Topology Preserving Neural Networks that Achieve a Prescribed Feature Map Probability Density Distribution

Jongeun Choi and Roberto Horowitz

Abstract— In this paper, a new learning law for one-dimensional topology preserving neural networks is presented in which the output weights of the neural network converge to a set that produces a predefined winning neuron coordinate probability distribution, when the probability density function of the input signal is unknown and not necessarily uniform. The learning algorithm also produces an orientation preserving homeomorphic function from the known neural coordinate domain to the unknown input signal space, which maps a predefined neural coordinate probability density function into the unknown probability density function of the input signal. The convergence properties of the proposed learning algorithm are analyzed using the ODE approach and verified by a simulation study.

I. INTRODUCTION

Identification of homeomorphic functions is a frequently encountered problem in many signal processing, pattern recognition, self-organizing and computational topology applications. The existence of a homeomorphism $u : \mathbf{X}_\mathbb{R} \to \mathbf{U}_\mathbb{R}$ implies the topological equivalence between the domain $\mathbf{X}_\mathbb{R}$ and co-domain $\mathbf{U}_\mathbb{R}$. For example, [1] developed a nonlinear constrained optimization algorithm that enables to track large nonlinear deformations while preserving the topology for medical images. A coordinate chart on a manifold also utilizes a homeomorphic function from an open subset of the manifold to the local coordinates [2]. Homeomorphic manifold learning is also useful in many other computer vision applications such as image contour trackers and pattern recognition schemes [3].

Spatial discretizations of homeomorphisms are often called topology preserving feature maps [4], [5]. In his classical paper, Kohonen [4] introduced a class of self-organizing adaptive systems that are capable of forming one- or two-dimensional feature maps of the input signal domain. Kohonen formulated an adaptive learning law that allows a set of N real vectors $\hat{\mathbf{U}}_\mathbb{Z} = \{\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_N\}$, where $\hat{u}_i \in \mathbb{R}^n$ is the vector associated with the ith node of a neural network, to form an ordered image of a random input variable $u \in \mathbf{U}_\mathbb{R} \subset \mathbb{R}^n$, which has a stationary probability density function $f_U(u)$. The network is trained from a sequence of samples of the input variable $u(t)$. The ordered image formed after convergence is commonly denoted as a topology preserving feature map, as it preserves some notion of the proximity of the input signal features.

Jongeun Choi is a PhD candidate in the Department of Mechanical Engineering, University of California at Berkeley jchoi@newton.berkeley.edu

Roberto Horowitz is a Professor in the Department of Mechanical Engineering, University of California at Berkeley horowitz@me.berkeley.edu

Kohonen also noted the importance that the probability distribution of the winning neuron, which is induced by the neural network's feature map, should be equiprobable, i.e., all neurons in the network should have an equal chance of becoming the winner. Kohonen's algorithm is closely related to vector quantization in information theory [6].

The convergence properties of Kohonen's self-organizing algorithm have been investigated by several researchers [4], [7], [8], [9], [10]. For the one-dimensional case, Kohonen himself [4] presented a proof of the ordering properties of his algorithm and showed that it converges when the input signal probability distribution, $f_U(u)$, is uniform. Ritter and Schulten [7] derived a Fokker-Planck equation to describe the learning process in the vicinity of equilibrium maps and investigated the stability of an equilibrium map for the case of a uniformly distributed input. The convergence of one-dimensional Kohonen algorithms was investigated in [9], [10] when the input signal was not uniform, and an a.s. convergence was proven when the input distribution admits a log-concave density.

In this paper, a new one-dimensional topology-preserving neural network is presented, whose output weights converge to a set that produces a predefined feature map probability distribution, when the probability density function (pdf) of the input signal is unknown but globally Lipschitz continuous. Moreover, the learning algorithm will produce the required orientation preserving homeomorphic function from a known domain to the unknown support of the input signal's pdf, which maps a predefined pdf into the unknown pdf of the input signal. A related algorithm, which used a conscience mechanism to achieve the equiprobability of the feature map was presented in [11], [12].

Several benefits result from our new approach. First, our new mathematical formulation of the *optimal codebook vector* enables us to control the feature map's probability distribution according to an arbitrarily assigned target probability. The feature map's probability does not necessarily have to be equiprobable [4] (unconditional information-theoretic entropy maximization). Second, the network is able to deal with non-uniform input distributions requiring only a mild global Lipschitz continuity condition and without employing a conscience mechanism in the learning law. Third, the orientation of the topology preserving map can be controlled, i.e., we can specify that the topology preserving map either be orientation preserving, orientation reversing map or simply a topology preserving map. Finally, the network produces a smooth homeomorphism $\hat{u}(x)$, which spatially discretizes to the topology preserving map.

II. PROBLEM STATEMENT

A. An optimal quantizer with a target feature map probability distribution

Suppose that we can collect time samples of the stationary random sequence $u : \mathbb{Z}_+ \to \mathbf{U}_{\mathbb{R}}$, where $\mathbf{U}_{\mathbb{R}} = [u_{min}, u_{max}] \subset \mathbb{R}$ is a finite interval and u is randomly distributed in $\mathbf{U}_{\mathbb{R}}$ with a probability density function (pdf) $f_U : \mathbf{U}_{\mathbb{R}} \to \mathbb{R}_+$. The support of $f_U(u)$ is assumed to be connected and finite. $u(t)$ denotes a sample of the random sequence u at time index t.

We now introduce a fictitious random sequence $x : \mathbb{Z}_+ \to \mathbf{X}_{\mathbb{R}} \subset \mathbb{R}$, where $\mathbf{X}_{\mathbb{R}} = [x_{min}, x_{max}]$ is a *known* finite interval and assume that u is a *homeomorphic function* of x, i.e., there exists a continuous map $u : \mathbf{X}_{\mathbb{R}} \to \mathbf{U}_{\mathbb{R}}$, which has a continuous inverse, such that the time samples of the input signal, $u(t)$, are given by $u(t) = u(x(t))$. Moreover, the probability density function of x, $f_X : \mathbf{X}_{\mathbb{R}} \to \mathbb{R}_+$, is given by

$$f_X(x) = \left| \frac{\partial u}{\partial x} \right| f_U(u(x)) \quad \text{for all } x \in \mathbf{X}_{\mathbb{R}}. \quad (1)$$

Notice that the homeomorphism

$$u : \mathbf{X}_{\mathbb{R}} \to \mathbf{U}_{\mathbb{R}},$$

which will be subsequently referred to as the reference or target map, induces the pdf $f_X(x)$ from the unknown pdf $f_U(u)$ via Eq. (1), and is uniquely specified up to its orientation (e.g., it is either orientation preserving or orientation reversing), if both pdfs are specified.

Let us define a set of N *output levels* of a scalar quantizer, which are labeled by the set of ordered indexes $\mathbf{X}_{\mathbb{Z}}$. Without loss of generality, we assume that $\mathbf{X}_{\mathbb{Z}} = \{1, \cdots, N\}$ and that $\mathbf{X}_{\mathbb{R}} = [1/2, N + 1/2]$, so that $\mathbf{X}_{\mathbb{Z}} \subset \mathbf{X}_{\mathbb{R}}$. We also define the quantizer codebook [6] as the set of N output levels

$$\mathbf{U}_{\mathbb{Z}} = \{u_\gamma \in \mathbb{R} \mid \gamma \in \mathbf{X}_{\mathbb{Z}}\}, \quad (2)$$

and a codebook vector

$$\bar{u} = [u_1 \cdots u_N]^T \in \mathbb{R}^N, \quad (3)$$

where u_γ is the quantizer output level with the index γ and the quantizer's *feature map* [4] $f(u, \bar{u}) : \mathbf{U}_{\mathbb{R}} \times \mathbb{R}^N \to \mathbf{X}_{\mathbb{Z}}$

$$f(u, \bar{u}) = \arg \left\{ \min_{\gamma \in \mathbf{X}_{\mathbb{Z}}} |u - u_\gamma| \right\}. \quad (4)$$

Notice that $f(u(t), \bar{u})$ returns the coordinate of the output level that most closely approximates the current sample $u(t)$.

The quantizer takes as its input the time sequence $u(t)$ and produces as its output a time sequence of *winning coordinates* $w : \mathbb{Z}_+ \to \mathbf{X}_{\mathbb{Z}}$, given by

$$w(t) = w(u(t)) = f(u(t), \bar{u}), \quad (5)$$

and a time sequence of estimates of $u(t)$ given by $u_w : \mathbb{Z}_+ \to \mathbb{R}$. Notice that the sequence $w(t)$ is a random sequence with a discrete probability distribution that is induced by the pdf $f_U(u)$ and the feature map $f(u, \bar{u})$. The feature map probability distribution vector is defined by

$$p = [p_1, \cdots p_N]^T \in \mathbb{R}_+^N, \quad (6)$$

where $p_\gamma = Pr[f(u, \bar{u}) = \gamma] \geq 0$ is the probability associated with the quantizer coordinate γ and $\sum_{\gamma=1}^N p_\gamma = 1$. Moreover, p_γ provides a measure of how frequently the γ coordinate in the quantizer is being used.

In many cases it is desired that *all* quantizer coordinates be used with the same level of frequency, i.e., $p_\gamma = 1/N$. This is often referred to as the feature map $f(u)$ in Eq. (5) being *equiprobable* [4]. However, in some cases we may not desire equiprobability and instead prefer that some specific coordinates be used more frequently than others.

We now define the *desired* probability distribution of the winning coordinate $w(t)$ in Eq. (5) by

$$p^o = [p_1^o, \cdots p_N^o]^T \in \mathbb{R}_+^N, \qquad \sum_{\gamma=1}^N p_\gamma^o = 1, \quad (7)$$

and assume that this can be achieved by a codebook with size N. p^o will also be referred to as the *target feature map probability distribution*. p^o must be produced by a quantizer that has an *optimal codebook*

$$\mathbf{U}_{\mathbb{Z}}^o = \{u_\gamma^o = u(\gamma) \mid \gamma \in \mathbf{X}_{\mathbb{Z}}\}, \quad (8)$$

and an *optimal codebook vector*

$$u^o = [u_1^o \cdots u_N^o]^T = [u(1) \cdots u(N)]^T \in \mathbb{R}^N, \quad (9)$$

where $u : \mathbf{X}_{\mathbb{R}} \to \mathbf{U}_{\mathbb{R}}$ is a homeomorphism. We will also impose an additional constraint that the optimal codebook vector u^o in Eq. (8) satisfies

$$
\begin{aligned}
p_\gamma^o &= \int_{\frac{u_{\gamma-1}^o + u_\gamma^o}{2}}^{\frac{u_\gamma^o + u_{\gamma+1}^o}{2}} f_U(u) du \qquad \forall \gamma \in \mathbf{X}_{\mathbb{Z}} \quad (10) \\
&= Pr[f(u, u^o) = \gamma] \qquad \forall \gamma \in \mathbf{X}_{\mathbb{Z}},
\end{aligned}
$$

where the p_γ^o's form the prescribed target feature map probability distribution given in Eq. (7), $u_{w(u)}^o \equiv u_{f(u,u^o)}$, $u_0^o \equiv 2u_{min} - u_1^o$, and $u_{N+1}^o \equiv 2u_{max} - u_N^o$.

Remark II-A.1: For a given codebook vector, the quantizer achieves minimum variance distortion [6] by the *nearest neighbor* winning rule in Eq. (5). In general, the centroid condition [6] is not satisfied by our optimal codebook vector because of the constraint on the feature map probability distribution in Eq. (10). By requiring the codebook vector to satisfy Eq. (9), we guarantee that the feature map $f(u, u^o)$ in Eq. (5) is topology preserving [4]. Requiring the codebook vector to simultaneously satisfy Eqs. (9) and (10) with $p_\gamma^o = 1/N$ for all $\gamma \in \mathbf{X}_{\mathbb{Z}}$ guarantees that $f(u, u^o)$ is a topology preserving and equiprobable feature map.

B. Self-organizing neural networks that converge to optimal quantizers

Consider the problem of obtaining an optimal quantizer, as described in section II-A, except that now we assume that the pdf $f_U : \mathbf{U}_\mathbb{R} \to \mathbb{R}_+$ of the random input sequence $u : \mathbb{Z}_+ \to \mathbf{U}_\mathbb{R}$ is *unknown* (including the support $\mathbf{U}_\mathbb{R}$ of $f_U(u)$). Given a prescribed *target feature map probability distribution* p^o in Eq. (7), we want to determine the optimal quantizer codebook vector u^o described by Eqs. (9)-(10) in an iterative manner, by sampling the random sequence $u(t)$.

Define a neural network whose output weight vector is given by

$$\hat{u}(t) = [\hat{u}_1(t) \cdots \hat{u}_N(t)]^T \in \mathbb{R}^N, \tag{11}$$

where $\hat{u}_\gamma : \mathbb{Z}_+ \to \mathbb{R}$ is the weight associated with the coordinate $\gamma \in \mathbf{X}_\mathbb{Z}$. $\hat{u}(t)$ can be thought of as the estimate at time t of the optimal codebook vector u^o in Eq. (9). The neural network feature map $f(\cdot, \cdot) : \mathbf{U}_\mathbb{R} \times \mathbb{R}^N \to \mathbf{X}_\mathbb{Z}$ is given by Eq. (4), only that the codebook vector $\bar{u}$ in Eq. (3) is replaced by the time varying vector weight $\hat{u}(t)$. We also define the neural network winning coordinate

$$w(t) = w(u(t)) = \arg\left\{ \min_{\gamma \in \mathbf{X}_\mathbb{Z}} |u(t) - \hat{u}_\gamma(t)| \right\}.$$

Notice that $w(t)$ is the coordinate of the output level that most closely approximates the current sample $u(t)$.

$w(t)$ is a random sequence with a non-stationary discrete feature map probability distribution that is given by

$$p_\gamma = \int_{\mathbf{U}_\mathbb{R}} \delta_{\gamma, w(u)} f_U(u) du, \qquad \forall \gamma \in \mathbf{X}_\mathbb{Z} \tag{12}$$

where

$$\delta_{\gamma, w(u)} = \begin{cases} 1 & \text{if} \quad \gamma = w(u) \\ 0 & \text{if} \quad \gamma \neq w(u) \end{cases} \tag{13}$$

and the codebook vector $\hat{u}(t)$ is kept constant in the expectation in Eq. (12).

In the remaining part of this paper we will describe an adaptive learning law for the neural network weight vector $\hat{u}(t)$ that achieves $\lim_{t \to \infty} \hat{u}(t) \to u^o$, where u^o is the optimal quantizer codebook vector satisfying Eqs. (9)-(10).

III. INTEGRALLY DISTRIBUTED NEURAL NETWORKS

In this section we describe a learning law for estimating the optimal codebook vector u^o in Eq. (9) indirectly, using an integrally distributed neural network. This network also generates a function $\hat{u}(\cdot, \cdot) : \mathbf{X}_\mathbb{R} \times \mathbb{Z}_+ \to \mathbb{R}$, which is smooth in x and converges to a homeomorphism that satisfies $\lim_{t \to \infty} \hat{u}(\gamma, t) = u_\gamma^o$ for all $\gamma \in \mathbf{X}_\mathbb{Z}$.

Assume that the homeomorphism $u(x)$ in Eq. (8) can be obtained by solving an integral equation of the first kind with a known smooth scalar symmetric Hilbert-Schmidt kernel $K(\cdot, \cdot) : \mathbf{X}_\mathbb{R} \times \mathbf{X}_\mathbb{R}^e \to \mathbb{R}_+$ that has a finite support, and an unknown influence function $c(\cdot) : \mathbf{X}_\mathbb{R} \to \mathbb{R}$ that satisfies

$$u(x) = \int_{\mathbf{X}_\mathbb{R}^e} K(x, \lambda) c(\lambda) d\lambda, \tag{14}$$

where $x \in \mathbf{X}_\mathbb{R}$ and $\lambda \in \mathbf{X}_\mathbb{R}^e$, with $\mathbf{X}_\mathbb{R} \subset \mathbf{X}_\mathbb{R}^e$, so that Eq. (14) is well defined at the boundaries of $\mathbf{X}_\mathbb{R}$. In all of the simulations shown in this paper we use a truncated Gaussian with a finite support

$$K(x, \lambda) = \begin{cases} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(x - \lambda)^2} & \text{if} \quad |x - \lambda| \leq \overline{\sigma} \\ 0 & \text{if} \quad |x - \lambda| > \overline{\sigma} \end{cases}, \tag{15}$$

where $\overline{\sigma} > 3\sigma > 0$.

Furthermore, we assume that $u(x)$ can be obtained using a kernel that is contained by a finite number of basis

$$u(x) = \sum_{\nu \in \mathbf{X}_\mathbb{Z}^e} K(x, \nu) c_\nu^o, \tag{16}$$

where

$$\mathbf{X}_\mathbb{Z}^e = \{-(m-1), -(m-2), \cdots, N+m\} \tag{17}$$

for some integer $0 < m << N$. The elements of the vector

$$c^o = [c_{-(m-1)}^o, \cdots, c_{N+m}^o]^T \in \mathbb{R}^{N+2m}$$

are *unknown* optimal influence coefficients that satisfy $u_\gamma^o = u(\gamma)$ for all $\gamma \in \mathbf{X}_\mathbb{Z}$, and are estimated using an integrally distributed neural network, which will be subsequently described.

The purpose of the distributed neural network is to estimate the optimal influence coefficients c_ν^o in Eq. (16), through a self-organizing learning process.

We now define the integrally distributed neural network, which consists of $N + 2m$ neurons. Neurons in the network are indexed using the coefficient set $\mathbf{X}_\mathbb{Z}^e$ in Eq. (17). Neurons $\nu \in \mathbf{X}_\mathbb{Z}^e \setminus \mathbf{X}_\mathbb{Z}$ are edge neurons. Each neuron $\nu \in \mathbf{X}_\mathbb{Z}^e$ has an associate influence coefficient estimate $\hat{c}_\nu(t) \in \mathbb{R}$, which is an estimate of the optimal influence coefficient estimate c_ν^o in Eq. (16).

The outputs of the network are:
1) The function $\hat{u}(\cdot, \cdot) : \mathbf{X}_\mathbb{R} \times \mathbb{Z}_+ \to \mathbb{R}$ generated by

$$\hat{u}(x, t) = \sum_{\nu \in \mathbf{X}_\mathbb{Z}^e} K(x, \nu) \hat{c}_\nu(t). \tag{18}$$

2) The output weight vector $\hat{u}(t)$ defined in Eq. (11), whose elements $\hat{u}_\gamma(t)$'s are generated by:

$$\hat{u}_\gamma(t) = \hat{u}(\gamma, t), \qquad \forall \gamma \in \mathbf{X}_\mathbb{Z}. \tag{19}$$

3) The winning neuron coordinate [1]

$$w(t) = \arg\{ \min_{\gamma \in \mathbf{X}_\mathbb{Z}} |u(t) - \hat{u}_\gamma(t)| \}. \tag{20}$$

In addition, the network recursively computes, for each non-edge neuron $\gamma \in \mathbf{X}_\mathbb{Z}$, an estimate of the probability p_γ in Eq. (12) of that neuron becoming a winning neuron. The set of these probability estimates, $\{\hat{p}_\gamma(t) | \gamma \in \mathbf{X}_\mathbb{Z}\}$, will be referred to as the network's *feature map probability distribution estimate*. Notice that edge neurons cannot become

---

[1]We will overload the notation $w(\cdot)$ by simultaneously defining: $w(t) : \mathbb{Z}_+ \to \mathbf{X}_\mathbb{Z}$, $w(u) : \mathbf{U}_\mathbb{R} \to \mathbf{X}_\mathbb{Z}$ and $w(x) : \mathbf{X}_\mathbb{R} \to \mathbf{X}_\mathbb{Z}$, noticing that $w(t) = w(u(t)) = w(u(x(t)))$. The functions will be distinguished from each other by the arguments used.

winning neurons and hence only have influence coefficient estimates.

The feature map probability distribution estimate is computed as follows:

$$\hat{p}_\gamma(t+1) = \hat{p}_\gamma(t) + \alpha_p(t)(-\hat{p}_\gamma(t) + \delta_{\gamma,w(t)}) \quad \forall \gamma \in \mathbf{X}_\mathbb{Z},$$

$$\hat{p}_\gamma(0) \geq 0 \quad \forall \gamma \in \mathbf{X}_\mathbb{Z}, \quad \sum_{\gamma=1}^{N} \hat{p}_\gamma(0) = 1, \qquad (21)$$

where $w(t)$ is the winning neuron coordinate given by Eq. (20), $\delta_{\gamma,w(t)}$ is a Kronecker delta, as defined in Eq. (13), and $\alpha_p(t)$ is the standard stochastic approximation adaptive gain, which has the following properties

$$\alpha_p(t) > 0, \quad \sum_{t=1}^{\infty} \alpha_p(t) = \infty, \quad \sum_{t=1}^{\infty} \alpha_p^2(t) < \infty. \quad (22)$$

It is easy to show that, under the initial condition in Eq. (21), the update law for $\hat{p}_\gamma$ guarantees that, for all $t > 0$

$$\hat{p}_\gamma(t) \geq 0 \quad \forall \gamma \in \mathbf{X}_\mathbb{Z}, \quad \sum_{\gamma=1}^{N} \hat{p}_\gamma(t) = 1,$$

Moreover, it can also be shown that, under the assumption that p_γ in Eq. (12) becomes stationary, $\lim_{t\to\infty} \hat{p}_\gamma(t) = p_\gamma$. Notice that p_γ becomes stationary if the influence coefficient estimates $\hat{c}_\gamma$'s become stationary.

To make our equations more compact, we define the neural network influence coefficient estimate vector and feature map probability estimate vector by

$$\begin{aligned} \hat{c}(t) &= [\hat{c}_{-(m-1)}(t), \ldots, \hat{c}_{N+m}(t)]^T \in \mathbb{R}^{N+2m} \quad (23) \\ \hat{p}(t) &= [\hat{p}_1(t), \ldots, \hat{p}_N(t)]^T \in \mathbb{R}^N. \quad (24) \end{aligned}$$

Notice that Eq. (19) can now be written as

$$\hat{u}(t) = K\,\hat{c}(t),$$

where $K \in \mathbb{R}^N \times \mathbb{R}^{N+2m}$ is the kernel matrix, with element $K_{i,j} = K(i, j-m)$, which must be rank N. This integrally distributed formulation of $\hat{u}$ was initially proposed in [13], in the context of repetitive and learning control algorithms for robot manipulators.

For the function $\hat{u}(x,t)$ in Eq. (18) to converge to an orientation preserving homeomorphism, it is necessary that

$$\lim_{t\to\infty} \hat{u}'(x,t) = \lim_{t\to\infty} \frac{\partial}{\partial x}\hat{u}(x,t) > 0$$

(Negative for orientation reversing).

We now define the vector of partial derivatives of the function $\hat{u}(x,t)$ evaluated at the coordinates $\gamma \in \mathbf{X}_\mathbb{Z}$ by

$$\hat{u}'(t) = \begin{bmatrix} \hat{u}'(1,t) & \cdots & \hat{u}'(N,t) \end{bmatrix}^T = K'\hat{c}(t), \quad (25)$$

where $K' \in \mathbb{R}^N \times \mathbb{R}^{N+2m}$ is the matrix of partial derivatives of the kernel function, whose (i,j) element is given by $K'_{i,j} = \partial K(x,\lambda)/\partial x|_{x=i,\lambda=j-m}$. We also define the column vectors $k_\gamma \in \mathbb{R}^{N+2m}$ and $k'_\gamma \in \mathbb{R}^{N+2m}$ which will respectively be the transpose of the γ-th rows of K and K':

$$K^T = [k_1, \cdots, k_N], \quad K'^T = [k'_1, \cdots, k'_N]. \quad (26)$$

Therefore, for $\gamma \in \mathbf{X}_\mathbb{Z}$,

$$\hat{u}_\gamma(t) = k_\gamma^T \hat{c}(t) \quad \text{and} \quad \left.\frac{\partial \hat{u}(x,t)}{\partial x}\right|_{x=\gamma} = \hat{u}'_\gamma(t) = k'^T_\gamma \hat{c}(t).$$

A. Influence coefficient learning algorithm

The influence coefficient estimate $\hat{c}(t)$ defined in Eq. (23) is updated recursively at each sampling time t, after the winning neuron coordinate $w(t)$ in Eq. (20) is determined.

The update law has two terms:

$$\hat{c}(t+1) = \hat{c}(t) + \alpha(t)[-\delta\bar{c}_1(t) - \delta\hat{c}_2(t)], \quad (27)$$

where $\alpha(t) > 0$ is another stochastic approximation gain. Let β_3 be the ratio between $\alpha_p(t)$ and $\alpha(t)$:

$$\beta_3 = \alpha_p(t)/\alpha(t). \quad (28)$$

The two terms $\delta\bar{c}_1(t)$ and $\delta\hat{c}_1(t)$ in Eq (27) will now be defined.

Feature map probability distribution tracking law: The structure of this term depends on whether the weight of winning neuron, $\hat{u}_{w(t)}(t)$, is an extremum value of $\widehat{\mathbf{U}}_\mathbb{Z}(t)$ or not. Define the two neuron coordinates with the extremum values of the set $\widehat{\mathbf{U}}_\mathbb{Z}(t)$ respectively by

$$\overline{\partial}x = \arg\left\{\max_{\gamma\in\mathbf{X}_\mathbb{Z}} \hat{u}_\gamma\right\} \quad \text{and} \quad \underline{\partial}x = \arg\left\{\min_{\gamma\in\mathbf{X}_\mathbb{Z}} \hat{u}_\gamma\right\}.$$

- If $w(t)$ is not an extremum neuron coordinate, i.e., $w(t) \in \mathbf{X}_\mathbb{Z}\backslash\{\overline{\partial}x, \underline{\partial}x\}$:

$$\delta\bar{c}_1(t) = \beta_1 \frac{\mathcal{Q}_{w(t)}}{p^o_{w(t)}} k'_{w(t)} \operatorname{sign}(\hat{u}_{w(t)}), \quad (29)$$

where $\beta_1 > 0$, $k'_{w(t)}$ is the transpose of the winning neuron coordinate row of the matrix of kernel partial derivatives, as defined in Eq. (26), and $\operatorname{sign}(\cdot)$ and $\mathcal{Q}_{w(t)}$ are defined by

$$\operatorname{sign}(y) = \begin{cases} 1 & \text{if } y > 0 \\ 0 & \text{if } y = 0 \\ -1 & \text{if } y < 0 \end{cases},$$

$$\mathcal{Q}_{w(t)} = 1 + \phi(\hat{p}_{w(t)} - p^o_{w(t)}), \quad (30)$$

where $\phi > 0$. $p^o_{w(t)} > 0$ and $\hat{p}_{w(t)} \geq 0$ are the target and estimated values of the feature map probability at the current winning neuron coordinate, respectively given by Eqs. (7) and (21).

- If $w(t)$ is an extremum neuron coordinate, i.e., $w(t) \in \{\overline{\partial}x, \underline{\partial}x\}$, then

$$\delta\bar{c}_1(t) = \beta_1 \frac{\mathcal{Q}_{w(t)}}{p^o_{w(t)}} \delta\hat{c}_1^X(t), \quad (31)$$

$$\delta\hat{c}_1^X(t) = \left[\frac{k_{w(t)} + k_{C(w(t))}}{2}\right] \operatorname{sign}\left[\hat{u}_{C(w(t))} - \hat{u}_{w(t)}\right],$$

where $C(w(t))$ is the neuron coordinate with the closest value to $\hat{u}_{w(t)}(t)$, i.e.,

$$C(w(t)) = \arg \left\{ \min_{\gamma \in \mathbf{X}_{\mathbb{Z}} \backslash w(t)} |\hat{u}_{w(t)}(t) - \hat{u}_\gamma(t)| \right\}.$$

Orientation preserving update law:

$$\delta \hat{c}_2(t) = \beta_2 \, k'_{w(t)} \left(|\hat{u}'_{w(t)}| - \hat{u}'_{w(t)} \right) \frac{[\text{sign}(\hat{u}'_{w(t)}(t)) - 1]}{p^o_{w(t)}},$$

$$(32)$$

where $\beta_2 > 0$, $\hat{u}'_{w(t)} = \hat{u}'(w(t), t)$ is defined in Eq (25), and $p^o_{w(t)} > 0$ is the target feature map probability at the current winning neuron coordinate, as defined in Eq. (7). A slight modification of the learning law $\delta \hat{c}_2(t)$ in Eq. (32) will result in an orientation reversing update law.

To analyze the convergence properties of the learning algorithm given by Eqs. (21)-(32), we use Ljung's Ordinary Differential Equation (ODE) approach [14]. We only consider the convergence analysis for unknown uniform input pdfs due to page limitations. For the case of nonuniform input pdfs, their global Lipschitz continuity condition is exploited in the convergence analysis. We now introduce our main result.

Theorem III-A.1: Given a compatible set of prescribed feature map target probability distribution defined by Eq. (7), and a measurable input stationary random sequence $u(t) \in \mathbf{U}_{\mathbb{R}} \subset \mathbb{R}$ with an unknown uniform pdf $f_U(u)$, the output weight vector $\hat{u}(t)$ of an integrally distributed neural network converges to an optimal codebook vector u^o satisfying Eqs. (9)-(10).

Sketch of Proof: Let's define the lower bounded functional

$$V(\hat{c}, \hat{p}) = V_1(\hat{c}) + V_2(\hat{c}) + V^{obv}(\hat{p})$$

$$V_1(\hat{c}) = \beta_1 \int_{\mathbf{X}_{\mathbb{R}}} \frac{\frac{\phi}{2}(p_{w(x)} - p^o_{w(x)})^2}{p^o_{w(x)} f_U(u^o_{w(x)})} f_X(x) dx$$

$$+ \beta_1 \int_{\mathbf{X}_{\mathbb{R}}} \frac{p_{w(x)}}{p^o_{w(x)} f_U(u^o_{w(x)})} f_X(x) dx$$

$$V_2(\hat{c}) = \frac{\beta_2}{2} \int_{\mathbf{X}_{\mathbb{R}}} \frac{\left(|\hat{u}'_{w(x)}| - \hat{u}'_{w(x)} \right)^2}{p^o_{w(x)}} f_X(x) dx$$

$$V^{obv}(\hat{p}) = \frac{\beta_1}{2} \int_{\mathbf{X}_{\mathbb{R}}} \frac{\phi \tilde{p}^2_{w(x)}}{p^o_{w(x)} f_U(u^o_{w(x)})} f_X(x) dx,$$

where the winning coordinate $w(x)$ is given by Eq. (20). p^o_γ is the predefined feature map target probability distribution, as defined in Eq. (7). $\beta_1 > 0$, $\beta_2 > 0$ and $\phi > 0$ are the relative weighting gains. $\tilde{p}_w$ is the wth entry of the probability estimation error vector defined by the formula $\tilde{p} = p - \hat{p}$.

Remark III-A.2: $V_1(\hat{c})$ achieves its minimum if and only if $p = p^o$ for the case of uniform input pdfs. $V_2(\hat{c})$ will be zero if and only if the function $\hat{u}(x)$ in Eq. (18) is orientation preserving. $V^{obv}(\hat{p})$ will be zero if and only if $\hat{p} = p$.

Applying the ODE approach to Eqs. (20), (21), (27)-(32) we obtain

$$\dot{\hat{c}}(\tau) = -\Delta \bar{c}_1(\tau) - \Delta \hat{c}_2(\tau)$$

$$= -\Delta \hat{c}_1(\tau) - \Delta \hat{c}_2(\tau) + \Delta \tilde{c}_1(\tau)$$

$$\Delta \bar{c}_1(\tau) = \int_{\mathbf{X}_{\mathbb{R}}} \delta \bar{c}_1(x, \tau) f_X(x) dx \qquad (34)$$

$$\Delta \hat{c}_2(\tau) = \int_{\mathbf{X}_{\mathbb{R}}} \delta \hat{c}_2(x, \tau) f_X(x) dx,$$

where the influence coefficient $\hat{c}(\tau)$ and $\hat{p}(\tau)$ are kept constant in the expectation. $\Delta \tilde{c}_1(\tau) \equiv \Delta \hat{c}_1(\tau) - \Delta \bar{c}_1(\tau)$, while $\Delta \hat{c}_1(\tau)$ is defined by

$$\Delta \hat{c}_1(\tau) = \int_{\mathbf{X}_{\mathbb{R}}} \delta \hat{c}_1(x, \tau) f_X(x) dx,$$

where $\delta \hat{c}_1(x, \tau)$ is defined by the same formula as $\delta \bar{c}_1(x, \tau)$, which is defined in Eqs.(29)-(31), except for having $\hat{p}_{w(t)}$ replaced by $p_{w(t)}$. Differentiating each of the terms in Eq. (33) with respect to time τ, and noticing that $\partial w(x)/\partial \hat{c} = 0$ by Lemma A.1 in [12], we can show that

$$\dot{V}_1(\hat{c}) = \left(\frac{\partial V_1(\hat{c})}{\partial \hat{c}} \right)^T \dot{\hat{c}}(\tau) = \Delta \hat{c}_1^T(\tau) \dot{\hat{c}}(\tau)$$

$$\dot{V}_2(\hat{c}) = \left(\frac{\partial V_2(\hat{c})}{\partial \hat{c}} \right)^T \dot{\hat{c}}(\tau) = \Delta \hat{c}_2^T(\tau) \dot{\hat{c}}(\tau) \qquad (35)$$

$$\dot{V}^{obv}(\hat{p}) = -\Delta \tilde{c}_1^T(\tau)(\Delta \hat{c}_1(\tau) + \Delta \hat{c}_2(\tau)) - \tilde{p}^T(\tau) M \tilde{p}(\tau).$$

The lengthy derivation of Eq. (35) will be omitted in this paper due to the page limitations. From Eq. (34) and (35),

$$\dot{V}(\hat{c}, \hat{p}) = -|\Delta \hat{c}_1(\tau) + \Delta \hat{c}_2(\tau)|^2 - \tilde{p}^T(\tau) M \tilde{p}(\tau) \leq 0, \quad (36)$$

where $M \in \mathbb{R}^{N \times N} \succ 0$ for some β_3 and ϕ. Integrating Eq. (36) with respect to time, for all $T \geq 0$,

$$V(\hat{c}, \hat{p})(T) - V(\hat{c}, \hat{p})(0) = \int_0^T \dot{V}(\hat{c}, \hat{p})(\tau) d\tau \leq 0. \quad (37)$$

This implies that $V(\hat{c}, \hat{p})(T) \leq V(\hat{c}, \hat{p})(0)$, and $V_1(\hat{c})$ and $V_2(\hat{c})$ are bounded. Eq. (37) implies that $V(\hat{c}, \hat{p}) \in \mathcal{L}_\infty$, $\tilde{p}(\tau) \in \mathcal{L}_2$ and $(\Delta \hat{c}_1(\tau) + \Delta \hat{c}_2(\tau)) \in \mathcal{L}_2$. Notice that $\tilde{p}(\tau) \in \mathcal{L}_\infty$ and $\Delta \hat{c}_1 \in \mathcal{L}_\infty$. Utilizing Schwartz's inequality we obtain from Eq. (33) and Eq. (34)

$$|\Delta \hat{c}_2(\tau)| \leq 2\sqrt{2\beta_2} \left\{ \int_{\mathbf{X}_{\mathbb{R}}} k'^T_{w(x)} k'_{w(x)} dx \right\}^{1/2} V_2^{1/2}(\hat{c})(\tau).$$

$$(38)$$

Thus, $\Delta \hat{c}_2 \in \mathcal{L}_\infty$ since $V_2(\hat{c}) \in \mathcal{L}_\infty$, and

$$\dot{\hat{c}} = -(\Delta \hat{c}_1(\tau) + \Delta \hat{c}_2(\tau)) + \Delta \tilde{c}_1(\tau) \in \mathcal{L}_2 \cap \mathcal{L}_\infty. \quad (39)$$

From Eq. (35), we obtain

$$\frac{\partial \Delta \hat{c}_1}{\partial \tau} = \left(\frac{\partial^2 V_1(\hat{c})}{\partial \hat{c}^2} \right)^T \dot{\hat{c}}(\tau), \quad \frac{\partial \Delta \hat{c}_2}{\partial \tau} = \left(\frac{\partial^2 V_2(\hat{c})}{\partial \hat{c}^2} \right)^T \dot{\hat{c}}(\tau).$$

$$(40)$$

Since $\dot{\hat{c}}$, $\left(\frac{\partial^2 V_1(\hat{c})}{\partial \hat{c}^2} \right)$ and $\left(\frac{\partial^2 V_2(\hat{c})}{\partial \hat{c}^2} \right)$ can be shown to be bounded, by differentiating $\dot{\hat{c}}(\tau)$ in Eq. (34) with respect

to time, we can conclude that $\ddot{\hat{c}}(\tau) \in \mathcal{L}_\infty$ and $\ddot{V}(\tau) \in \mathcal{L}_\infty$. Thus, $\dot{V}(\tau)$ is uniformly continuous in time τ and by Barbalat's lemma [15], $\lim_{\tau \to \infty} \dot{V}(\tau) = 0$, which implies that $\lim_{\tau \to \infty} \tilde{p}(\tau) = \lim_{\tau \to \infty} (p - \hat{p}(\tau)) = 0$, and $\lim_{\tau \to \infty} -|\Delta\hat{c}_1(\tau) + \Delta\hat{c}_2(\tau)|^2 = 0$. By the structure of $\delta\hat{c}_1(\tau)$ and $\delta\hat{c}_2(\tau)$ in Eqs.(29)-(31) and Eq. (32) respectively, $|\Delta\hat{c}_1(\tau) + \Delta\hat{c}_2(\tau)| = 0$ guarantees that $\Delta\hat{c}_1(\tau) = 0$ and $\Delta\hat{c}_2(\tau) = 0$ simultaneously. For some ϕ, it can be shown that $\Delta\hat{c}_1(\tau) = 0$ along with $\Delta\hat{c}_2(\tau) = 0$ implies that the discrete feature map probability p_γ in Eq. (12) is equal to the target feature map probability p_γ^o in Eq. (7) for all $\gamma \in \mathbf{X}_\mathbb{Z}$, i.e., $p = p^o$. $\Delta\hat{c}_2(\tau) = 0$ along with $\Delta\hat{c}_1(\tau) = 0$ implies that $\hat{u}_\gamma$ is monotonically non-decreasing and $\hat{u}(x)$ is orientation preserving. A simple modification of $V_2(\hat{c})$ would provide an orientation reversing map control. The map $\hat{u}(x)$ having a co-domain restricted to the range space of $\hat{u}(x)$, $\hat{u}(x) : \mathbf{X}_\mathbb{R} \to \mathcal{R}(\hat{u}(\mathbf{X}_\mathbb{R}))$ is continuous (and smooth) by construction in Eq. (18) and it is orientation preserving when $V_2(\hat{c}) = 0$. Then there exists a $\hat{u}^{-1}(x)$ that is also continuous, which proves that $\hat{u}(x)$ is a homeomorphism. Thus, Theorem III-A.1 follows. $\diamondsuit$.

IV. SIMULATION RESULTS

In this section, simulation results for the two different nonuniform input pdfs, and two different (uniform and nonuniform) target probability distributions, are presented. We compare the performance of the integrally distributed learning algorithm presented in this paper against the well-known Kohonen self-organizing learning algorithm [4] to achieve an equiprobable target probability p^o. In the simulations, the integrally distributed neural network had a total of $N = 38$ output weights and $N + 2m = 42$ neurons (i.e., 4 neurons were edged neurons, as defined in Eq. (17)). A value of $\sigma = 1.5$ was used in the kernel given by Eq. (15). The Kohonen neural network had a total of 38 neurons and output weights. A kernel given by Eq. (15) with $\sigma = 2$ was used in this learning law to update the output weights of neighboring neurons to the winning neuron [4]. The initial conditions of the output weights $\hat{u}_\gamma(0)$'s in both algorithms were set to the same values. For the integrally distributed learning algorithm, the Kullback-Leibler (KL) measure of cross-entropy [16] was used to evaluate the convergence of the feature map probability estimate $\hat{p}$ to a given target probability p^o. The KL measure between $\hat{p}$ and p^o is given by the formula

$$D(\hat{p}, p^o) = \sum_{\gamma=1}^{N} \hat{p}_\gamma \ln \frac{\hat{p}_\gamma}{p_\gamma^o}, \qquad (41)$$

where $D(\hat{p}, p^o) \geq 0$, and $D(\hat{p}, p^o)$ vanishes if and only if $\hat{p} = p^o$.

A. Gaussian input distribution

Fig. 1 shows the output weights $\hat{u}_\gamma$'s for the two algorithms after 20,000 iterations, for a Gaussian input probability density function $u \sim \mathcal{N}[0, 3^2]$ and an equiprobability
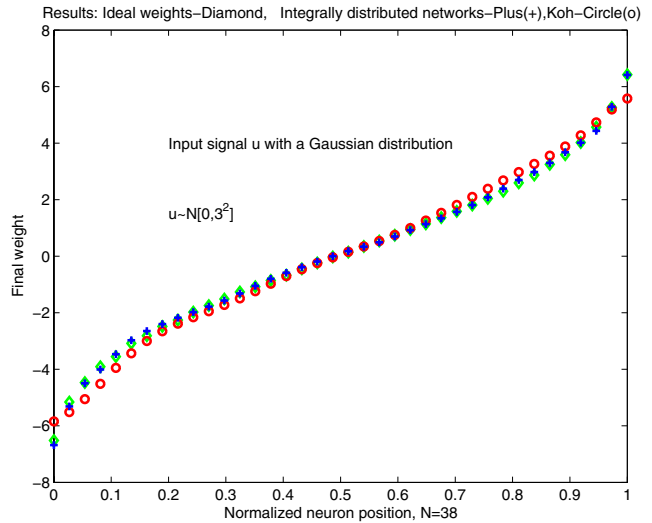

Fig. 1. Results for a random input signal with a Gaussian distribution $u \sim \mathcal{N}(0, \sigma^2)$, where $\sigma = 3$ (20,000 iterations) for the uniform target probability p^o. Analytically calculated ideal output weights (diamonds[$\diamondsuit$]), the Kohonen's law (circles[$\circ$]), and the integrally distributed law (pluses[+]).

target feature map distribution $p^o = 1/N$. It is clear from Fig. 1 that the integrally distributed network converges to the analytically calculated ideal output weights, while those in Kohonen's algorithm converge to an affine approximation. Fig. 2(a) depicts the estimated feature map probability $\hat{p}_\gamma$ of the algorithm proposed, as computed in Eq. (21), which shows $p \approx \hat{p}$ converges to the equiprobability of p^o. Fig. 2(b) shows the Root Mean Square (RMS) error for the neuron output weights by comparing the analytically calculated ideal output weights with the output weights $\hat{u}_\gamma$'s produced by each algorithm. This plot shows that the proposed algorithm has better steady-state performance than Kohonen's law. Fig. 2(b) also shows that the proposed algorithm has a faster convergence rate than the Kohonen's law. The final RMS error value for the Gaussian input pdf after 20,000 iterations are shown in Table. I. Fig. 2(c) shows the Kullback-Leibler's measure shown in Eq. (41) between $\hat{p}$ and p^o v.s. iteration time, showing that $\hat{p} \to p^o$ as $t \to \infty$. The output weights $\hat{u}_\gamma$ and the estimated homeomorphic map $\hat{u}(x)$ with the winner determination borders are illustrated in Fig. 3 for the same Gaussian input along with the equiprobable target feature map distribution.

B. Nonuniform target feature map probability p^o

We now choose a truncated Gaussian-like nonuniform target feature map probability distribution p^o, as depicted in Fig. 5(a), for the same Gaussian input distribution $\mathcal{N}[0, 3^2]$. Fig. 4 shows the converged output weights $\hat{u}_\gamma$'s after 20,000 iterations. Fig. 5 depicts (a) the values of $\hat{p}$ as compared to p^o after 20,000 iterations, (b) the RMS error between the ideal output weights and the converged output weights from the integrally distributed law, and (c) the convergence rate of $\hat{p}$ to p^o, as illustrated by the KL measure. As shown in Fig.

Estimated $p_{w(x)}$, Integrally disrtributed Networks

(a)

$P_{w(x)}$

Neuron number

(b)

Rms error

Integrally disrtributed Networks
Kohonen

time (iteration)

x 10⁴

(c)

KL measure

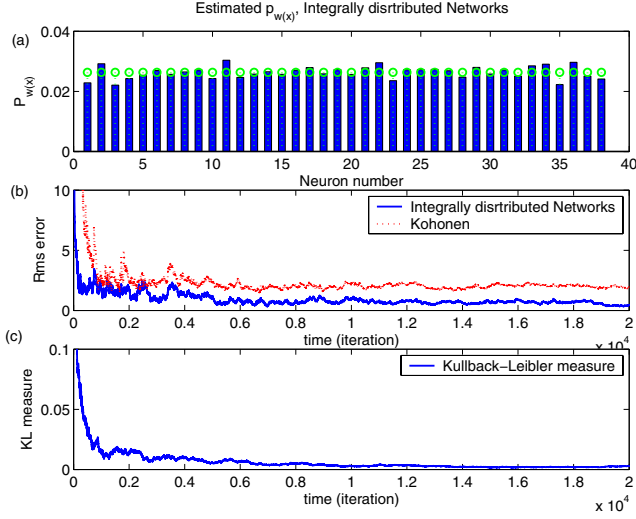Kullback–Leibler measure

time (iteration)

x 10⁴

Fig. 2. A Gaussian input distribution: (a) The discrete feature map probability estimate $\{\hat{p}_\gamma\}$ (solid bars), and the uniform target feature map probability distribution $\{p_\gamma^o\}$ (balls). (b) The RMS error between u_γ^o's and $\hat{u}_\gamma$'s of the integrally distributed law (solid line) and the Kohonen's law (dotted line). (c) The Kullback-Leibler's measure of cross-entropy between $\hat{p}$ and p^o v.s. iteration time.
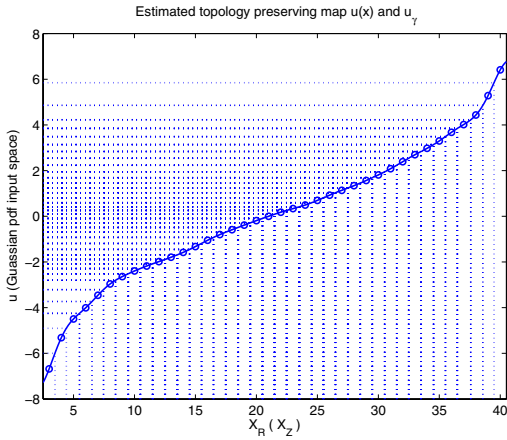

Estimated topology preserving map $u(x)$ and u_γ

u (Gaussian pdf input space)

$X_\mathbb{R} (X_\mathbb{Z})$

Fig. 3. A Gaussian input distribution: the homeomorphic map $\hat{u}(x)$ (solid line), the output weights $\hat{u}_\gamma$ (circles) and winner determination boarders (dotted lines) in $\mathbf{U}_\mathbb{R}$ and $\mathbf{X}_\mathbb{R}$ after 20,000 iterations.

5, the feature map probability estimate $\hat{p}$ converges to the nonuniform target probability distribution p^o successfully. The final RMS error value from Fig. 5(b) for the new algorithm is shown in Table I.

C. Cubic input distribution

In order to evaluate the performance of the proposed algorithm along with the Kohonen's law under a harsh input pdf condition, we consider a cubic distribution $u = 20v^3$, where v is uniformly distributed $v \sim U[0, 1]$, as shown in Fig. 6. Fig. 6 shows that this input pdf has a large Lipschitz constant. Fig. 7 demonstrates that the proposed algorithm's output weights $\hat{u}_\gamma$'s converge to the ideal output weights distribution, while the output weights $\hat{u}_\gamma$'s produced by the

Results: Ideal weights–Diamond, Integrally distributed networks–Plus+

Final weight

Input signal u with a Gaussian distribution

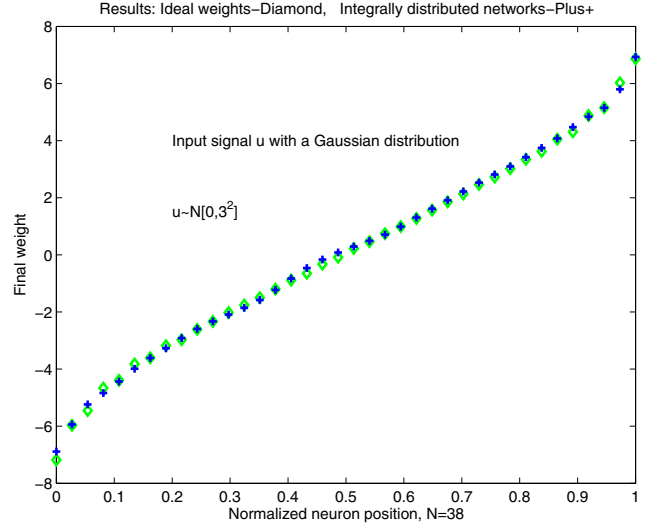$u \sim N[0, 3^2]$

Normalized neuron position, N=38

Fig. 4. Results for a random input signal with a Gaussian input distribution for the Gaussian-like nonuniform target probability p^o (20,000 iterations). Analytically calculated ideal output weights (diamonds[$\diamond$]). The output weights of the integrally distributed law (pluses[+]).

TABLE I
ROOT MEAN SQUARE (RMS) ERROR OF THE OUTPUT WEIGHTS.

Input distribution: Target probability p^o: Number of iterations:	Gaussian uniform 20,000	Gaussian nonuniform 20,000	Cubic uniform 20,000
Integrally distributed law	0.4630	0.6792	0.5630
Kohonen's algorithm	1.8990	n/a	8.9086

Kohonen's law fail to do so, after 20,000 iterations. Fig. 8 shows (a) the converged values of $\hat{p}$ in Eq. (21) as compared to p^o, (b) the evolution of the RMS error between u_γ^o and $\hat{u}_\gamma$ for both algorithms, and (c) the convergence rate of $\hat{p}$ to p^o for the proposed algorithm, as illustrated by the KL measure. The final RMS error values after 20,000 iterations for both algorithms for the cubic input pdf are shown in Table I.

V. CONCLUSIONS

A new one-dimensional integrally distributed neural network was presented. The adaptive network converges to a set that produces a predefined target probability p^o for unknown and possibly nonuniform random input signals under mild conditions. The network also produces the orientation preserving homeomorphism that maps a predefined pdf into the unknown pdf of the input signal. The convergence properties of the learning algorithm was analyzed by using the ODE approach [14]. Solid simulation results verified the convergence of our new algorithm.

REFERENCES

[1] O. Musse, F. Heitz, and J.-P. Armspach, "Topology preserving deformable image matching using constaned hierarchical parametric models," *IEEE Transactions on Image Processing*, vol. 10, no. 7, pp. 1081–1093, July 2001.

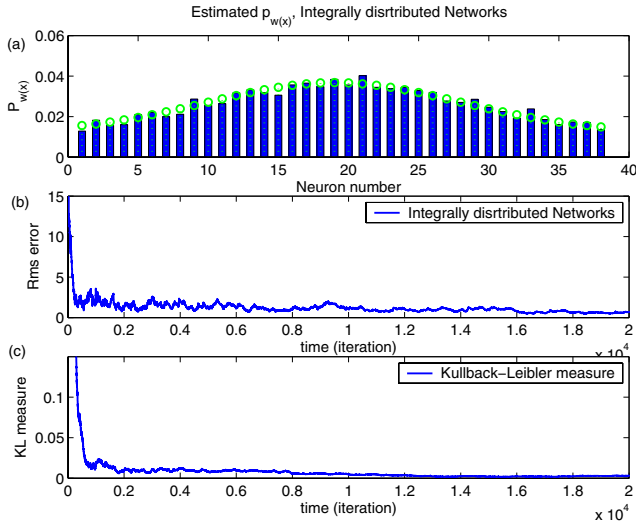[2] J. M. Lee, *Introduction to Smooth Manifolds*. Springer-Verlag, 2003.

Fig. 5. A Gaussian input distribution: (a) The discrete feature map probability estimate $\{\hat{p}_\gamma\}$ (solid bars), and the truncated Gaussian-like target feature map probability distribution $\{p_\gamma^o\}$ (balls) for the integrally distributed algorithm. (b) The RMS error between the ideal output weights and the output weights from the integrally distributed law. (c) The Kullback-Leibler's measure of cross-entropy between $\hat{p}$ and p^o.
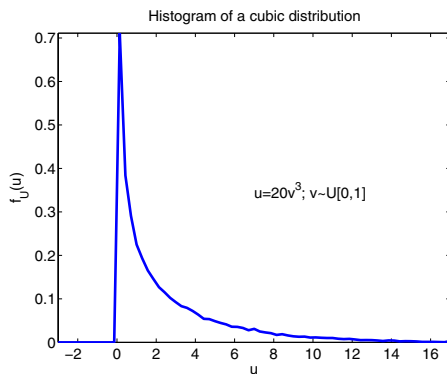

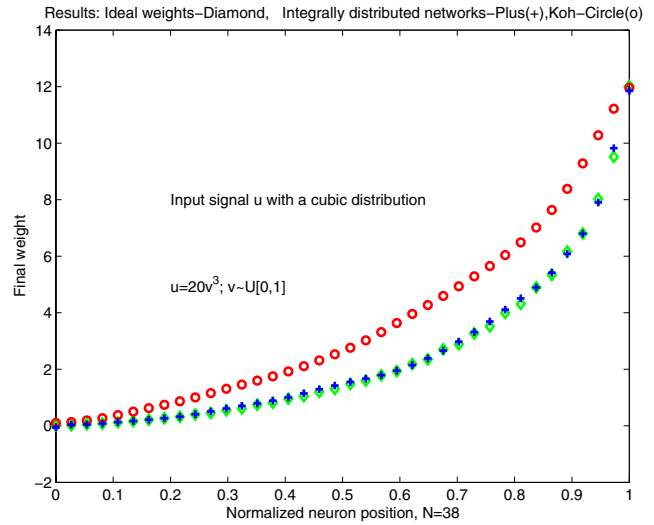Fig. 6. The histogram of a cubic distribution $(f_U(u))$.

Fig. 7. Results for a random input signal with a cubic distribution (20,000 iterations) for the uniform target probability p^o. Analytically calculated ideal output weights (diamonds[◊]), the Kohonen's law (circles[○]), and the integrally distributed law (pluses[+]).
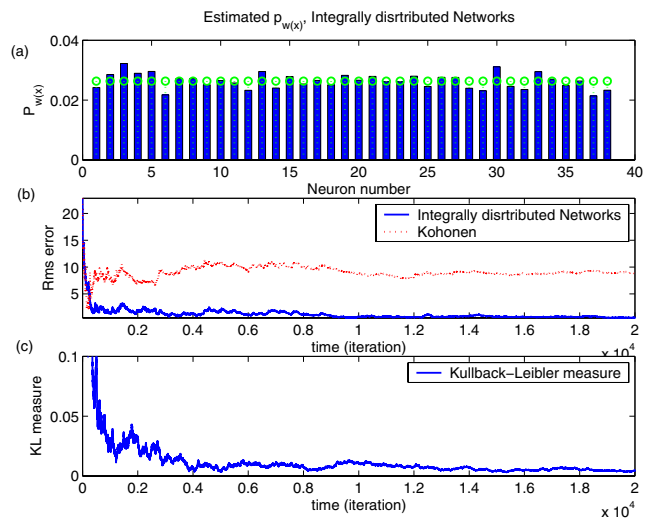

Fig. 8. A cubic input distribution: (a) The discrete feature map probability estimate $\{\hat{p}_\gamma\}$ (solid bars), and the uniform target feature map probability distribution $\{p_\gamma^o\}$ (balls) for the integrally distributed algorithm. (b) The RMS error of the integrally distributed law (solid line) and the Kohonen's law (dotted line). (c) The Kullback-Leibler's measure between $\hat{p}$ and p^o.

[3] D. Freedman, "Efficient simplicial reconstructions of manifolds from their samples," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 10, pp. 1349–1357, October 2002.

[4] T. Kohonen, *Self Organizing Maps*. Spinger-Verlag, 1995.

[5] T. Martinetz and K. Schulten, "Topology representing networks," *Neural Networks*, vol. 7, no. 3, pp. 507–522, 1994.

[6] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 2001.

[7] H. Ritter and K. Schulten, "Convergence properties of kohonen's topology conserving maps: Fluctuation, stability and dimension selection," *Biological Cybernetics*, vol. 60, pp. 59–71, 1988.

[8] Z.-P. Lo, Y. Yu, and B. Bavarian, "Analysis of the convergence properties of topology preserving neural networks," *IEEE Transaction on Neural Networks*, vol. 4, number =, 1993.

[9] C. Bouton and G. Pagès, "Self-organization and a.s. convergence of the one-dimensional kohonen algorithm with non-uniform distributed stimuli," *Stochastic Processes and their Applications*, vol. 47, pp. 249–274, 1993.

[10] ——, "Convergence and distribution of the one-dimensional kohonen algorithm when the stimuli are not uniform," *Advanced Applied Probability*, vol. 26, pp. 80–103, 1994.

[11] R. Horowitz and L. Alvarez, "Self-organizing neural networks: convergence properties," *IEEE International Conference on Neural Networks*, vol. 1, pp. 7–12, 1996.

[12] ——, "Convergence properties of self-organizing neural networks," *Proceeding of the American Control Conference*, pp. 1339–1344, June 1995.

[13] W. Messner, R. Horowitz, W. W. Kao, and M. Boals, "A new adaptive learning rule," *IEEE Transaction on Automatic Control*, vol. 36, no. 2, pp. 188–197, February 1991.

[14] L. Ljung, "Analysis of recursive stochastic algorithms," *IEEE Transactions on Automatic Control*, vol. 22, no. 4, pp. 551–575, 1977.

[15] H. K. Khalil, *Nonlinear systems*, 2nd ed. Prentice Hall, 1996.

[16] J. Kapur and H. K. Kesavan, *Entropy Optimization Principle with Applications*. Academic Press Inc., 1992.