{aligned}1-\kappa_j(t) &= \langle \vec{x}_k(t), \vec{w}_j(t)\rangle + \sqrt{(1-\|\vec{x}_k(t)\|^2)}\sqrt{(1-\|\vec{w}_j(t)\|^2)} \\ &\geq -\frac{1}{2}(\|\vec{x}_k(t)\|^2 + \|\vec{w}_j(t)\|^2) + \sqrt{(1-R^2)}\sqrt{(1-R^2)} \\ &\geq 1-2R^2.\end{aligned}$$

Thus,

$$0 \leq \kappa_j(t) \leq 2R^2$$

On the other hand, for any learning parameter $\alpha_j(t) \in [0,1]$, the following inequality is true,

$$0 \leq \alpha_j(t)\,(1-\alpha_j(t)) \leq 0.25.$$

According to Eq.(6), we get $1 \leq c_j(t) \leq \frac{1}{\sqrt{1-R^2}}$. According to the ESOM learning rule, we have

$$\begin{aligned}1-\|\vec{w}_j(t+1)\|^2 &= \left[[c_j(t)]^{-2} - \|\vec{w}_j(t)+\alpha_j(t)(\vec{x}_k(t)-\vec{w}_j(t))\|^2\right] \times (c_j(t))^2 \\ &= \frac{\left[(1-\alpha_j(t))\sqrt{1-\|\vec{w}_j(t)\|^2} + \alpha_j(t)\sqrt{1-\|\vec{x}_k(t)\|^2}\right]^2}{(c_j(t))^{-2}} \\ &\geq \left[(1-\alpha_j(t))\sqrt{1-R^2} + \alpha_j(t)\sqrt{1-R^2}\right]^2 \\ &= 1-R^2.\end{aligned} \tag{10}$$

This implies that $\|\vec{w}_j(t+1)\| \leq R$ for any $j = 1, \cdots, M$. Thus, by induction, $\vec{w}_j(t) \in S_R$ for any j and t.

(ii). We rewrite $\vec{x}_k(t)$ and $\vec{w}_j(t)$ as follows,

$$\begin{aligned}\vec{x}_k(t) &= \rho \times \vec{e}_{x_k} \\ \vec{w}_j(t) &= r \times \vec{e}_{w_j}\end{aligned}$$

Here $\vec{e}_{x_k}$ and $\vec{e}_{w_j}$ are two unit vectors, and $\rho = \|\vec{x}_k(t)\|$ and $r = \|\vec{w}_j(t)\|$. According to the assumption that $\rho \geq r$ holds. Let

$$\begin{aligned} F(\rho) &= \langle \vec{w}_j(t), \vec{x}_k(t) \rangle \ + \sqrt{(1 - \|\vec{w}_j(t)\|^2)(1 - \|\vec{x}_k(t)\|^2)} \\ &= \rho \cdot r \cdot \left\langle \vec{e}_{w_j}, \vec{e}_{x_k} \right\rangle + \sqrt{(1-\rho^2)(1-r^2)}. \end{aligned} \tag{11}$$

According to Eq.(6), it is obvious that $F(\rho) = 1 - \frac{1 - c_j^{-2}(t)}{2\alpha_j(t)(1-\alpha_j(t))}$. $F(\rho)$ decreases with the expanding coefficient $c_j(t)$. So, to justify the increasing property of $c_j(t)$, it is sufficient to show that $F(\rho)$ decreases with ρ whenever $\rho \geq r$. A direct calculation shows

$$\frac{\partial F(\rho)}{\partial \rho} = r \cdot \left\langle \vec{e}_{w_j}, \vec{e}_{x_k} \right\rangle - \frac{\rho}{\sqrt{1-\rho^2}} \sqrt{1-r^2} \tag{12}$$

$$\leq r - \rho \leq 0. \tag{13}$$

This implies that the decreasing property of $F(\rho)$ on ρ when $\rho \geq r$. ■

Theorem 1 (i) says that the expanding coefficient $c_j(t)$ is always larger than or equal to 1.0. In other words, it always pushes neurons away from the origin. Thus, during learning, the neuron net is expanding. Furthermore, though the expanding force is always greater than or equal to 1.0, it will never push the output neurons to infinite locations. In fact, it is restricted by sphere S_R in which the data items are located. This point is substantiated by Eq.(9). In other words, the expanding coefficient is never very large and enables the ESOM to learn the neighborhood relationships as the SOM does.This supports the feasibility of the proposed ESOM.

Theorem 1 (ii) gives a theoretic support that the ESOM aims to detect and preserve the ordering relationship among the training data items. It points out that the expanding coefficient $c_j(t)$, or the expanding influence, is different for various data items. The larger the distance between a data item and the

center of all data items is, the stronger the expanding force will influence on the associated output neuron. Consequently, the output neurons are located in the data space according to the linear ordering of their associated data items.

We now briefly discuss another interesting trend based on the proof procedure. If $\vec{w}_j(t)$ is far away from $\vec{x}_k(t)$ [1], $\left\langle \vec{e}_{w_j}, \vec{e}_{x_k} \right\rangle$ will be very small or even less than 0. From Eq.(12), $\frac{\partial F(\rho)}{\partial \rho} \approx -\frac{\rho}{\sqrt{1-\rho^2}}\sqrt{1-r^2} \leq 0$. In other words, the expanding coefficient $c_j(t)$ increases with ρ which is the distance of the input data item $\vec{x}_k(t)$ from the center. So, the ordering of $\|\vec{x}_k(t)\|$ is reflected in by the expanding coefficient $c_j(t)$ and then is learned by $\vec{w}_j(t)$. This also explains why the topological error of the ESOM decreases more quickly than that of the SOM at the beginning of learning. A typical example can be found in Fig. 4 in Section 4.

In this paragraph, we derive the computation complexity of the ESOM algorithm. Two differences between the ESOM and the SOM algorithms are the preprocessing Step 1 and the learning rule. The computation complexity of the preprocessing step is $O(N)$. The learning rule of the ESOM in Eq.(4) needs a few extra arithmetic operations in comparison with the one of the conventional SOM. Thus, the total computation complexity of the ESOM algorithm is comparable with that of the SOM which is $O(MN)$ [16].

[1] the case is common at the beginning of learning since the weight vector $\vec{w}_j(t)$ is randomly initialized.

Table 1

Data Sets Properties

Data Set	No. of dimensions	No. of records	No. of classes
Data Set 1	2	3000	3
Data Set 2	3	2000	2
Data Set 3	3	3000	3
Mars	4	234	unknown
Viking 2	7	29225	unknown

4 Experimental results

We have examined the ESOM on 3 synthetic data sets and 2 real-life data sets. Table 1 shows the properties of the five data sets. The first three synthetic data sets are interesting in both their special cluster shapes and locations as illustrated in Fig.3. The conventional clustering algorithms such as K-means and expectation-maximization (EM) are unable to identify the clusters. The fourth data set, Mars, contains the temperatures taken from the Mars Pathfinder's three sensors in different height. The fifth data set, Viking 2, contains the measurements of solar longitude, wind speed, pressure, and temperature taken from the Viking Lander 2. The Mars and Viking 2 data sets are downloaded from the web site "The Live from Earth & Mars" of University of Washington (http://www-k12.atmos.washington.edu/k12/resources/mars_data-information/data.html).

All data sets have been pre-processed by using the linear transformation described in Eq.(2) in order to compare results fairly. The initial values of ϵ, σ, and R are 0.5, 0.9 and 0.999 respectively. The values of α and σ are decreased

by 0.998 per iteration. Except for the Mars data set, we have used a rectangular grid with 20*20 neurons. As there are not many data items in Mars data sets, an output grid with 10 * 10 neurons is used. All experiments have been executed for 2000 iterations.

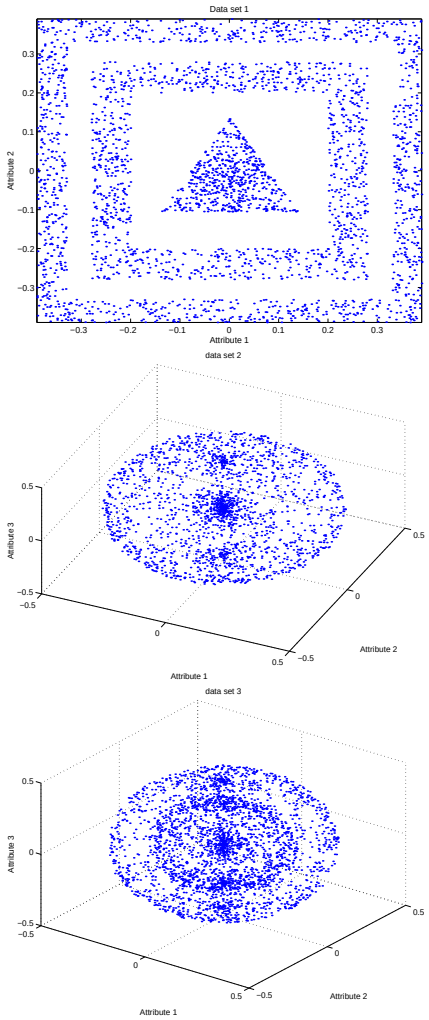

Fig. 3. Illustration of the 3 synthetic data sets, Data Set 1 (Upper), Data Set 2 (Middle) and Data Set 3 (Lower)

Researchers have introduced several measures to evaluate the quality of a mapping [1, 17]. In this paper, we employ the topological error E_T used in [5, 15] to evaluate the mapping obtained by our ESOM. E_T is defined as the proportion of the data items for which the closest and the second-closest

neurons are not adjacent on the grid. On the other hand, since the number of data items is normally larger than the number of output neurons, the weight vectors in the trained SOM become representatives of the original data set. The quantization error evaluates how well the weight vectors represent the data set [5, 16]. It is specified as follows:

$$E_Q = \frac{1}{N}\sum_{k=1}^{N} \|\vec{x}_k(t) - \vec{w}_{m_k}(t)\| \tag{14}$$

where m_k is the winner for the data vector $\vec{x}_k(t)$. These two criteria usually conflict in the SOM.

4.1 Results of 10 independent runs

We have performed 10 independent runs of both algorithms for the five data sets. The average topological errors, the average quantization errors and the average execution time are summarized in Tables 2, 3 and 4 respectively.

Table 2

Average Topological Errors based on 10 independent runs.

Data Set Name	ESOM	SOM	Improvement (%)
Data Set 1	0.24088	0.26039	7.49154
Data Set 2	0.30215	0.30555	1.11241
Data Set 3	0.33132	0.34763	4.69203
Mars	0.47059	0.53589	12.18534
Viking 2	0.79041	0.82103	3.72946

On average, the quantization errors of the ESOM for the three synthetic data

Table 3

Average Quantization Errors based on 10 independent runs.

Data Set Name	ESOM	SOM	Improvement (%)
Data Set 1	0.01767	0.01800	1.80060
Data Set 2	0.04753	0.04832	1.64508
Data Set 3	0.05991	0.06042	0.83417
Mars	0.01308	0.01305	-0.22989
Viking 2	0.02370	0.02369	-0.04221

Table 4

Average Execution Times based on 10 independent runs.

Data Set Name	ESOM	SOM	Difference (%)
Data Set 1	2371.6	2283.2	3.87176
Data Set 2	2068.9	2051.2	0.86291
Data Set 3	2904.7	2872.1	1.13506
Mars	32.3	31.7	1.85758
Viking 2	4844	4704	2.89017

sets are 0.01767, 0.04753 and 0.05991, which are little smaller than those of the SOM. The average topological errors of the ESOM are 0.24088, 0.30215, and 0.33132 respectively. They are 7.49152%, 1.11241% and 4.69203% smaller than those of SOM. In addition, the ESOM only needs little longer execution time as listed in Table 4. Therefore, for the synthetic data sets, the ESOM generates better mappings than the SOM in terms of both the topological and

the quantization errors with similar execution time.

For the first real-life data, the topological error of ESOM is 0.47059, which is 12.18534% shorter than the value of the SOM, 0.53589. On the other hand, the quantization error of ESOM is only 0.22989% larger than the one of the SOM. Similar comparison is found for the second real-life data set, where the ESOM makes 3.72946% improvement on the topological error and only 0.04221% loss on the quantization error. The gain on the topological error is very obvious in comparison with the loss on the quantization error. Therefore, using similar execution time, the ESOM can generate mappings with much better topological errors than the SOM. The ESOM has similiar quantization errors as SOM.

4.2 Typical results for the synthetic data sets

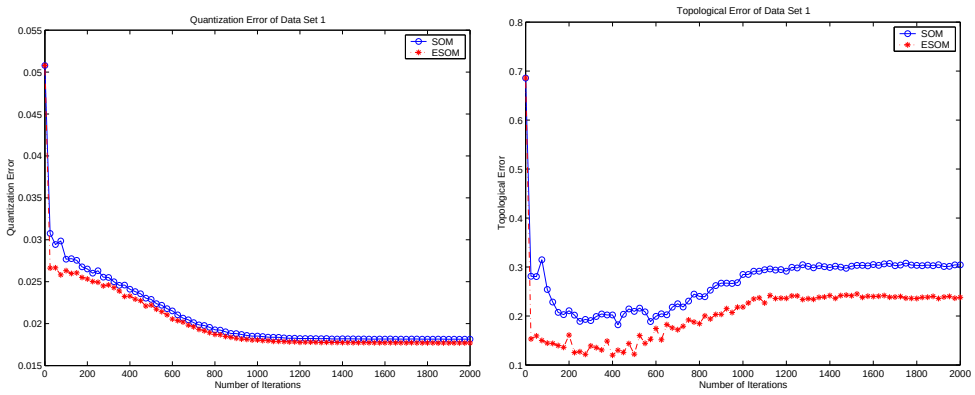

Fig. 4. The quantization error (Left) and the topological error (Right) during the learning of the ESOM and the SOM for the first data set.

Fig.4 illustrates the quantization and the topological errors during a typical run on the first data set of both algorithms. It is clearly seen that the quantization error decreases gradually as the learning process continues. The quantization error of the trained ESOM is 0.017 which is a bit smaller than

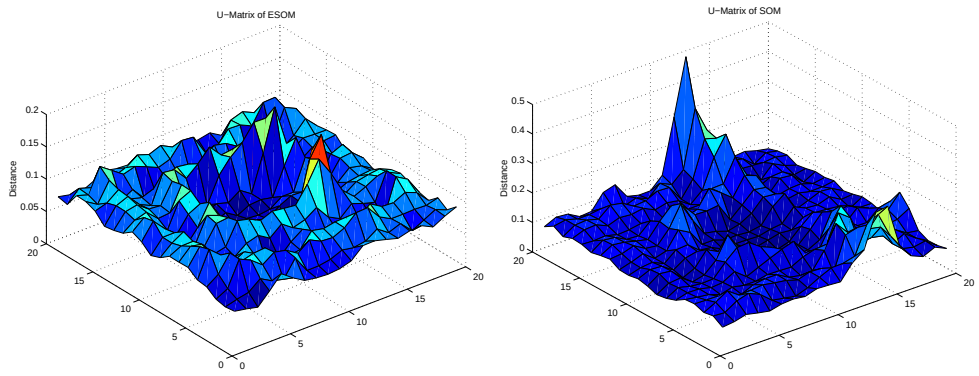

Fig. 5. U-Matrix of the trained ESOM (Left) and the trained SOM (Right) for the first data set.

that of the trained SOM, 0.018. During learning, the decreasing, the increasing and the converging stages can be observed in the topological error curve. At the very beginning of the training process, the neuron's weights are fairly dislike, while some of them even contain remnants of random initial values, thus higher topological errors are obtained. After executing several iterations, the topological error decreases dramatically. Because the learning rate ϵ and the neighborhood function are large, the neurons adjacent on the grid may move much closer to the input data item together. At this stage, the ESOM can learn the ordering topology of data items quickly. As shown in Fig.4, the topological error of the ESOM is much smaller than that of the SOM. The final topological errors of the ESOM and the SOM are 0.238 and 0.304 respectively, ESOM gains about 20% improvement. Thus, the ESOM can generate better topology preserving maps than the SOM.

Fig.5 illustrates the trained ESOM and SOM for the first data set in the form of U-matrix. The x-axis and the y-axis of the U-matrix indicate a neuron's position on the grid, and the z-axis is the average Euclidean distance of neurons from its adjacent ones [3]. A sequence of consecutive peaks can be regarded as a boundary among clusters, while the basins are regarded as clusters. There

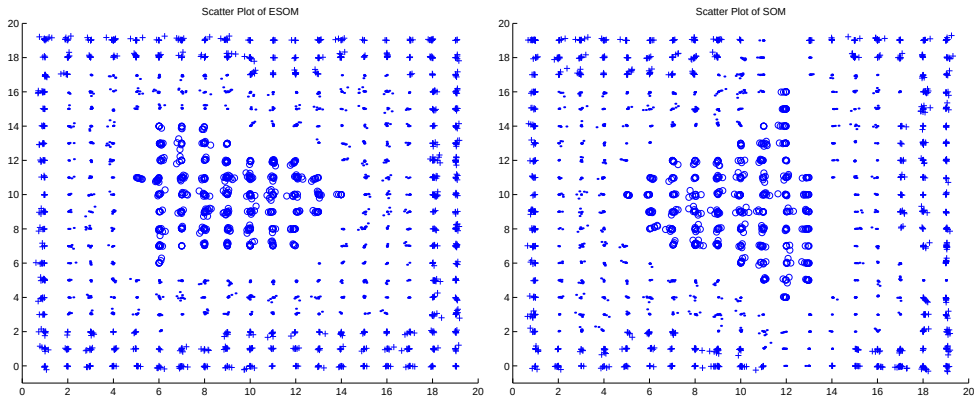

Fig. 6. Scatter plot the trained ESOM (Left) and the trained SOM (Right) for the first data set.

are clearly two sequences of peaks in the ESOM's U-matrix, which indicate three clusters in the first data set. The layer structure in the data set is clearly illustrated. In contrast, The boundaries in the SOM's U-matrix is not so clear because some high peaks blur the boundaries. The scatter plots shown in Fig.6 illustrate data clusters on the grid. In the scatter plot, each marker represents a mapping of a data item and the shape of the marker indicates its cluster label. The marker is placed on the winning neuron of the data item. To avoid overlapping, the marker has plotted with a small offset which is determined according to the data item's Euclidean distance from the winning neurons. The ESOM maps data items in well-organized layers. We can easily find the three clusters in its scatter plot which is quite similar with the original data set as shown in Fig.3. However, the SOM cannot map the data items very well. The outer cluster in Fig.3 is even separated into three subclusters (indicated by '+').

Fig.7 illustrates the scatter plots of a typical run of both algorithms for the second data set. The scatter plot of the ESOM clearly shows two clusters where one cluster surrounds the other as in the original data set depicted in

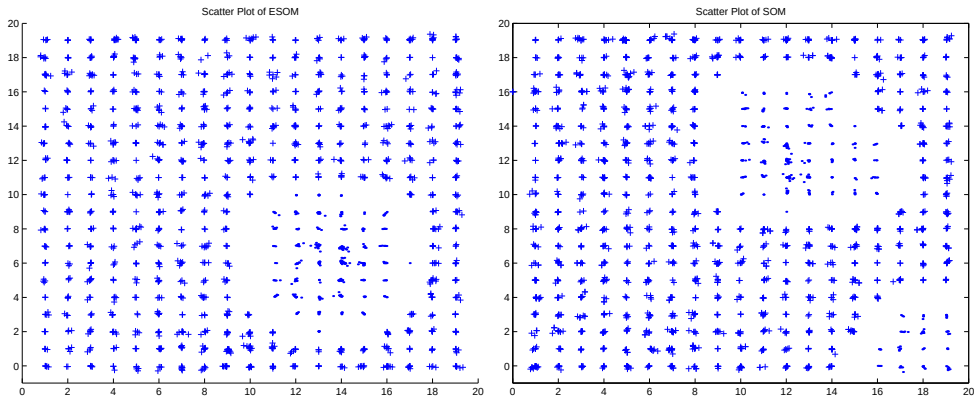

Fig. 7. Scatter plot the trained ESOM (Left) and the trained SOM (Right) for the second data set.

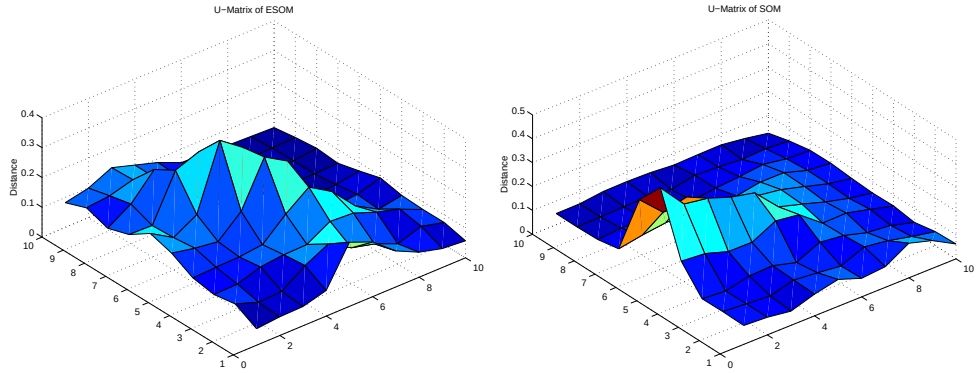

Fig. 8. U-Matrices of the ESOM (Left) and the SOM (Right) for the Mars data set.

Fig.3. In contrast, the SOM separates one of the clusters in the original data set into two clusters(represented by '.').

4.3 Typical results for the real-life data sets

Fig.8 illustrates the U-matrices during a typical run on the Mars data set of both algorithms. By comparing all the data items and the weight values of all the neurons in the trained ESOM and SOM respectively, we have found that both of the algorithms are able to discover two clusters, but only ESOM can show the boundary clearly in the U-matrix.

From the above experimental and comparison results, we conclude that the ESOM can generate better topology preserving mappings than the SOM in terms of both the topological error and quantization error. The ESOM is more likely to identify the correct clusters from data sets than the SOM.

5 Conclusion

In this paper, we have proposed an Expanding Self-Organizing Map (ESOM) to detect and preserve better topology correspondence between the input data space and the output grid. During the learning process of our ESOM, the flexible neuron net is expanding and the neuron corresponding to a distant data item gets large expanding force. Besides the neighborhood relationship as in the SOM (Self-Organizing Map), the ESOM can detect and preserve a linear ordering relationship as confirmed by our theoretical analysis. Our experiment results have substantiated that, with similar execution time, the ESOM constructs better visualization results than the classic SOM, especially, in terms of the topological error. Furthermore, clustering results generated by the ESOM are more accurate than those obtained by the SOM on both the synthetic and the real-life data sets.

A rigorous theoretical analysis of the ESOM algorithm is subject to our future work, which heavily relies on the convergence analysis of SOM. We are also interested in applying the ESOM algorithm to large-scale and high-dimensional data sets.

6 Acknowledgement

This research was supported by RGC Earmarked Grant for Research CUHK 4212/01E of Hong Kong.

References

[1] H. U. Bauer, M. Herrmann, and T. Villmann. Neural maps and topographic vector quantization. *Neural Networks*, 12:659–676, 1999.

[2] C. M. Bishop, M. Svensen, and C. K. I. Williams. Gtm: a principled alternative to the self-organizing map. In *Proceedings of ICANN'96, International Conference on Artificial Neural Networks*, pages 165–170, 1996.

[3] J.A.F. Costa and M.L. de Andrade Netto. A new tree-structured self-organizing map for data analysis. In *Proceedings of International Joint Conference on Neural Networks*, volume 3, pages 1931–1936, 2001.

[4] Huidong Jin, Kwong-Sak Leung, and Man-Leung Wong. An integrated self-organizing map for the traveling salesman problem. In Nikos Mastorakis, editor, *Advances in Neural Networks and Applications*, pages 235–240, World Scientific and Engineering Society Press, Feb. 2001.

[5] J. S. Kirk and J. M. Zurada. A two-stage algorithm for improved topography preservation in self-organizing maps. In *2000 IEEE International Conference on Systems, Man, and Cybernetics*, volume 4, pages 2527–2532. IEEE Service Center, 2000.

[6] Teuvo Kohonen. *Self-Organizing Maps*. Springer-Verlag, New York, 1997.

[7] Teuvo Kohonen, Samuel Kaski, Krista Lagus, Jarkko SalojAvi, Vesa Paatero, and Antti Saarela. Organization of a massive document collection. *IEEE Transactions on Neural Networks, Special Issue on Neural*

Networks for Data Mining and Knowledge Discovery, 11(3):574–585, May 2000.

[8] R.D. Lawrence, G.S. Almasi, and H.E. Rushmeier. A scalable parallel algorithm for self-organizing maps with applications to sparse data mining problems. *Data Mining and Knowledge Discovery*, 3:171–195, 1999.

[9] Siming Lin and Jennie Si. Weight-value convergence of the SOM algorithm for discrete input. *Neural Computation*, 10(4):807–814, 1998.

[10] Z. Meng and Y.-H. Pao. Visualization and self-organization of multidimensional data through equalized orthogonal mapping. *IEEE Transactions on Neural Networks*, 11(4):1031–1038, July 2000.

[11] U. Seiffert and B. Michaelis. Adaptive three-dimensional self-organizing map with a two-dimensional input layer. In *Intelligent Information Systems, 1996., Australian and New Zealand Conference on , 1996*, pages 258 –263, 1996.

[12] Mu-Chun Su and Hsiao-Te Chang. A new model of self-organizing neural networks and its application in data projection. *IEEE Transactions on Neural Networks*, 12(1):153 –158, Jan. 2001.

[13] A. Tenhagen, U. Sprekelmeyer, and W.-M. Lippe. On the combination of fuzzy logic and kohonen nets. In *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, pages 2144 –2149, 2001.

[14] Juha Vesanto. SOM-based data visualization methods. *Intelligent Data Analysis*, 3:111–26, 1999.

[15] Juha Vesanto. Neural network tool for data mining: SOM toolbox. In *Proceedings of Symposium on Tool Environments and Development Methods for Intelligent Systems (TOOLMET2000)*, pages 184–196, Oulu, Finland, 2000.

[16] Juha Vesanto and Esa Alhoniemi. Clustering of the self-organizing map.

IEEE Transactions on Neural Networks, 11(3):586–600, May 2000.

[17] Thomas Villmann, Ralf Der, Michael Herrmann, and Thomas M. Martinetz. Topology preservation in self-organizing feature maps: exact definition and measurement. *IEEE Transactions on Neural Networks*, 8(2):256 –266, March 1997.

[18] Hujun Yin and Nigel M. Allinson. Self-organizing mixture networks for probability density estimation. *IEEE Transactions on Neural Networks*, 12:405 –411, March 2001.