THE TIME ADAPTIVE SELF-ORGANIZING MAP FOR DISTRIBUTION ESTIMATION

H. Shah Hosseini and R. Safabakhsh

Computer Engineering Department, AmirKabir University of Technology
15914,Tehran, Iran, haamed@ce.aku.ac.ir and safa@ce.aku.ac.ir

(Received: October 6, 1999 – Accepted in Revised Form: October 23, 2001)

Abstract The feature map represented by the set of weight vectors of the basic SOM (Self-Organizing Map) provides a good approximation to the input space from which the sample vectors come. But the time-decreasing learning rate and neighborhood function of the basic SOM algorithm reduce its capability to adapt weights for a varied environment. In dealing with non-stationary input distributions and changing environments, we propose a modified SOM algorithm called "Time Adaptive SOM", or TASOM, that automatically adjusts learning rate and neighborhood function of each neuron independently. Each neuron's learning rate is determined by a function of distance between an input vector and its weight vector. The width of the neighborhood function is updated by a function of the distance between the weight vector of the neuron and the weight vectors of neighboring neurons. Only one time parameter initialization is sufficient throughout the lifetime of TASOM to enable it to work with stationary as well as non-stationary input distributions without the need for retraining. The proposed TASOM is tested with five different input distributions and its performance is compared with that of the basic SOM for these cases. The quantization errors of the TASOM in all of these experiments are lower than the errors of the basic SOM. Moreover, the convergence speed of the TASOM outperforms that of the basic SOM. These experiments demonstrate that the TASOM is stable and convergent. The TASOM network is also tested with non-stationary environments when the input distribution completely changes from one distribution to another. The TASOM in these changing environments moves its weights gradually from the old distribution to the clusters of the new distribution. This property is comparable to the memory of human brain, which gradually forgets old memory and memorizes new sensory data.

Key Words Self-Organizing Map, Non-Stationary Distribution, Neighborhood Function, Quantization Error, Time Adaptive, TASOM

چکیده نگاشت ویژگی که توسط مجموعه بردارهای وزن SOM پایه ارائه می‌شود تقریب خوبی برای فضای ورودی که بردارهای نمونه از آن می‌آیند فراهم می‌آورد. اما تابع همسایگی و نرخ یادگیری کاهنده با زمان الگوریتم SOM پایه توانایی آن را در تطبیقی بودنی وزنها با محیطهای پویا کاهش می‌دهد. برای کار با محیطهای متغییر و توزیع های ناایستا، ما یک الگوریتم اصلاح شده SOM به نام TASOM پیشنهاد می‌کنیم به گونه ای که نرخ یادگیری و تابع همسایگی هر نورون را نابسته با زمان تنظیم می‌کند. نرخ یادگیری هر نورون توسط تابعی از فاصله بین بردار ورودی و بردار وزن آن تعیین می‌شود. پهنای تابع همسایگی توسط تابعی از فاصله بین بردار وزن و بردارهای وزن نورونهای همسایه مشخص می‌شود. تنها یک بار مقداردهی اولیه پارامترها برای TASOM کافی است تا بتواند در محیطهای ناایستا و ایستا بدون آموزش دوباره کار کند. TASOM پیشنهادی با پنج توزیع گوناگون آزمایش و کارکرد آن با SOM پایه مقایسه می‌شود. خطای چندی کنش TASOM برای این آزمایشها همگی پایینتر از خطای SOM پایه است. افزون بر این، TASOM تندتر از SOM پایه همگرا می‌شود. این آزمایشها نشان می‌دهند که TASOM پایدار و همگرا است. شبکه TASOM در محیطهای ناایستا نیز آزمایش می‌شود به گونه‌ای که توزیع ورودی به طور کامل تغییر می‌کند. در چنین محیطهایی TASOM به تدریج وزنهای خود را از توزیع کهنه دور می‌کند و به سوی گرانیگاه‌های توزیع نو می‌کشاند. این ویژگی را می‌توانیم با حافظه خودمان بسنجیم که به تدریج حافظه کهنه را فراموش می‌کند و داده‌های ورودی نو را یاد می‌گیرد.

INTRODUCTION

SOM, which was developed by Kohonen [1], transforms input sample vectors of arbitrary dimensions to a one- or two- dimensional discrete map in an adaptive fashion. It has been used in applications such as vector quantization [2], texture segmentation [3], brain modeling [4], phonetic typewriter [5], and image compression [6].

The SOM uses a Hebb-like learning rule with time decreasing learning parameters. At the beginning of the training, the learning rate is set to a value close to unity. Then, it is decreased gradually during the training. The first phase of the algorithm

in which the topological ordering of the weight vectors $\mathbf{w}_j(n)$ takes place is called the ordering phase. The second phase of the algorithm, during which the weight vectors are updated to provide a good approximation of the input distribution, is called the convergence phase.

For topological ordering of the weight vectors to take place, the neighborhood function usually begins such that it includes all neurons in the network and then gradually shrinks with time. A good choice for the dependence of the learning-rate on time n is the exponential decay [4], described as $\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_1}\right)$ where τ_1 is a time constant and η_0 is the initial value of the learning-rate parameter [4,7]. The width $\sigma(n)$ of the Gaussian neighborhood function: $h_{j,i(\mathbf{x})}(n) = \exp(-\frac{d_{j,i}^2}{2\sigma^2(n)})$ is often described by: $\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_2}\right)$.

Again, τ_2 is a time constant and σ_0 is the initial value of σ at the beginning of the SOM algorithm.

In fact, these parameters are at their highest values at the beginning of learning. Then, they decrease with time so that the feature map stabilizes and learns the topographic map of the input samples. At the final step, the learning-rate parameter usually has a very small value and so does the neighborhood function. Therefore, the SOM algorithm cannot learn with adequate speed the new input samples that may be different in statistical characteristics from the previously learnt samples. In other words, the learning process is incapable of responding appropriately to a varied environment that embodies incoming samples. Even constant learning rates cannot deal with such environments [8].

In addition, the appropriate form of the neighborhood function and the method of determining the learning-rate parameter for the SOM are not known. In fact, these parameters are usually determined experimentally. However, some efforts have been made to resolve these problems. One [9] uses the model of Kalman filters to automatically adjust the learning parameters, which is valid only within the system model, and cannot adapt itself to varied environments. Another [10] assumes an individual neighborhood size, which is a function of distance between the input vector and the relevant weight vector, with no suggestion for the learning-rate parameter adjustment. Moreover, a vector quantization method has been introduced with classified learning rates for image compression [11]

A suggestion for resolving the aforementioned problem is to choose time-independent learning parameters, which change their values with the conditions of the incoming samples, and not with the elapse of time. These parameters should increase the capability of the SOM in dealing with varied environments, but should not decrease the speed of convergence of the SOM algorithm. The algorithm must also stay stable in such environments.

Some steps have been taken before [12,13]. The TASOM with neighborhood sets has been introduced in [14]. Moreover, the TASOM has been used for adaptive pattern classification [15]. In this paper, we propose a new version of the algorithm in which we use neighborhood functions for the TASOM. The performance of the proposed algorithm in approximating stationary environments is compared with that of the basic SOM, and several experiments in changing environments are also conducted.

The proposed SOM algorithm is described in the next section. Section 3 briefly mentions the expected distortion definition and its relation to the input approximation error of the SOM networks. Experimental results are presented in section 4, and concluding remarks form the final section of the paper.

THE PROPOSED SELF-ORGANIZING FEATURE MAP ALGORITHM

The learning parameters of the basic SOM are only a function of time n, and decrease gradually with time. The parameters are chosen this way to assure the stabilization of synaptic weights, with the assumption that the input sample vectors come from a specific stationary distribution. Therefore, there is no mechanism or understanding to find out whether the input distribution is changing or not.

Any fundamental change in the input distribution causes severe problems for the SOM, and the learning rule cannot change the synaptic weights of the network with adequate speed. On the other hand, since the neighborhood function does not follow changes in the environment, the neurons of the network are unable to modify the topographic map based on the changed input vectors, and thus the feature map cannot take the appropriate form. For more details, see [12,13].

We have proposed a modified SOM algorithm called TASOM that automatically adjusts the learning parameters, and incorporates possible changes of the input distribution in updating the synaptic weights. For this purpose, the learning rate of each neuron is considered to follow the values of a function of distance between the input vector and its synaptic weight vector. This way, the parameter will be changed independently for each neuron, and the number of these parameters will be equal to the number of output neurons. A similar updating rule is proposed to automatically adjust the width of the neighborhood function of each neuron. The width of each output neuron is considered to follow the distance between the neuron's synaptic weight vector and the weight vectors of its neighboring neurons. This width is used in the Gaussian neighborhood function, similar to that of the basic SOM, which was described earlier. The learning rate modification has been discussed in [12,13].

The basic SOM, as observed by [16], fails to provide a suitable topological ordering for the input distributions that are non-symmetric. They proposed to use a normalizing vector specific to each neuron for distance calculation between any input vector and the neuron's weight vector. These normalizing vectors are updated during the network training. The TASOM normalizes all distance calculations such that each distance calculation in the network algorithm is normalized by a scaling vector, which is composed of standard deviations of input vectors' components. An application of this normalization is seen in [14].

However, for input approximation, the main task is to lower quantization errors and topological ordering is less important. Therefore, in this paper, a symmetric scaling is employed in the TASOM algorithm, which is equal to the norm of standard deviation vector of input vectors.

The proposed TASOM may be summarized in eight steps as follows:

Initialization Choose some values for the initial weight vectors $\mathbf{w}_j(0)$, where $j = 1,2,...,N$; and N is the number of neurons in the lattice. The learning-rate parameters $\eta_j(0)$ should be initialized with values close to unity. The constant parameters α, β, α_s, and β_s can have any values between zero and one. The constant parameters s_f and s_g should be set to satisfy the application's needs. The neighborhood widths of the neighborhood function $\sigma_j(0)$ should be set to positive values greater than one. The scaling value $sl(0)$ should be set to any positive value, preferably one. The parameters $E_k(0)$ and $E2_k(0)$ may be initialized with some small random values. Neighboring neurons of any neuron i in a lattice is included in the set NH_i. In this paper, for any neuron i in a one-dimensional lattice N, $NH_i = \{i-1, i+1\}$, where $NH_N = \{N-1\}$ and $NH_1 = \{2\}$. Similarly, for any neuron i in a two-dimensional lattice $N=M\times M$, $NH_{(i1,i2)} = \{(i1-1,i2),(i1+1,i2),(i1,i2-1),(i1,i2+1)\}$ where

$NH_{(1,1)} = \{(1,2),(2,1)\}$

$NH_{(1,M)} = \{(1,M-1),(2,M)\}$

$NH_{(M,1)} = \{(M,2),(M-1,1)\}$

$NH_{(M,M)} = \{(M,M-1),(M-1,M)\}$

Sampling Draw a sample-input vector $\mathbf{x}$ from the input distribution.

Similarity Matching Find the best-matching or winning neuron $i(\mathbf{x})$ at time n, using the minimum-distance Euclidean norm as the matching measure:

$$\mathrm{i}(\mathbf{x}) = \arg_{\mathrm{j}} \min \left\| \mathbf{x}(\mathrm{n}) - \mathbf{w}_{\mathrm{j}}(\mathrm{n}) \right\|, \; j = 1,2,...,N \qquad (1)$$

where

$$\left\| \mathbf{x}(\mathrm{n}) - \mathbf{w}_{\mathrm{j}}(\mathrm{n}) \right\| = \left(\sum_{\mathrm{k}} \left((\mathrm{x}_{\mathrm{k}}(\mathrm{n}) - \mathrm{w}_{\mathrm{j,k}}(\mathrm{n}) \right)^2 \right)^{\frac{1}{2}} \qquad (2)$$

Updating The Neighborhood Widths Adjust the neighborhood width of the winning neuron

$i(\mathbf{x})$ by the following equation:

$$\sigma_i(n+1) = \sigma_i(n) + \beta\left(g\left(\frac{1}{s_g \,.\, sl(n).\#(NH_i)} \sum_{j \in NH_i} \| w_i(n) - w_j(n) \|\right) - \sigma_i(n)\right) \quad (3)$$

where the function #(.) gives the cardinality of a set. The neighborhood widths of the other neurons do not change.

Updating The Learning-Rate Parameters Adjust the learning-rate parameters $\eta_j(n)$ of all neurons in the network by:

$$\eta_j(n+1) = \eta_j(n) + \alpha\left(f\left(\frac{1}{s_f \,.\, sl(n)} \| x(n) - w_j(n) \|\right) - \eta_j(n)\right) \quad \text{for all j} \quad (4)$$

Updating The Synaptic Weights Adjust the synaptic weight vectors of all output neurons in the network using the following update rule:

$$w_j(n+1) = w_j(n) + \eta_j(n+1)\, h_{j,i(x)}(n+1)[x(n) - w_j(n)] \quad \text{for all j} \quad (5)$$

where $\eta_j(n+1)$ is the learning-rate parameter, and $h_{j,i(\mathbf{x})}(n+1)$ is the Gaussian neighborhood function centered on the wining neuron $i(\mathbf{x})$.

Updating The Scaling Value Adjust the scaling vector $sl(n+1)$ with the following equations:

$$sl(n+1) = \sqrt{\sum_k s_k(n+1)} \quad (6)$$

where

$$s_k(n+1) = \left(E2_k(n+1) - E_k(n+1)^2\right)^+ \quad (7)$$

$$E2_k(n+1) = E2_k(n) + \alpha_s\left(x_k^2(n) - E2_k(n)\right) \quad (8)$$

and

$$E_k(n+1) = E_k(n) + \beta_s\left(x_k(n) - E_k(n)\right) \quad (9)$$

The function $(z)^+ = z$ if $z > 0$; otherwise it is zero.

Continuation Continue with step 2. For function $f(.)$ we should have $f(0) = 0$, $0 \le f(z) \le 1$ and $\frac{df(z)}{dz} \ge 0$ for positive values of z. Similarly, $g(0) = 0$, $0 \le g(z) < N$ for one-dimensional lattices of N neurons and $0 \le g(z) < M\sqrt{2}$ for two-dimensional lattices of M×M neurons, and $\frac{dg(z)}{dz} \ge 0$ for positive values of z. Examples of functions $f(.)$ and $g(.)$ are $f_1(z) = 1 - \frac{1}{1+z}$ and $g_1(z) = (N-1)(1 - \frac{1}{1+z})$ or $(M\sqrt{2} - 1)(1 - \frac{1}{1+z})$ respectively.

It should be noted that in this algorithm, the initialization step of the algorithm is used only once during the lifetime of the network.

There are some other self-organizing maps that may be used for non-stationary and changing environments [17-22]. These SOM algorithms add or delete neurons in response to their changing environments. This is different from our proposed TASOM in which no neuron deletion or addition is needed and the network adaptation is achieved through dynamic parameter adjustment.

EXPECTED DISTORTION AND FEATURE MAP

The weight vectors of a SOM network represent a non-linear mapping Φ, called the feature map, from the input space X onto the discrete output space A, as shown by neuron $i(\mathbf{x})$:

$$\Phi : X \to A \quad (10)$$

Given an input vector **x**, the feature map Φ maps it to the winning

$$i(\mathbf{x}) = \arg_j \min \left\| \mathbf{x} - \mathbf{w}_j \right\| \qquad \textit{for all } j \quad (11)$$

The weight vector $\mathbf{w}_{i(\mathbf{x})}$ may be viewed as a pointer of the winning neuron $i(\mathbf{x})$ to the input space X. In fact, the weight vectors of an SOM network represent a one-to-many mapping from the output space A to the input space X. This means that a large set of input vectors may be

represented by a smaller number of weight vectors of the network in order to provide a good approximation of the input space. An optimum solution may be found by minimizing the following expected distortion:

$$D = \frac{1}{2}\int_{-\infty}^{+\infty} d\mathbf{x}\, f_X(\mathbf{x})\, d(\mathbf{x}, \mathbf{w}_{i(\mathbf{x})}) \qquad (12)$$

This integration is over the entire input space. The probability density function of the input space is represented by $f_X(\mathbf{x})$ from which the input samples are selected. A popular choice for the distortion measure $d(\mathbf{x}, \mathbf{w}_{i(\mathbf{x})})$ is the square of the Euclidean distance between the input vector $\boldsymbol{x}$ and the wining weight vector $\mathbf{w}_{i(\mathbf{x})}$ for that input vector. Consequently, the expected distortion D may be rewritten as

$$D = \frac{1}{2}\int_{-\infty}^{+\infty} d\mathbf{x}\, f_X(\mathbf{x}) \left\| \mathbf{x} - \mathbf{w}_{i(\mathbf{x})} \right\|^2 \qquad (13)$$

The SOM networks may be viewed as vector quantization algorithms trying to provide good approximations to their input spaces.

EXPERIMENTAL RESULTS

To test the performance of the proposed TASOM for the input distribution approximation, several experiments are developed in stationary and non-stationary environments. For this purpose, five different two-dimensional input distributions including Uniform distribution, Gaussian distribution, Exponential distribution, Laplacian distribution, and Rayleigh distribution are simulated. Each random vector of any of these distributions is composed of two independent and identically distributed one-dimensional random values. Specifically, we may say that the probability density function (pdf) of each distribution is $f_{X,Y}(x,y) = f_X(x) f_X(y)$ for the Uniform distribution, we have $f_X(x) = \begin{cases} 1 & \text{if } 0 \le x \le 1 \\ 0 & \text{otherwi} \end{cases}$ for the Gaussian distribution, $f_X(x) = \dfrac{e^{-(x-m)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}$ where $-\infty < x < \infty$, $\sigma > 0$, and m and σ are the mean and variance of random variable X, respectively. The pdf of Exponential distribution is $f_X(x) = \lambda e^{-\lambda x}$ where $x \ge 0$ and $\lambda > 0$. For Laplacian distribution we have $f_X(x) = \dfrac{\alpha}{2} e^{-\alpha|x|}$ where $-\infty < x < \infty$ and $\alpha > 0$. Finally, the pdf of Rayleigh distribution is $f_X(x) = \dfrac{x}{\alpha^2} e^{-x/2\alpha^2}$ where $x \ge 0$ and $\alpha > 0$. In this paper, $m = 0$ and $\sigma = 1$ are set for the Gaussian distribution. For the Exponential distribution, we set $\lambda = 1$. For the Laplacian and Rayleigh distribuitons, we set $\alpha = 1$.

The five distributions with 10000 points are shown in Figures 1(a)-(e) for the Uniform, Gaussian, Exponential, Laplacian, and Rayleigh distributions, respectively. The test is carried out with 50 neurons forming one-dimensional lattices of neurons for both the basic SOM and TASOM networks. The expected approximated distortion of Equation 13 is used to compare the approximation error of the networks. The variations of the quantization error of the basic SOM and TASOM networks for the five distributions versus the number of iterations are depicted for every 1000 samples in Figures 2(a)-(e), where dashed lines represent the SOM's behavior and solid lines represent the TASOM's. It should be mentioned that different values for the basic SOM parameters were tested, and the results given here are for those giving the best quantization performance that could be achieved. In this paper, for the SOM network, the initial learning-rate and the initial width of neighborhood function parameters are $\eta_0 = 0.9$ and $\sigma_0 = N$, respectively. Moreover, the time constants are $\tau_1 = \tau_2 = 2000$. For the TASOM network, we use $\alpha = \beta = 0.1$ $\alpha_s = \beta_s = 0.01$ $s_f = s_g = 15$ $\sigma_j(0) = N$ $\eta_j(0) = 1$.

The initial weight vector $\mathbf{w}_j(0)$, the scaling value $sl(0)$, and the parameters $E_k(0)$ and $E2_k(0)$ are small positive values which are randomly chosen.

The first point which is extracted from Figures 2(a)-(e) is that the TASOM converges much faster than the basic SOM. In fact, fewer iterations are needed for the TASOM to approximate input space with enough accuracy. This is due to adaptive

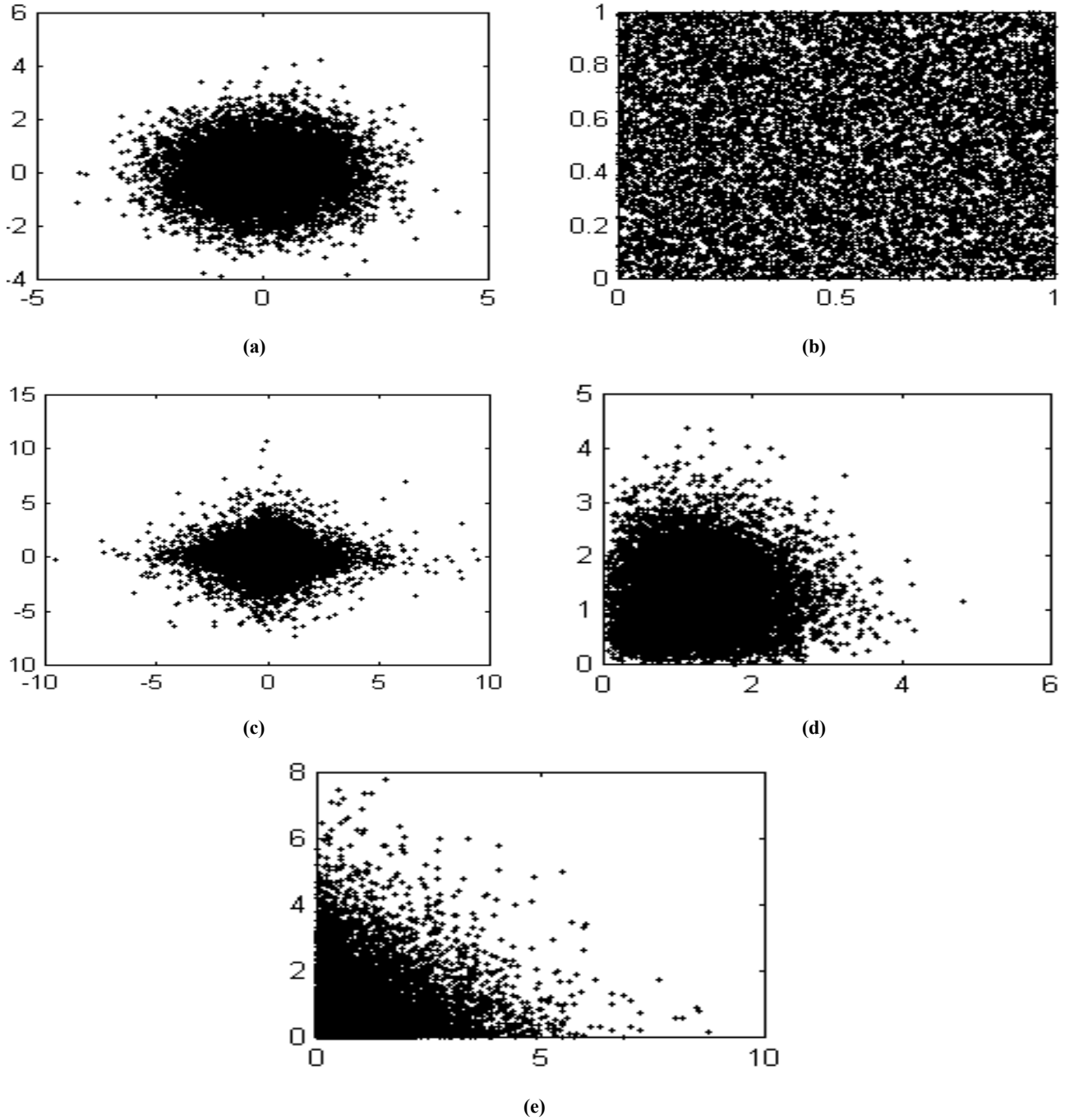

Figure 1. The five distributions used in the experiments each one represented by 10000 points. (a) the Gaussian distribution, (b) the Uniform distribution, (c) the Laplacian distribution, (d) the Exponential distribution, and (e) the Rayleigh distribution.

adjustment of learning rates and neighborhood functions of neurons in the TASOM in which parameters are adjusted according to the environmental conditions. The second point is that the TASOM converges with less quantization error than the basic SOM. This property is partly due to using separate learning rates and neighborhood functions for neurons. With separate learning rates and neighborhood functions, the TASOM network is able to locate neurons in the environment with

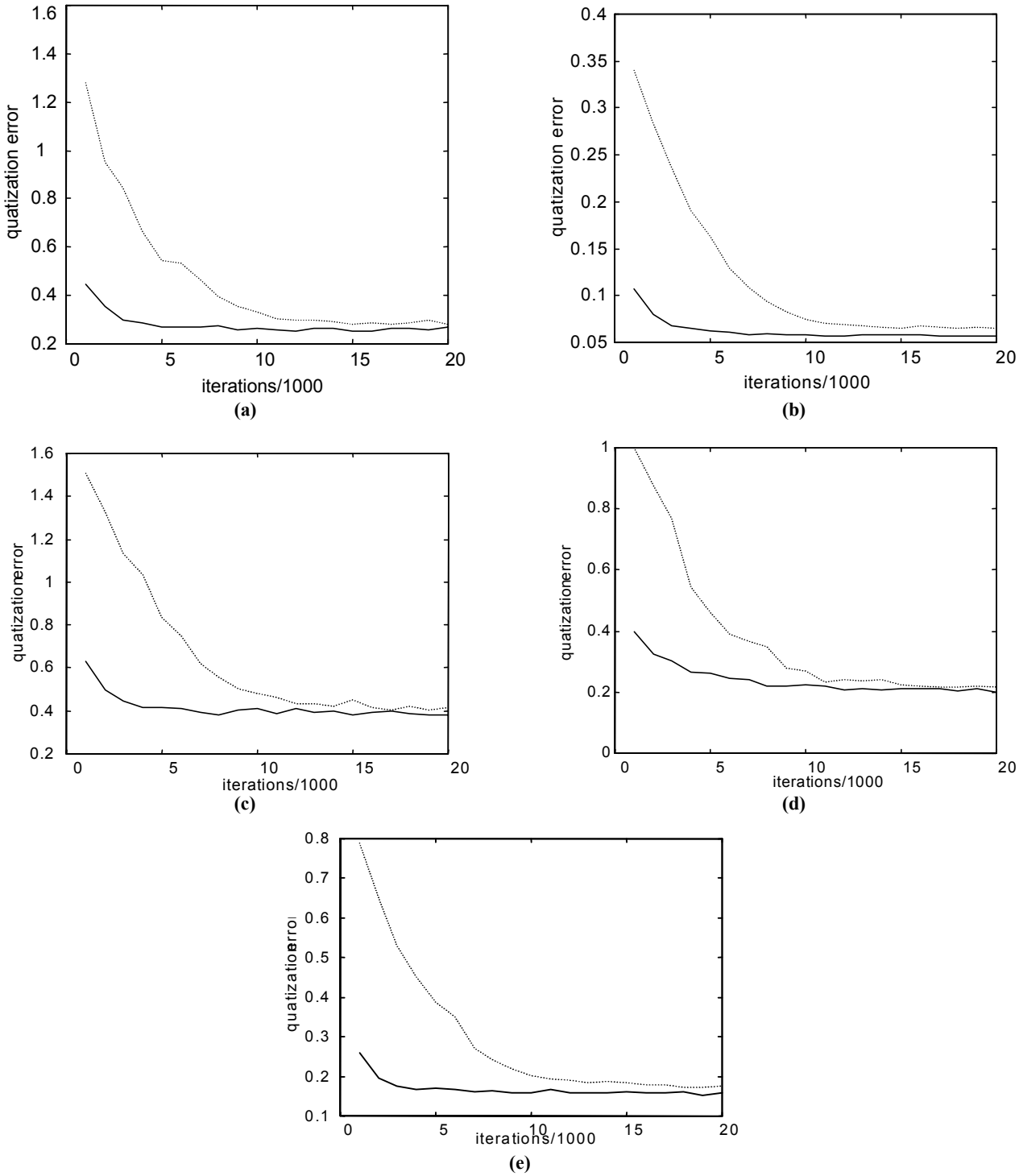

Figure 2. The quantization error of TASOM and the basic SOM as iteration goes on where dotted lines represents the basic SOM and solid lines represents the TASOM. (a) for Gaussian distribution, (b) for Uniform distribution, (c) for Laplacian distribution, (d) for Exponential distribution, and (e) for Rayleigh distribution.

more flexibility than the basic SOM, which has only one learning rate and neighborhood function.

Now consider the TASOM network of Figure 2(b) trained by the Uniform distribution. This network undergoes some change in its input space distribution. The first change is to Gaussian

TABLE1. The Quantization Error of the Basic SOM, the TASOM with Initialization, and the TASOM Without Initialization for the Five Input Distributions.

SOM \ Distribution	Uniform	Gaussian	Exponential	Laplacian	Rayleigh
Basic SOM	0.0651	0.2866	0.2283	0.4108	0.1788
TASOM with initialization	0.0592	0.2714	0.2182	0.4053	0.1662
TASOM without initialization	--------	0.2745	0.2090	0.4038	0.1678

distribution. Then it changes to Exponential, then Rayleigh, and finally Laplacian distributions. In fact, the TASOM network used in this experiment undergoes a very changing environment, which endure five complete changes in its input space while the TASOM is initialized only one time during its lifetime, irrespective of the changes in the environment. The quantization errors of the TASOM network in this case for the five distributions are shown in the third row of Table 1. For comparison, the quantization errors for the basic SOM and the TASOM networks are also presented and shown in the first and second rows of Table 1, respectively. The initial values of the parameters of the two networks are the same as those used in the previous experiments.

The TASOM is thus tested in two different cases. In one case, the initialization step of the TASOM algorithm is executed for each new environment. We call this case of using TASOM as "TASOM with initialization". This case is surely stationary, since we have to initialize the weight vectors and other parameters of the TASOM network for learning its environment, and it is assumed that the statistical characteristics of the environment do not change with time. The basic SOM can be used only in this case.

The other case of using the TASOM network is a non-stationary case in which it is assumed the environment may be faced with some changes in its statistical characteristics as time goes on. In this case, the initialization step is used only the first time that the TASOM network is trained. When the distribution of the environment changes, the network has to learn the new environment with the values of the weights and parameters that have been learned with its former environment. The ability of TASOM in learning the new environment is solely dependent on the learning rate and neighborhood width updating proposed in the TASOM algorithm. We call this case "TASOM without initialization".

According to Table 1, most of the time, the errors for the TASOM without initialization is slightly lower than the case where the TASOM with initialization is used. The basic SOM produces higher errors than the TASOM with initialization and TASOM without initialization. In other words, the TASOM network performs well in non-stationary as well as stationary environments.

The TASOM network uses separate learning rate and neighborhood width for each neuron of the network. So, the neurons of TASOM are more flexible than the basic SOM to represent the input vectors. This way, the TASOM networks obtain better performance than the basic SOM as shown in Table 1.

To assure that the TASOM has preserved topological ordering in the mentioned experiments, the converged weights of TASOM for the five distributions with initialization for each distribution, the weights of the SOM for the five distributions, and the converged weights of the TASOM for the four distributions without initialization are all presented in Figures 3(a)-(e), 4(a)-(e), and 5(a)-(d), respectively. According to these Figures, the TASOM always preserves topological ordering.

It may be interesting to see the stages during which the input distribution changes while the TASOM moves its weights to gradually represent the most recent distribution. Assume that the TASOM network converged to represent the two-dimensional Uniform distribution in the region $[0,1]\times[0,1]$ faces a new Uniform input distribution in the region $[4,5]\times[4,5]$. Figure 3(b) shows the TASOM weights for the former Uniform distribution. Figure 6(a) demonstrates the intermediary stage of convergence of the TASOM weights toward the new Uniform

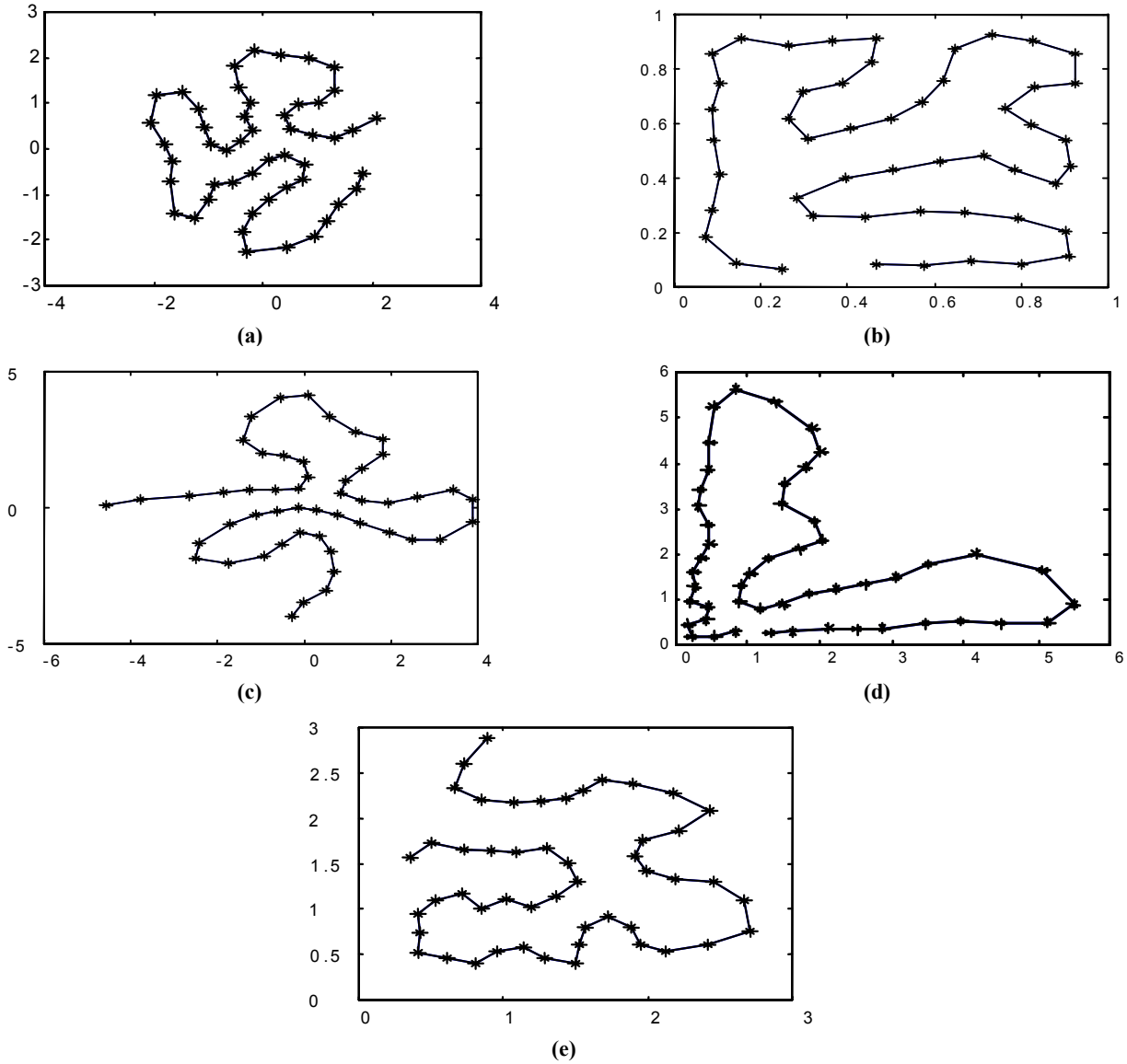

Figure 3. The topographic map of the proposed TASOM. (a) for Gaussian. (b) Uniform. (c) Laplacian. (d) Exponential. (e) Rayleigh.

distribution. At the end, Figure 6(b) shows the converged weights for the new Uniform distribution, which not only preserves topological ordering, but also distributes uniformly in the new input space and completely covers it.

It seems that The TASOM moves its weights in topological order, no matter what happens in the environment. Moreover, it adapts its weights to the new environment in an elegant fashion. This means that the weights gradually leave the old distribution and move gradually toward the clusters of the new distribution. This phenomenon reminds us of the memory of our brains. The past memory remains refresh as long as new sensory data emphasize that. If new sensory inputs different from those of the past ones are received, our memory tries to represent and memorize the recent data. However, forgetting the past memory and memorizing the new one is also gradual in our brain, we don't forget past memory and we also don't memorize the new one suddenly.

CONCLUDING REMARKS

The decreasing time-dependent learning

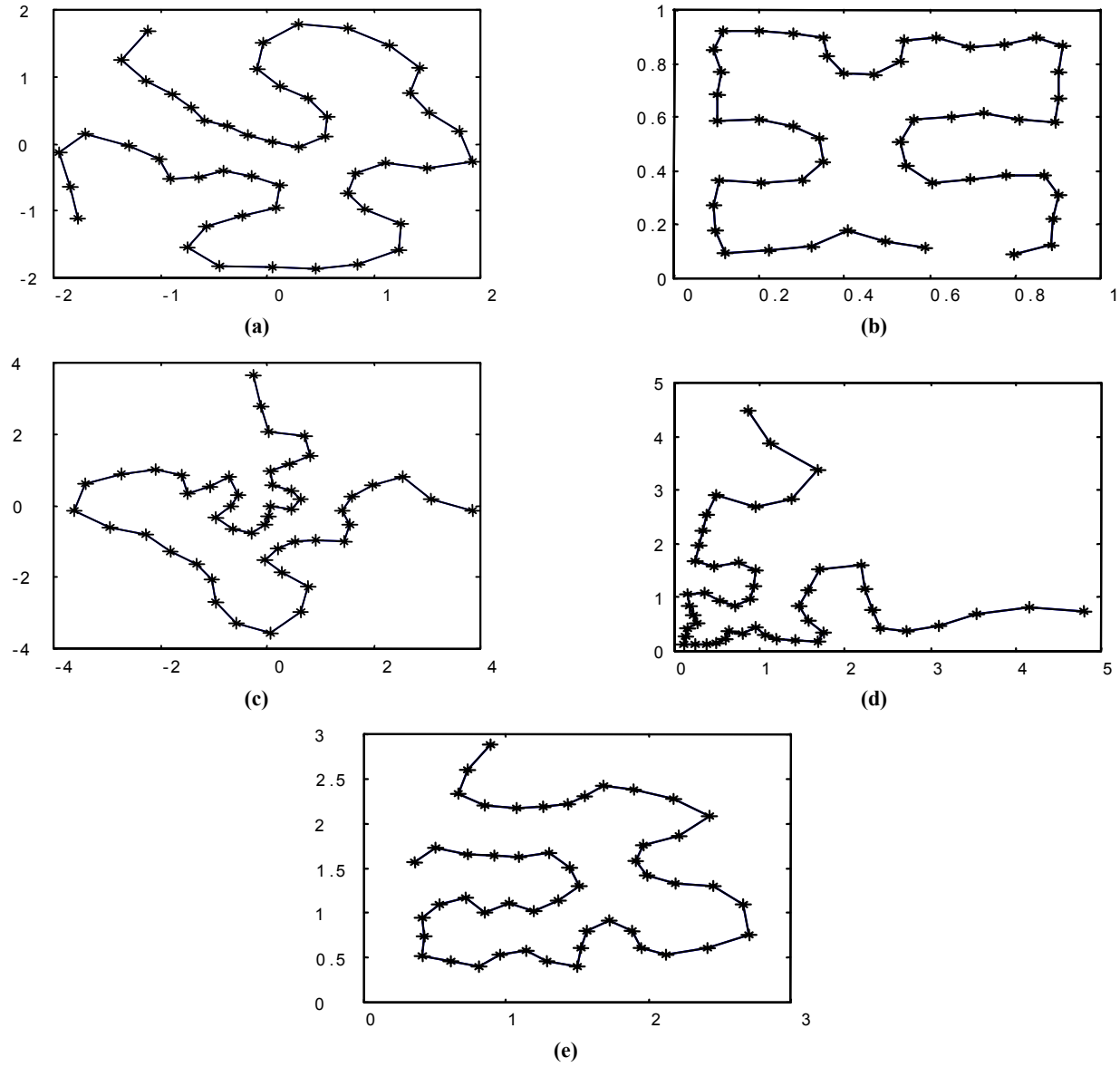

Figure 4. The topographic map of the basic SOM. (a) for Gaussian. (b) Uniform. (c) Laplacian. (d) Exponential. (e) Rayleigh,

parameters of the SOM lower the incremental learning capability of the algorithm in response to varied environments. In this paper, a modified SOM algorithm called "TASOM" with neighborhood function was proposed to automatically adjust the learning rate parameter and the neighborhood function of each output neuron. Each output neuron is assumed to have its own learning rate and neighborhood function, which are updated repeatedly in the proposed SOM algorithm in response to new input samples.

The proposed TASOM was tested in stationary environments, and was compared to the basic SOM for input approximation. According to these tests, the TASOM can be claimed to converge with fewer iterations than the basic SOM. Moreover, the quantization errors of the TASOM are lower than the errors of the basic SOM. The TASOM was also tested in non-stationary environments in which the input distribution completely changes to another distribution. There is no need to reinitialize the TASOM in response to changes in the input distribution. Experimental results confirm this claim. In the TASOM, learning of the new

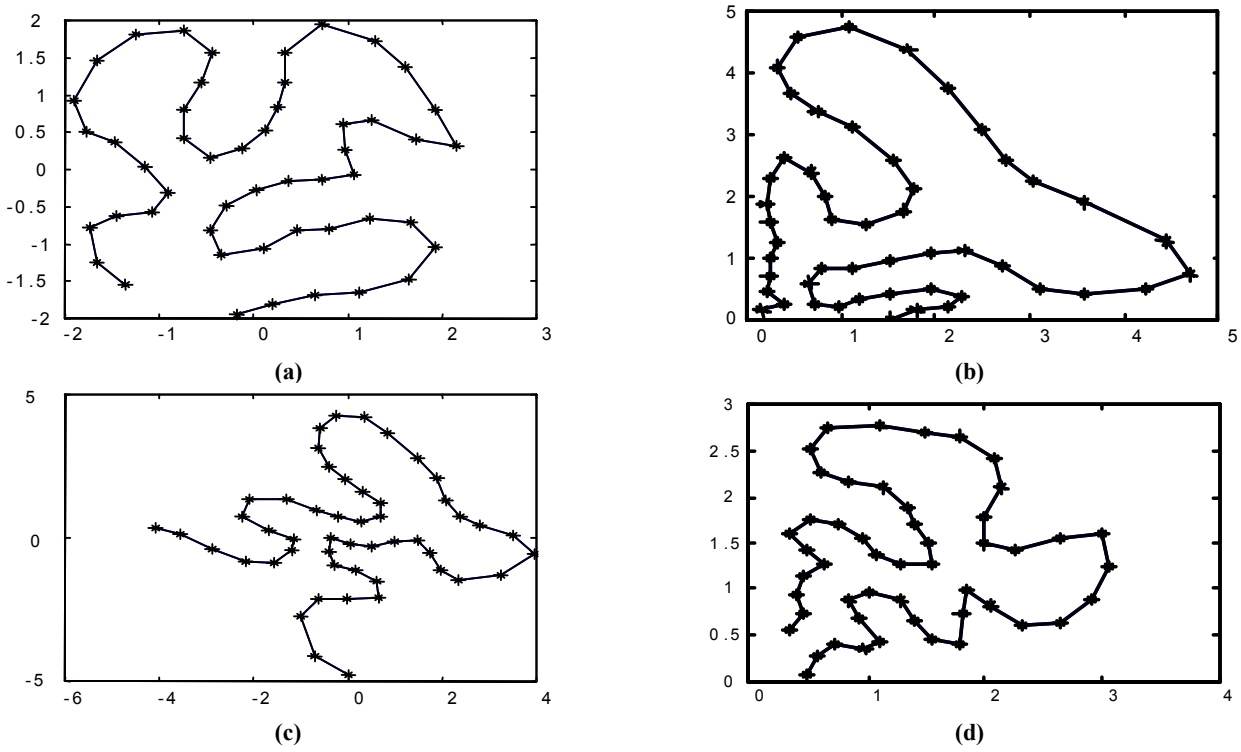

Figure 5. (a) The TASOM network of Figure 2(b) trained by the Uniform distribution faces the Gaussian distribution. (b) The TASOM network of Figure 5(a) faces the Exponential distribution. (c) The TASOM network of Figure 5(b) faces the Laplacian distribution. (d) The TASOM network of Figure 5(c) faces the Rayleigh distribution.

distribution is gradual, similar to the human memory, which constantly and gradually updates itself to the more recent data while forgets the old memory in the long run.

These experiments suggest that the TASOM is suitable for both stationary and non-stationary environments. In fact, the TASOM learns continuously from its environment, and only a one- time initialization is needed for it to work in its possibly changing environment.

REFERENCES

1. Kohonen, T., "Self-Organized Formation of Topologically Correct Feature Maps", ***Biological Cybernetics***, Vol. 43, (1982), 59-69.
2. Chiuch, T. D., Tang, T. T. and Chen, L. G., "Vector Quantization Using Tree-Structured Self-Organizing Feature Maps", In Applications of Neural Networks to Telecommunications (J. Alspector, R. Goodman, and T. X. Brown, eds.), Hillsdale, NJ: Lawrence Erlbaum, (1993), 259-265.
3. Oja, E., "Self-Organizing Maps and Computer Vision", In Neural Networks for Perception (H.Wechsler, ed.), San Diego, CA: Academic Press, Vol. 1, (1992), 368-385.

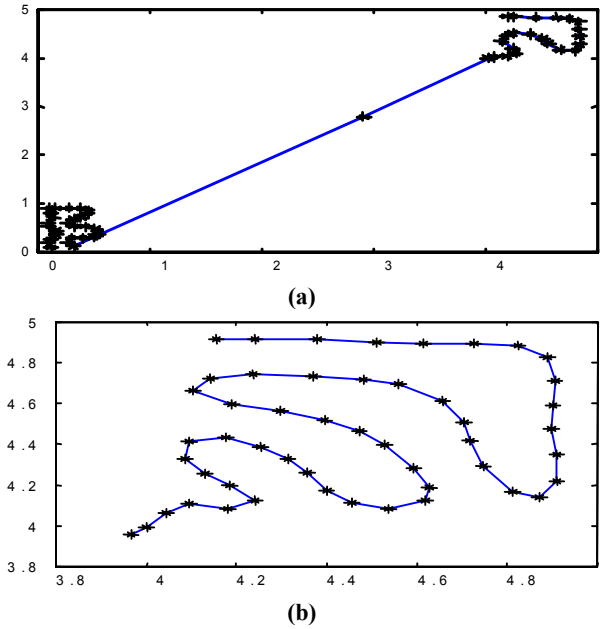

Figure 6. (a) The topographic map of the TASOM in an intermediary stage during which the input distribution undergoes translation by the vector (4,4). (b) The final topographic map of the TASOM with its fully converged weights representing the new distribution.

4. Ritter, H. J., Martinez, T. M. and Schulten, K. J., "Neural Computation and Self-Organizing Maps: An Introduction, Reading MA", Addison Wesley, (1992).
5. Kohonen, T., "The Neural Phonetic Typewriter," ***Computer***, Vol. 21, (1988), 11-22.
6. Amerijckx, C., Verleysen, M., Thissen, P. and Legat, J., "Image Compression by Self-Organized Kohonen Map", ***IEEE Trans. on Neural Networks***, Vol. 9, No. 3, (1998), 503-507.
7. Haykin, S., "Neural Networks", Prentice Hall, New Jersey, (1999).
8. Heskes, T. and Kappen, B., "Neural Networks Learning in a Changing Environment", ***IJCNN***, (1991), 823-828.
9. Haese, K., "Self-Organizing Feature Maps With Self-Adjusting Learning Parameters", ***IEEE Trans. On Neural Networks***, Vol. 9, No. 6, (1998), 1270-1278.
10. Maillard, E and Gresser, J., "Reduced Risk of Kohonen's Feature Map Non-Convergence by an Individual Size of the Neighborhood", ***IEEE International Conference on Neural Networks***, Vol.2, (1994), 704-707.
11. Kim, C. W., Cho, S. and Lee, C. W., "Fast Competitive Learning With Classified Learning Rates for Vector Quantization", ***Signal Processing Image Communication***, Vol. 6, No. 6, (1995), 499-505.
12. Shah-Hosseini, H. and Safabakhsh, R., "Automatic Adjustment of Learning Rates of the Self-Organizing Feature Map", Accepted for Publication in Scientia Iranica.
13. Shah-Hosseini, H. and Safabakhsh, R., "A Learning Rule Modification in the Self-Organizing Feature Map Algorithm," ***Fourth Int. CSI Computer Conference***, Tehran, Iran, (1999), 1-9.
14. Shah-Hosseini, H. and Safabakhsh, R., "TASOM: The Time Adaptive Self-Organizing Map", ***Proc. Int'l Conf. Information Technology Coding and Computing, IEEE Press***, Las Vegas, Nevada, (2000), 422-427.
15. Shah-Hosseini, H. and Safabakhsh, R., "Pattern Classification by the Time Adaptive Self-Organizing Map", ***Proc. IEEE Int'l Conf. Electronics, Circuits and Systems***, Beirut, Lebanon, (December 17-20, 2000), 495-498.
16. Kangas, J. A., Kohonen, T. and Laaksonen, J. T., "Variants of Self-Organizing Maps", ***IEEE Trans. on Neural Networks***, Vol. 1, No. 1, (1990), 93-99.
17 Trautmann, T. and Denoeux, T., "Comparison of Dynamic Feature Map Models for Environmental Monitoring", ***IEEE Int. Conf. On Neural Networks***, Vol.1, (1995), 73-78.
18. Datta, A. Pal, T. and Parui, S. K., "A Modified Self-Organizing Neural Net for Shape Extraction," Neurocomputing, Vol. 14, No. 1, (1997), 3-14.
19. Chau-Yun, H. and Hwai-En, W., "An Improved Algorithm for Kohonen's Self-Organizing Feature Maps", ***IEEE Int. Conf. On Circuits and Systems***, Vol. 1, 328-331.
20. Blackmore, J. and Miikulainen, R., "Incremental Grid Growing: Encoding High Dimensional Structures to a Two-Dimensional Feature Map", IEEE International Conference on Neural Networks, Vol. 1, (1993), 450-455.
21. Fritzke, B., "Growing Cell Structures-a Self-Organizing Network for Unsupervised and Supervised Learning", ***Neural Networks***, Vol. 7, (1994), 1441-1460.
22 Villmann, TH. and Baur, H. U., "Applications of the Growing Self-Organizing Map", Neurocomputing, Vol. 21, No. 1-3, (1998), 91-100.