A Gaussian Process-based Self-Organizing Incremental Neural Network

Xiaoyu Wang*, Giona Casiraghi†, Yan Zhang† and Jun-ichi Imura*
*Graduate School of Engineering, Tokyo Institute of Technology, Japan
Email: wang.x.al@m.titech.ac.jp, imura@sc.e.titech.ac.jp
†Chair of Systems Design, ETH Zürich, Switzerland
Email: gcasiraghi@ethz.ch, yanzhang0@ethz.ch

***Abstract*—This paper proposes a Gaussian process-based self-organizing incremental neural network (GPINN) to address the density estimation problem of online unsupervised learning. First, we adopt Gaussian process models with adaptive kernels that map the distribution of the neighbors of each node to its link relationship. Second, combining GPINN and kernel density estimation, we derive the bandwidth matrix updating rule for adapting to the generated network. We theoretically analyze the advantages of the proposed approach in determining threshold regions over using distance measures. The experimental results on both synthetic data sets and real-world data sets show that our method achieves remarkable improvement in density estimation accuracy for large noisy data.**

I. Introduction

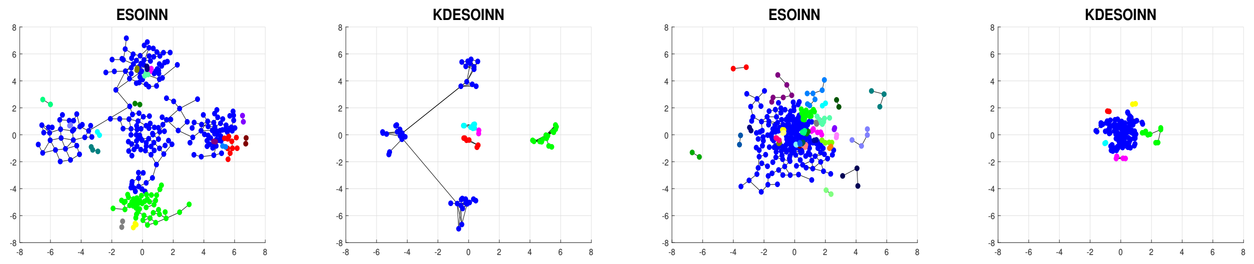
In general online Bayesian nonparametric mixture models [1]–[3], data becomes available in sequential order to update the parameters of mixture components sampled from conjugate prior parametric distributions (e.g., exponential families). In these models, the trained density function is based on partition space exploration. There, 1) the commonly used Gibbs samplers could suffer from poor mixing across partition space and non-convergence [4]–[6], and 2) each component is simulated by only one parametric distribution model, so that the knowledge of the underlying local distribution at each sample cannot be conserved in the learning process, thus limiting the performance for complicatedly distributed data. Moreover, in real-world data sets, data often includes substantial amounts of noise that can significantly affect the latent allocation variables and component parameters. Prevalent methods assume a parametric distribution of noise to solve the overfitting problem, but if the assumed distribution does not correspond to the true distribution of noise, the accuracy decreases severely. Thus, online learning algorithms that can successfully learn large, noisy data are required.

Several approaches have been developed to address the above requirements. Considering the knowledge that the data belonging to the same subclass lie near a much lower nonlinear dimensional manifold [7], [8], instead of exploring the partition space and using just one Gaussian distribution to simulate each component, a higher accuracy is expected to be achieved by learning the underlying manifold on which the probability density estimation is based. In manifold learning, the classic topology-preserving self-organizing mapping (SOM) is frequently employed [9]–[11]. SOM, in particular, generally adopts a 1-D or 2-D lattice to represent the input space of samples. However, the fixed number of nodes and the discontinuity associated with boundaries in SOM [12] make it undesirable for learning large data with unknown complex distributions.

Recently, based on competitive learning, SOM, and growing neural gas theory, researchers introduced self-organizing incremental neural networks (SOINN) [13]–[16]. During its learning process, the number of nodes and the link relationships are determined adaptively from the data, and the influence of noise is reduced based on these link relationships. The experimental results in [16] show that its density estimation accuracy outperforms the current state-of-the-art algorithms.

In SOINN, for each node, there is a corresponding threshold region determined either by the maximum Euclidean distance (ESOINN), or by Mahalanobis distance (KDESOINN) with its neighbors. An edge is then generated between a pair of nodes (i.e., the first and second winning nodes) when a new arrival falls into the intersection of their threshold regions. However, the threshold regions given by the distance measures – adopted by existing research – are shown to be either over-broad or over-narrow. Specifically, in ESOINN, the threshold region of one node is the interior of the sphere with radius defined by the maximum Euclidean distance from its neighbors. This definition does not comprehensively consider the distribution of the neighbors, so that numerous inappropriate edges are generated between nodes far away, and a large number of isolated subgraphs are distributed in the area with low probability density. This is shown in Figure 1(a,c). In KDESOINN instead, 1) the threshold regions given by Mahalanobis distance have extremely high fractional anisotropy values, which also results in a neural network with many inappropriate edges. Furthermore, 2) nodes are only distributed in a very small area with high probability density. This is shown in Figure 1(b,d). Please refer to Section II for detail.

By mapping data to a high-dimensional Hilbert space in implicit representations [17], [18], kernel functions are extensively employed to simulate complex relations between data points. Kernel density estimation (KDE) is then the most prevalent tool among density estimators for complex distributed data. However, for large, noisy data, the computational complexity of calculating the optimal bandwidth parameters for the popular cross-validation methods [19] is extremely



(a) **ESOINN** (b) **KDESOINN** (c) **ESOINN** (d) **KDESOINN**

Fig. 1. Sample Volume=2000,λ=500, (a)(b)$x \sim \frac{1}{5} \sum_{|x_1|+|x_2|=\{0,5\},x_1x_2=0} \mathcal{N}((x_1,x_2),\mathbf{I})$, (c)(d)$x \sim \frac{1}{2} \sum_{a\in\{1,5\}} (\mathcal{N}(\mathbf{0},a\mathbf{I}))$.

TABLE I
DEFINITIONS OF VARIABLES, PARAMETERS AND SYMBOLS

$\mathcal{N}(\mathcal{E})$	the set of nodes (edges)
$\mathcal{N}_{n_i}$	the set of the nodes connected with node n_i $\mathcal{N}_{n_i} \triangleq \{n_{1,i},...,n_{\lvert\mathcal{N}_{n_i}\rvert,i}\}$, $\lvert\mathcal{N}_{n_i}\rvert$ is the cardinality
D	the distance measure adopted
T_{n_i}	the threshold region of node n_i
r_{n_i}	the similarity threshold of n_i
W_{n_i}	the number of node n_i being the 1st winning node in competitive learning, that is the winning times of n_i
W	the set of the winning times of nodes, that is $\{W_{n_i}\}$
$age(n_i,n_j)$	the age of the edge linking node n_i and node n_j
age_{max}	a predetermined upper bound for deleting initially formed edges
$f(W_{n_j})$	gives the coefficient to update the position vector, generally $f(W_{n_j})=\frac{1}{100W_{n_j}}$,
λ	the number of inputs in one learning period

high, and KDE is well-known to be sensitive to the noise present. As far as we know, very little work has been done to address these issues.

The remainder of this paper is organized as follows. In Section II, we present the basic idea of SOINN and the shortcomings of previous work. In Section III, we 1) propose a novel online learning self-organizing neural network (GPINN) that adopts Gaussian process models in calculating winning nodes and threshold regions, and 2) we derive the kernel bandwidth matrix updating rule. Finally, we theoretically analyze the optimal threshold parameter selection. The experimental results in Section IV demonstrate that, compared with previous work, our approach achieves considerably higher density estimation accuracy.

II. OVERVIEW OF SOINN

Algorithm 1 shows the general learning process of various versions of SOINN, and Table I gives the definition of all variables, parameters, and symbols.

As an extension to SOM where the number of nodes $|\mathcal{N}|$ and link relationship $\mathcal{E}$ are fixed, in order to adapt to online machine learning, in SOINN, $\mathcal{N}$ and $\mathcal{E}$ are adjusted based on the positional relationship between new inputs and on the threshold regions of nodes. Specifically, when a new arrival s_i comes, first lines 2-4 calculate the nearest 2 nodes w_1, w_2 (named the first and second winning nods of s_i, respectively) based on the employed distance measure – ESOINN: Euclidean distance; KDESOINN: Mahalanobis distance – and compute their threshold regions $T_{w_{1(2)}}$. For node n_i, its threshold region

Algorithm 1 General SOINN

```
**Initialization:** $\mathcal{N} \leftarrow \{s_1,s_2\}, \mathcal{E} \leftarrow \emptyset$
 1: **while** $\sim isempty(s_i)$ **do**
 2:     $w_1 \leftarrow \arg\min_{n_i\in\mathcal{N}} D(s_i,n_i)$
 3:     $w_2 \leftarrow \arg\min_{n_i\in\mathcal{N}\setminus w_1} D(s_i,n_i)$
 4:     $r_{w_1}, r_{w_2} \leftarrow$ (1, 2)
 5:     **if** $D(s_i,w_1) > r_{w_1} \parallel D(s_i,w_2) > r_{w_2}$ **then**
 6:         $\mathcal{N} \leftarrow \mathcal{N} \cup \{s_i\}$
 7:     **else**
 8:         $\mathcal{E} \leftarrow \mathcal{E} \cup \{(w_1,w_2)\}$,
 9:         $age(w_1,w_2) = 0$
10:         $age(w_1,n_j) = age(w_1,n_j)+1 \quad \% \; n_j \in \mathcal{N}_{w_1}$
11:         $W_{w_1} \leftarrow W_{w_1}+1$
12:         $w_1 = w_1 + \frac{1}{W_{w_1}+1}(s_i - w_1)$
13:         $n_j = n_j + f(W_{w_1})(s_i - n_j)$
14:         $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \mid age(e) > age_{max}\}$
15:         $\mathcal{N} \leftarrow \mathcal{N} \setminus \{n_i \mid |\mathcal{N}_{n_i}| = 0\}$
16:     **end if**
17:     **if** $mod(i,\lambda) = 0$ **then**
18:         $\mathcal{N} \leftarrow \mathcal{N} \setminus \{s \mid |\mathcal{N}_s| = 0\}$
19:     **end if**
20: **end while**
```

$$T_{n_i} = \{x \mid D(x,n_i) \le r_{n_i}\}$$

where

$$r_{n_i} = \begin{cases} \max\limits_{n_j\in\mathcal{N}_{n_i}} D(n_j,n_i), & \text{if } |\mathcal{N}_{n_i}| \neq 0 \\ \min\limits_{n_j\in\mathcal{N}\setminus n_i} D(n_j,n_i), & \text{if } |\mathcal{N}_{n_i}| = 0, \end{cases}$$

$$D(x,n_i) = \begin{cases} \|x-n_i\|_2, & \text{ESOINN} \\ \sqrt{(x-n_i)^T M_{n_i}^{-1}(x-n_i)}, & \text{KDESOINN}, \end{cases} \tag{1}$$

and

$$M_{n_i} = \frac{1}{\sum_{j=1}^{|\mathcal{N}_{n_i}|} W_{n_{j,i}}} \sum_{j=1}^{|\mathcal{N}_{n_i}|} W_{n_{j,i}} (n_{j,i} - n_i)(n_{j,i} - n_i)^T + \sigma^2 I, \tag{2}$$

σ is a fixed very small number. Then, depending on whether $s_i \in T_{w_1} \cap T_{w_2}$, lines 5-16 update the network parameters (e.g., the node set $\mathcal{N}$, or the position vector of w_1 and its neighbors). Considering that the initially formed edges may not be appropriate as the input of the samples, one simple but effective method is to delete the initially formed edges when their ages come to an upper bound age_{max}. After each learning period (i.e., the number of input samples is an integer multiple of λ), outliers defined with degree 0 are deleted at line 18.

In ESOINN, the threshold region is determined by the maximum Euclidean distance from the connected neighbors. However, 1) because of the curse of dimensionality, and 2) because the underlying much lower dimensional nonlinear manifold [7], [8], ESOINN can 1) generate numerous inappropriate edges between nodes far away, measured by geodesic distance on the manifold, and 2) many subgraphs with very few nodes are isolated from the main graph, and distributed in the area with low probability density. The recently proposed KDESOINN attempted to solve the problems in ESOINN by employing Mahalanobis distance. However, the first principal component derived from the covariance matrix given in (2) can be greatly affected by the inappropriate edges. For this reason, first, T_{n_i} has high fractional anisotropy value, and seriously neglects the probability of the subsequent samples falling into the span of the principal axes of T_{n_i} with small eigenvalues. Second, many initially formed inappropriate edges are conserved. Third, many of the temporarily remained representative nodes are mistaken as outliers and deleted after each learning period. As shown in the experiments, the nodes of the generated network are distributed in a very small area with high probability density.

III. Proposed Method

In this section, we comprehensively explain the Gaussian process-based incremental neural network, its kernel bandwidth matrix updating rule, and its advantages compared to using distance measures as done in previous research.

A. Gaussian Process-based Calculations of Winning Nodes and Threshold Regions

Gaussian processes (GPs) [20]–[22] extend multivariate Gaussian distributions to infinite dimensionality, and can be used to represent the underlying function rigorously in a non-parametric form. Formally, a Gaussian process generates data located at the domain where any finite subset of the dependent variables is subject to a multivariate Gaussian distribution. Specifically, $\mathbf{y} = \{y_{x_1}, y_{x_2}, ..., y_{x_n}\}$, the observations of a dependent variable at n values of the independent variable $\mathbf{x} = \{x_1, x_2, ..., x_n\}$, is sampled from a $n-$dimensional Gaussian distribution with a kernel matrix as its covariance and mean $\mathbf{0}_{n\times 1}$ very often. So, given $\mathbf{x}$ and $\mathbf{y}$, using Gaussian process regression (GPR), the prediction of the dependent variable at a new value x_0, y_{x_0}, follows the Gaussian distribution

$$y_{x_0} \mid \mathbf{x}, \mathbf{y} \sim N(K_{x_0,\mathbf{x}} K_{\mathbf{x}}^{-1} \mathbf{y}, K_* - K_{x_0,\mathbf{x}} K_{\mathbf{x}}^{-1} K_{x_0,\mathbf{x}}^T) \tag{3}$$

where

$$K_{x_0,\mathbf{x}} = \begin{bmatrix} k(x,x_1) & k(x,x_2) & \cdots & k(x,x_n) \end{bmatrix}, K_* = k(x,x), \tag{4}$$

$$K_{\mathbf{x}} = \begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & \cdots & k(x_1,x_n) \\ k(x_2,x_1) & k(x_2,x_2) & \cdots & k(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n,x_1) & k(x_n,x_2) & \cdots & k(x_n,x_n) \end{bmatrix}, \tag{5}$$

and k is a kernel function. Since using a fixed bandwidth scalar or matrix leads to a sub-optimal rate of convergence [23]–[26], this paper adopts an adaptive bandwidth matrix M_{n_i}, that is, $\forall n_j, n_k \in \mathcal{N}_{n_i}$,

$$k_i(n_j, n_k) = \exp\big\{ -\frac{1}{2}(n_j - n_k)^T M_{n_i}^{-1} (n_j - n_k) \big\}, \tag{6}$$

where M_{n_i} is calculated in subsection *B*.

GPs have the property that when $x_i \in \mathbf{x}$ is distant from $\{x_0\} \cup \mathbf{x} \setminus x_i$, the divergence between $y_{x_0}|(\mathbf{x}, \mathbf{y})$ and $y_{x_0}|(\mathbf{x} \setminus x_i, \mathbf{y} \setminus y_{x_i})$ is very small. Applying Gaussian process classification (GPC) to SOINN, for node n_i, with the position vector as the independent variable, and the link relationship with n_i as the dependent variable, the nodes in $\mathcal{N}_i$ far away from n_i have little influence on inferring the link relationship between a new arrival and n_i. This means that such an incremental neural network is robust to inappropriate edges, and can solve the high fractional anisotropy problem in KDESOINN.

Specifically, let's assume that an incremental network $\mathcal{I} \triangleq (\mathcal{N}, \mathcal{E}, W)$ (please refer to Table I) is generated after the input of the first $m-1$ training samples (the training samples are d-dimensional). Then, for new arrival s_m, according to Bayes theorem, the probability of node n_i being chosen as the winning node of s_m is

$$P(n_i \mid s_m, \mathcal{I}) \propto P(n_i \mid \mathcal{I}) P(s_m \mid \mathcal{N}_{n_i}). \tag{7}$$

There, $P(n_i \mid \mathcal{I})$ is the probability of drawing n_i from $\mathcal{I}$, and is proportional to its winning times W_{n_i}. $P(s_m \mid \mathcal{N}_{n_i})$ is the probability of s_m linking with n_i given the distribution of its neighbors $\mathcal{N}_{n_i}$, and is calculated by a GPC model with $\Phi(z|0, \delta_1^2)$ (the cumulative normal distribution function with mean 0 and variance δ_1^2) as the squashing function, that is

$$\begin{aligned} P(s_m \mid \mathcal{N}_{n_i}) &= \int \Phi(z|0, \delta_1^2) N(z | E_{s_m,n_i}, V_{s_m,n_i}) dz \\ &= \Phi\left(\frac{K_{s_m,n_i} K_{n_i}^{-1} \mathbf{y}}{\sqrt{1 + \delta_1^2 - K_{s_m,n_i} K_{n_i}^{-1} K_{s_m,n_i}^T}} \right) \end{aligned} \tag{8}$$

where

$$\begin{aligned} K_{s_m,n_i} &= \left[k_i(s_m,n_{1,i}) \;\; k_i(s_m,n_{2,i}) \;\; \cdots \;\; k_i(s_m,n_{|\mathcal{N}_{n_i}|,i})\right]^T, \\ E_{s_m,n_i} &= K_{s_m,n_i} K_{n_i}^{-1} \mathbf{y}_{\mathcal{N}_{n_i}}, \\ V_{s_m,n_i} &= 1 - K_{s_m,n_i}^T K_{n_i}^{-1} K_{s_m,n_i}, \end{aligned} \tag{9}$$

and $K_{n_i} = [k(n_{p,i}, n_{q,i})]_{1 \le p,q \le |\mathcal{N}_{n_i}|}$.

Regarding the values of the dependent variable $\mathbf{y}_{\mathcal{N}_{n_i}}$ at $\mathcal{N}_{n_i}$, for $\forall n_j \in \mathcal{N}_{n_i}$, $P(n_j|\mathcal{N}_{n_i})$ is set to a constant α_1, and since K_{x,n_i} is a row of K_{n_i}, from the property of determinant calculation, we have

$$\begin{aligned} \mathbf{y}_{\mathcal{N}_{n_i}} &= \left[\Phi^{-1}(\alpha_1)\sqrt{1+\delta_1^2 - K_{n_j,n_i}K_{n_i}^{-1}K_{n_j,n_i}^T}\right]_{n\times 1} \\ &= \Phi^{-1}(\alpha_1)\delta_1 \mathbf{1}_{n\times 1}, \end{aligned} \tag{10}$$

thus (7) is equivalent to

$$P(n_i \mid s_m, \mathcal{I}) \propto W_{n_i} \Phi\left(\frac{\Phi^{-1}(\alpha_1)\delta_1 K_{s_m,n_i}K_{n_i}^{-1}\mathbf{1}_{n\times 1}}{\sqrt{1+\delta_1^2 - K_{s_m,n_i}K_{n_i}^{-1}K_{s_m,n_i}^T}}\right). \tag{11}$$

T_{n_i} is given by setting a threshold β_i (discussed in subsection *C*) for $P(x|\mathcal{N}_{n_i})$,

$$\begin{aligned} T_{n_i} =& \{x \mid P(x \mid \mathcal{N}_{n_i}) \ge \beta_i\} \\ =& \left\{x \mid \frac{\delta_1 K_{x,n_i}K_{n_i}^{-1}\mathbf{1}_{n\times 1}}{\sqrt{1+\delta_1^2 - K_{x,n_i}K_{n_i}^{-1}K_{x,n_i}^T}} \ge \frac{\Phi^{-1}(\beta_i)}{\Phi^{-1}(\alpha_1)}\right\}. \end{aligned} \tag{12}$$

B. Optimal Bandwidth Matrices and Online Kernel Density Estimation

1) Bandwidth Matrix Optimization: Since calculating M by directly maximizing $\prod_{n_j\in\mathcal{N}} (P(n_j \mid \mathcal{N}, M))^{W_{n_j}}$ can cause overfitting [27], [28], the leave-one-out method is employed here, that is

$$M_{n_p} = \arg\max_{M_{n_p}} \prod_{n_j\in\mathcal{N}} (P(n_j \mid \mathcal{N}\setminus\{n_j\}, M\setminus\{M_{n_j}\}))^{W_{n_j}}. \tag{13}$$

One natural idea is to calculate the derivative of its log-likelihood:

$$\begin{aligned} &\nabla_{M_{n_p}} \sum_{n_j} W_{n_j}\log P(n_j \mid \mathcal{N}\setminus\{n_j\}, M\setminus\{M_{n_j}\}) \\ &= \sum_{j\ne p} \frac{W_{n_p}W_{n_j}M_{n_p}^{-1}(M_{p,j}-M_{n_p})M_{n_p}^{-1}k_p(n_j,n_p)}{2|M_{n_p}|^{\frac{1}{2}}P(n_j \mid \mathcal{N}\setminus\{n_j\}, M\setminus\{M_{n_j}\})}, \end{aligned} \tag{14}$$

where $M_{p,j} = (n_p-n_j)(n_p-n_j)^T$. By setting it equal to 0, one finds

$$M_{n_p} = \frac{\sum_{j\ne p} W_{n_j}\frac{(n_p-n_j)(n_p-n_j)^T k_p(n_j,n_p)}{\sum_{i\ne j} W_{n_i}k_i(n_j,n_i)|M_{n_i}|^{-\frac{1}{2}}}}{\sum_{j\ne p} W_{n_j}\frac{k_p(n_j,n_p)}{\sum_{i\ne j} W_{n_i}k_i(n_j,n_i)|M_{n_i}|^{-\frac{1}{2}}}}. \tag{15}$$

It is a transcendental equation, hence we employ an EM algorithm. After some derivations, we have

E-Step:

$$c_{j,p} = \frac{W_{n_p}k_p(n_j,n_p)|M_{n_i}|^{-\frac{1}{2}}}{\sum_{q\ne j} W_{n_q}k_q(n_j,n_q)|M_{n_q}|^{-\frac{1}{2}}} \tag{16}$$

M-Step:

$$M_{n_p} = \frac{\sum_{j\ne p} W_{n_j}c_{j,p}(n_p-n_j)(n_p-n_j)^T}{\sum_{j\ne p} W_{n_j}c_{j,p}}. \tag{17}$$

From the perspective of kernel density estimation, $c_{j,p}$ refers to the probability of one sample located at n_j being drawn from the kernel component at n_p. Considering the underlying lower dimensional manifold, the kernel component at n_p should be aligned with the plane locally tangent to this underlying manifold, and the information about this tangent plane can be gathered from the neighboring points of n_p [7], [8]. So, from Bayes theorem, $c_{j,p}$ can be reasonably approximated as

$$\hat{c}_{j,p} \approx \begin{cases} \frac{P(n_p|n_j,\mathcal{I})}{\sum_{n_k\in\mathcal{N}_{n_j}} P(n_k|n_j,\mathcal{I})} = \frac{W_{n_p}}{\sum_{n_k\in\mathcal{N}_{n_j}} W_{n_k}}, & n_p \in \mathcal{N}_{n_j} \\ 0, & n_p \notin \mathcal{N}_{n_j}. \end{cases} \tag{18}$$

Thus, M_{n_p} can be calculated in a significantly simplified form according to the following 3 steps:

(1) *Preliminary Approximation.* Substituting (18) into (17), we obtain the preliminary approximation of M_{n_p}

$$M_{n_p}^{'} = \frac{\sum_{n_i\in N_p\cap\mathcal{N}_{n_p}} \frac{W_{n_i}}{\sum_{n_j\in N_i\cap\mathcal{N}_{n_i}} W_{n_j}}(n_i-n_p)(n_i-n_p)^T}{\sum_{n_i\in N_p\cap\mathcal{N}_{n_p}} \frac{W_{n_i}}{\sum_{n_j\in N_i\cap\mathcal{N}_{n_i}} W_{n_j}}}. \tag{19}$$

where $N_{p(i)}$ is the set of the first $\max\{|\mathcal{N}_{n_{p(i)}}|, 3d\}$-nearest nodes of $n_{p(i)}$ measured by Euclidean distance (d is the dimensionality of data points), and we adopt $N_{p(i)} \cap \mathcal{N}_{n_{p(i)}}$ here to exclude possible inappropriate edges.

(2) *Positive Definiteness Adjustment.* In order to guarantee the positive definiteness of M_{n_p}, we add a small isotropic (spherical) Gaussian noise of variance in all directions,

$$M_{n_p} = M_{n_p}^{'} + \frac{\rho}{d}\min_{n_j\in\mathcal{N}} \|n_j-n_p\|_2^2 \mathbf{I}_d,. \tag{20}$$

where $0 < \rho \le 1$ ($\rho = 0.1$ in this paper).

(3) *Bandwidth Matrix Smoothing.* For the sake of, firstly, simulating the underlying manifold and local distribution around each node more accurately, and, secondly, further decreasing the influence of inappropriate edges on M_{n_p} calculation, we improve the similarity between the threshold regions of connected nodes instead of regarding them as independent. Because different nodes n_i have different winning times W_{n_i}, instead of considering only the spatial distribution of nodes, we set an estimated density $e_i \triangleq \frac{W_{n_i}}{|M_{n_i}|V_d}$ (V_d is the volume of d-dimensional unit sphere) as the value of the dependent variable at each node $n_i \in \mathcal{N}$, and then adopt GPR to calculate

the likelihood of e_i given the values at $\mathcal{N}_{n_i}$. Specifically, after each learning period, if

$$\sum_{j=1}^{|\mathcal{N}_{n_i}|}\frac{W_{n_{j,i}}}{\sum_{j=1}^{|\mathcal{N}_{n_i}|}W_{n_{j,i}}}\Phi\left(\left|\frac{\frac{W_{n_i}}{|M_{n_i}|V_d}-K_{n_i,n_{j,i}}K_{n_{j,i}}^{-1}\frac{W_{n_{j,i}}}{|M_{n_{j,i}}|V_d}}{\sqrt{1-K_{n_i,n_{j,i}}K_{n_{j,i}}^{-1}K_{n_i,n_{j,i}}^T}}\right|\right) \geq \frac{1}{2}(erf(\frac{\alpha_2}{\sqrt{2}})+1), \tag{21}$$

where *erf* is the error function and $\alpha_2 \triangleq 2$ in this paper, M_{n_i} is adjusted as

$$M_{n_i}=\frac{1}{\sum_{j=1}^{|\mathcal{N}_{n_i}|}W_{n_{j,i}}}\sum_{j=1}^{|\mathcal{N}_{n_i}|}M_{n_{j,i}}W_{n_{j,i}}. \tag{22}$$

2) Online Kernel Density Estimator: Based on kernel density theory, the probability density estimation at point y is thus expressed as

$$P(y)=\frac{1}{\sum_{j=1}^{|\mathcal{N}|}W_{n_j}}\sum_{i=1}^{|\mathcal{N}|}\frac{W_{n_i}\exp\{-\frac{1}{2}(n_i-y)^TM_{n_i}^{-1}(n_i-y)\}}{\sqrt{(2\pi)^d|M_{n_i}|}}. \tag{23}$$

C. Discussion of Thresholds β_i

Consider the property that GPC is robust to insignificant edges, for each node n_i, $\lim_{n\to+\infty}\max_{n_{j1},n_{j2}\in\mathcal{N}_i\cup n_i}\frac{W_{n_{j1}}}{W_{n_{j2}}}=1$ (n is the number of training samples), and $\mathcal{N}_{n_i}$ are fine-tuned and tend to evenly distribute on a spherical surface $\mathcal{S}_{n_i}$ in $\{n_i+Q\}$ (for a set $\mathcal{A}$, $\{n_i+\mathcal{A}\}\triangleq\{n_i+a|a\in\mathcal{A}\}$, $Q\triangleq Span\{n_{j,i}-n_i \mid n_{j,i}\in\mathcal{N}_{n_i}\}$ and assume its dimensionality is d_1). So, to simplify the theoretical analysis, we assume that the cardinality of $\{W_{n_j\in\hat{N}_i}\}$ is 1 and $\mathcal{N}_{n_i}$ forms a regular polytope with center n_i and outer d_1-radius h.

Then, we have

Theorem 1. *When $P(n_i \mid \mathcal{N}_{n_i})=\beta_i$,*

$$d^*(T_{n_i},\mathcal{N}_{n_i})=h. \tag{24}$$

where for set A and set B, $d^*(A,B)=\sup_{x\in A}\inf_{y\in B}\|x-y\|_2$.

Proof.
1). Since $n_i\in T_{n_i}$, $d^*(T_{n_i},\mathcal{N}_{n_i})\geq h$.
2). from (8),

$$\begin{aligned}&P(x\mid\mathcal{N}_{n_i})<\beta_i,\\ \Leftrightarrow\ &\delta_1\frac{\Phi^{-1}(\alpha_1)}{\Phi^{-1}(\beta_i)}K_{x,n_i}K_{n_i}^{-1}\mathbf{1}_{|\mathcal{N}_{n_i}|\times 1}<\sqrt{1+\delta_1^2-K_{x,n_i}K_{n_i}^{-1}K_{x,n_i}^T}\\ \Leftrightarrow\ &K_{x,n_i}(K_n^{-1}+\frac{\delta_1^2(\Phi^{-1}(\alpha_1))^2}{c_1^2(\Phi^{-1}(\beta_i))^2}\mathbf{1}_{|\mathcal{N}_{n_i}|\times|\mathcal{N}_{n_i}|})K_{x,n_i}^T<1+\delta_1^2,\end{aligned} \tag{25}$$

where $c_1=\sum_{j=1}^{|\mathcal{N}_{n_i}|}k_i(n_{j,i},n_{1,i})$ and the second $\Leftrightarrow$ is derived from the regular polytope assumption.
3). For point $\mathbf{p}$, $s_p\triangleq\arg\min_{x\in\mathcal{N}_{n_i}}\|\mathbf{p}-x\|_2$, decompose $\mathbf{p}-s_p=\mathbf{p}_1+\mathbf{p}_2$, where $s_p+\mathbf{p}_1\in\{n_i+Q\}$ and $\mathbf{p}_2\in Q^\perp$. Considering the form of M_{n_i} given by (19), (20), we have the following property

$$K_{\mathbf{p},n_i}=\exp\{-\frac{1}{2}\mathbf{p}_2^TM_{n_i}^{-1}\mathbf{p}_2\}K_{s_p+\mathbf{p}_1,n_i}. \tag{26}$$

Thus

$$\begin{aligned}&K_{\mathbf{p},n_i}K_{n_i}^{-1}K_{\mathbf{p},n_i}=\exp\{-\mathbf{p}_2^TM_{n_i}^{-1}\mathbf{p}_2\}K_{s_p+\mathbf{p}_1,n_i}K_{n_i}^{-1}K_{s_p+\mathbf{p}_1,n_i}^T\\ &\leq\exp\{-\mathbf{p}_2^TM_{n_i}^{-1}\mathbf{p}_2\}\exp\{-\mathbf{p}_1^TM_{n_i}^{-1}\mathbf{p}_1\}\mathbf{1}_{1\times|\mathcal{N}|_{n_i}}K_{n_i}^{-1}\mathbf{1}_{1\times|\mathcal{N}|_{n_i}}\\ &\leq\exp\{-\frac{d(\|\mathbf{p}-s_p\|_2^2-h^2)}{h^2}\}K_{n_i,n_i}K_{n_i}^{-1}K_{n_i,n_i},\end{aligned} \tag{27}$$

the second $\leq$ results from the condition $0<\rho\leq 1$.
4). For $\forall\mathbf{p}_1$ satisfying $\inf_{y\in\mathcal{N}_{n_i}}\|\mathbf{p}_1-y\|_2>h$,

$$K_{\mathbf{p}_1,n_i}\mathbf{1}_{|\mathcal{N}|_{n_i}\times|\mathcal{N}|_{n_i}}K_{\mathbf{p}_1,n_i}^T<K_{n_i,n_i}\mathbf{1}_{|\mathcal{N}|_{n_i}\times|\mathcal{N}|_{n_i}}K_{n_i,n_i}. \tag{28}$$

Define the left-hand side of (25) as $f_{n_i}(x)$. When $P(n_i\mid\mathcal{N}_{n_i})=\beta_i$, multiple $\exp\{\frac{d(\|\mathbf{p}_1-s_p\|_2^2-h^2)}{h^2}\}$ at both sides of (27) and sum with (28), we have

$$f_{n_i}(\mathbf{p}_1)<f_{n_i}(n_i)=1+\delta_1^2. \tag{29}$$

From (25), it is equivalent to $P(\mathbf{p}_1\mid\mathcal{N}_{n_i})<\beta_i$, that means $d^*(T_{n_i},\mathcal{N}_{n_i})\leq h$.

Combining 1) and 4), $d^*(T_{n_i},\mathcal{N}_{n_i})=h$. □

Theorem 2. *When $\beta_i=P(n_i\mid\mathcal{N}_{n_i})$ and $d_1<d$, for $\hat{Q}\triangleq T_{n_i}\cap(\cup_{s\in\mathcal{N}_{n_i}}\{s+Q^\perp\})$,*

$$d^*(\hat{Q},\mathcal{N}_{n_i})\leq h\sqrt{\rho}. \tag{30}$$

Proof. For $\forall s\in\partial\hat{Q}$ ($\partial\hat{Q}$ means the boundary of $\hat{Q}$), by substituting n_i into (12) and from the property (26), one can obtain an upper bound of $d^*(s,\mathcal{N}_{n_i})$ (denoted as $\hat{h}$) that satisfies

$$\frac{\exp\{-\frac{d\hat{h}^2}{2\rho h^2}\}}{\sqrt{1+\delta_1^2-\exp\{-\frac{d\hat{h}^2}{\rho h^2}\}}}=\frac{\frac{e^{-\frac{d}{2}}|\mathcal{N}_{n_i}|}{\sum_{j=1}^{|\mathcal{N}_{n_i}|}k_i(n_{j,i},n_{1,i})}}{\sqrt{1+\delta_1^2-\frac{e^{-d}|\mathcal{N}_{n_i}|}{\sum_{j=1}^{|\mathcal{N}_{n_i}|}k_i(n_{j,i},n_{1,i})}}}, \tag{31}$$

and since $\frac{x}{\sqrt{1+\delta_1^2-x^2}}$ ($0\leq x\leq\sqrt{1+\delta_1^2}$) is a monotonically increasing function of x, we have

$$\exp\{-\frac{d\hat{h}^2}{2\rho h^2}\}>e^{-\frac{d}{2}}\sqrt{\frac{|\mathcal{N}_{n_i}|}{\sum_{j=1}^{|\mathcal{N}_{n_i}|}k_i(n_{j,i},n_{1,i})}}>e^{-\frac{d}{2}},$$

that is $\hat{h}<h\sqrt{\rho}$, (30) is proven. □

These theorems show when $\beta_i=P(n_i\mid\mathcal{N}_{n_i})$, $d^*(\hat{Q},\mathcal{N}_{n_i})$ is smaller than h. This means that our algorithm can assign different link-generating probabilities to different directions based on the distribution of neighbors, so that the over-broad threshold region problem in ESOINN can be solved. Besides, for the continuity of GP posterior probability function and the robustness of T_{n_i} to inappropriate edges, the proposed method can also effectively overcome the shortcomings of KDESOINN described above.

In the learning process, considering the variance of the distribution of $\mathcal{N}_{n_i}$, in our algorithm, β_i is given as

$$\beta_i = \max\{\min_{n_j \in \mathcal{N}_{n_i}} P(n_j|\mathcal{N}_{n_i} \setminus n_j), P(n_i|\mathcal{N}_{n_i})\}, \tag{32}$$

and actually, for a large $|\mathcal{N}_{n_i}|$,

$$\beta_i \approx \Phi(\frac{\delta_1 \Phi^{-1}(\alpha_1)}{\sqrt{\gamma^2(1+\delta_1^2)e^{d_1} - \gamma}}) \tag{33}$$

where

$$\gamma = \frac{\Gamma(\frac{d_1}{2}) \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \exp\{d_1 \sin\theta\}(\cos\theta)^{d_1-2} d\theta}{\sqrt{\pi} e^{d_1} \Gamma(\frac{d_1-1}{2})}. \tag{34}$$

D. Algorithm: Gaussian Process-based Self-organizing Incremental Neural Network (GPINN)

Based on the proposed 1) winning node and threshold region calculations, and 2) bandwidth matrix updating rule, a Gaussian Process-based self-organizing incremental neural network (GPINN) is achieved. The Algorithm is shown in 2.

First, in order to reduce the searching time of winning nodes, we preliminarily determine the set of candidates $\mathcal{A}$ for new arrivals s_i, using Euclidean distance at line 2. Second, we further identify the first 2 winning nodes w_1 and w_2 by calculating the probability of connecting with s_i using (11) in lines 3-4. Third, depending on whether $s_i \in T_{w_1} \cap T_{w_2}$ by (12), (32) at line 5, the network parameters (e.g., nodes, position vectors, and edges) are updated in lines 5-16 where the kernel bandwidth matrix is adjusted as (19), (20) at line 15. Fourth, at the end of each learning period, the bandwidth matrices are modified according to (21), (22) in lines 18-22. Then we re-check whether the currently maintained edges are appropriate or not by comparing $P(n_{k(j)} \mid \mathcal{N}_{n_{j(k)}} \setminus n_{k(j)})$ and $P(n_{j(k)} \mid \mathcal{N}_{n_{j(k)}})$ using (8) in lines 23-29. In addition to this, in order to make the currently achieved latent manifold more robust in our online learning network and following the idea of the manifold learning algorithms in [29]–[31], we add some edges between the nodes that have duplex edges in a k-NN graph ($k \triangleq 3d$) on the set of nodes $\mathcal{N}$ at line 31.

Complexity: 1). In the winning node and threshold region calculations, the complexity of line 2 is $O(|\mathcal{N}|)$. Regarding the GPC involved in line 3-4, instead, the time complexity for calculating $\{K_{n_{\phi_j}}^{-1}\}$ is $O(age_{max}^3|\mathcal{A}|)$. It is worth noting that $|\mathcal{A}| \leq 3d$ and age_{max} is a predetermined value, so it is $O(1)$. 2). In the adjustment process for the network parameters, the complexity of $\{M_{n_i}\}$ modification in lines 18-22, and the inappropriate edges deletion in line 23-29 are both $O(|\mathcal{N}|)$. The complexity in obtaining the k-NN graph is $O(|\mathcal{N}|^2)$. The time complexity of other lines are $O(1)$.

Algorithm 2 GPINN

```
Initialization: $\mathcal{N} \leftarrow \{s_1, s_2\}, \mathcal{E} \leftarrow \emptyset$
 1: while $\sim isempty(s_i)$ do
 2:   $\mathcal{A} \leftarrow arg \min_{\{n_{\phi_{1:\min(i,3d)}}\} \subset \mathcal{N}} \sum_{j=1}^{\min(i,3d)} \|s_i - n_{\phi_j}\|_2$
      %$\{\phi_j\}$ the indexes of the nodes in $\mathcal{A}$
 3:   $\zeta_1 \xleftarrow{(11)} \arg\max_{\phi_j} P(n_{\phi_j} \mid s_i, \mathcal{I}), w_1 \leftarrow n_{\zeta_1}$
 4:   $\zeta_2 \xleftarrow{(11)} \arg\max_{\phi_k \in \{\phi_j\} \setminus \zeta_1} P(n_{\phi_k} \mid s_i, \mathcal{I}), w_2 \leftarrow n_{\zeta_2}$
 5:   if $P(s_i \mid \mathcal{N}_{w_1}) \leq \beta_{\zeta_1} \,||\, P(s_i \mid \mathcal{N}_{w_2}) \leq \beta_{\zeta_2}$%(12),(32) then
 6:     $\mathcal{N} \leftarrow \mathcal{N} \cup \{s_i\}$
 7:   else
 8:     $\mathcal{E} \leftarrow \mathcal{E} \cup \{(w_1, w_2)\}$
 9:     $age(w_1, w_2) = 0$
10:     $age(w_1, n_j) = age(w_1, n_j) + 1$   % $n_j \in \mathcal{N}_{w_1}$
11:     $W_{w_1} \leftarrow W_{w_1} + 1, w_1 = w_1 + \frac{1}{W_{w_1}+1}(s_i - w_1)$
12:     $n_j = n_j + f(W_{w_1})(s_i - n_j)$   % $n_j \in \mathcal{N}_{w_1}$
13:     $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e \mid age(e) > age_{max}\}$
14:     $\mathcal{N} \leftarrow \mathcal{N} \setminus \{n_i \mid |\mathcal{N}_{n_i}| = 0\}$
15:     $M_{n_i} \leftarrow$ (19), (20)
16:   end if
17:   if $mod(i, \lambda) = 0$ then
18:     for $j = 1 : |\mathcal{N}|$ do
19:       while (21) holds do
20:         $M_j \leftarrow$ (22)
21:       end while
22:     end for
23:     for $j = 1 : |\mathcal{N}|$ do
24:       for $n_k \in \mathcal{N}_{n_j}$ do
25:         if $P(n_k \mid \mathcal{N}_{n_j} \setminus n_k) \leq P(n_j \mid \mathcal{N}_{n_j})$&&$P(n_j \mid \mathcal{N}_{n_k} \setminus n_j) \leq P(n_k \mid \mathcal{N}_{n_k})$ %(8) then
26:           $\mathcal{E} \leftarrow \mathcal{E} \setminus \{(n_j, n_k), (n_k, n_j)\}$
27:         end if
28:       end for
29:     end for
30:     $\mathcal{N} \leftarrow \mathcal{N} \setminus \{s \mid |\mathcal{N}_s| < 0\}$
31:     $\mathcal{E} \leftarrow \mathcal{E} \cup \{(n_{j1}, n_{j2}) \mid \{(n_{j1}, n_{j2}), (n_{j2}, n_{j1})\} \subset E\}$, %$(V, E) =$ k-NNG($\mathcal{N}$)
32:     $M_{n_i} \leftarrow$ (19), (20)
33:   end if
34: end while
```

IV. Experimental Results

In order to demonstrate the performance of GPINN, we have conducted experiments on both synthetic data and real-world data. Since [16] shows that KDESOINN outperforms or achieves performance comparable to the current state-of-the-art approaches in terms of density estimation accuracy, we took it as benchmark. Moreover, considering that so far there existed no density estimation method based on ESOINN, we calculated the bandwidth matrix at its nodes using the proposed optimization method in (19), (20). Hence, besides GPINN, this paper also gives another density estimator by

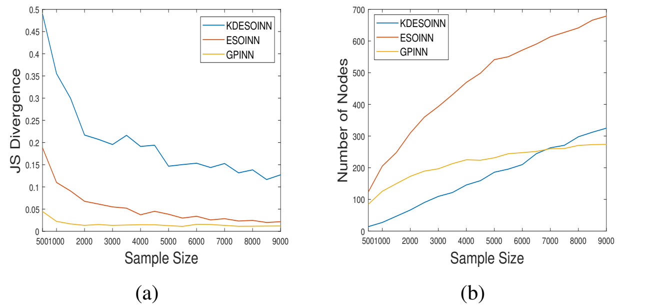


Fig. 2. Multimodal data: (a) The change in Jensen-Shannon Divergence with sample size; (b) The change in the number of nodes with sample size.

combining ESOINN and kernel density estimation theory.

A. Synthetic data

1) Multimodal data: The training data are sampled from $f(x_1, x_2)$ composed of 5 Gaussian distributions that have the same covariance matrix, but different mean values. In addition, we have added as noise a 5% contaminating distribution (Gaussian distribution with the same mean and 50 times the covariance matrix as those of $f(x_1, x_2)$). That is

$$\mathbf{x} \sim 0.95 \sum_{|x_1|+|x_2|=\{0,5\}, x_1 x_2=0} \frac{1}{5}\mathcal{N}((x_1, x_2), \mathbf{I}) + 0.05 f_{noise}.$$

2) Multiscale data: The training data are randomly sampled from $f(x_1, x_2)$ composed of 2 Gaussian distributions with the same mean value but different covariance matrices. Similarly, 5% noise is mixed as above,

$$\mathbf{x} \sim 0.95 \sum_{s\in\{0,5\}} \frac{1}{2}\mathcal{N}((0,0), s\mathbf{I}) + 0.05 f_{noise}.$$

$agemax = 10, \lambda = 500, \delta_1^2 = 2$ in both cases. In order to compare these 3 approaches comprehensively, we have tested on different training sample size. In the multimodal case, the size ranges from 500 to 9000, in the multiscale case, it ranges from 1000 to 5000. Furthermore, for each input data size, we tested 10 times and took the average Jensen-Shannon (JS) divergence between the estimated density function and the true density function. The changes in JS divergence and the number of nodes with sample size are depicted in Figure 2 and Figure 3, respectively.

As the results show, GPINN gives the most accurate density estimation among the three methods. In correspondence with our analysis above, at the initial training process, KDESOINN performs the worst since most of the currently maintained nodes are deleted after the first learning periods. This corresponds to the nearly linear increase in $|\mathcal{N}|$ of KDESOINN at the initial learning stage. Thus, a large input size is required to simulate the underlying distribution. Moreover, although ESOINN generates a neural network with large amounts of nodes for its over-broad threshold regions, the noise data cannot be efficiently detected which greatly reduces its estimation accuracy. In contrast, GPINN reasonably considers the

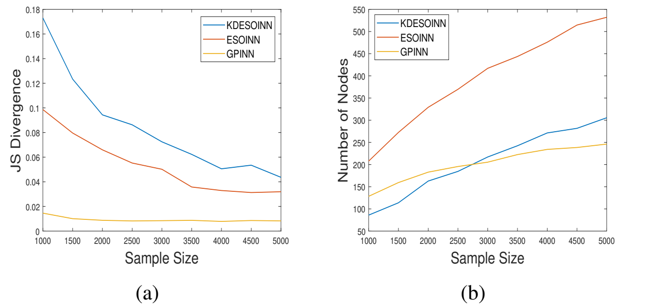


Fig. 3. Multiscale data: (a) The change in Jensen-Shannon Divergence with sample size; (b) The change in the number of nodes with sample size.

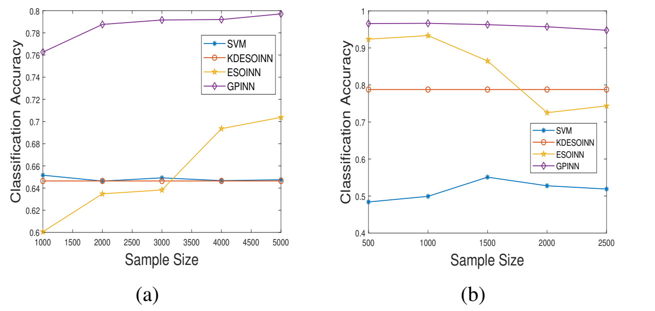


Fig. 4. (a) The change in Classification Accuracy with sample size on Gamma Telescope Data Set with 5% Noise; (b) The change in Classification Accuracy with sample size on Occupancy Detection Data set with 10% Noise.

different probabilities of the incoming samples falling along different principal directions, and can learn the probability distribution much more accurately.

B. Real-world Data Sets

We have tested our methodology on two real-world data sets from UCI ML Repository: 1) the MAGIC Gamma Telescope data set (10 dimensional, 2 classes) and 2) the Occupancy Detection data set (5 dimensional without the time attribute, 2 classes). Additionally, 5% and 10% noise is added to the MAGIC Gamma Telescope data set and to the Occupancy Detection data set, respectively. We have first estimated the probability density functions using different approaches for each class, and then determined the labels for the test data by a naïve Bayes classifier. For each training sample size, we tested 10 times and recorded the average accuracy.

Figure 4 gives the results of KDESOINN, ESOINN and GPINN. We have used $agemax = 10, \delta_1^2 = 2$, and $\lambda = 500$ for the Occupancy Detection data set, or $\lambda = 1000$ for the Telescope data set). In addition, the performance of the commonly used SVM is given as a comparison. The no-node-left states of KDESOINN clearly reveal the high fractional anisotropy shortcoming of the threshold region in KDESOINN (where the default accuracy is given). It is worth noting that, although SVM is a batch learning, the noise data can greatly affect the location of separating hyper-plane. This can very easily cause over-fitting. In fact, for the noisy occupancy detection data set, SVM gives the lowest accuracy in these approaches. As the results show, GPINN achieves the highest classification accuracy on both data sets.

V. Conclusion

This study proposes a Gaussian process-based self-organizing incremental neural network (GPINN) to address the online probability density estimation problem. Instead of using distance measures, GPINN adopts Gaussian process models with adaptive kernels. This allows to calculate winning nodes and threshold regions by mapping the distribution of the neighbors of each node to its link relationship. The bandwidth matrix updating rule is derived, and thus a novel online kernel density estimator is presented. Moreover, we theoretically analyze the threshold parameter optimization problem and the advantages of the proposed approach. We find that 1) the selection of winning nodes is robust to inappropriate edges, and 2) the calculation of threshold regions can effectively solve the shortcomings of adopting distance measures. The experimental results on both synthetic data sets and real-world data sets demonstrate that GPINN achieves remarkable improvement in density estimation accuracy for large noisy data.

Acknowledgment

X.W. and J.I. acknowledge support from the Japan Society of the Promotion of Science (JSPS) Grant-in-Aid for Scientific Research (A) 26249062.

References

[1] Dahua Lin, Online Learning of Nonparametric Mixture Models via Sequential Variational Approximation, *Conference on Neural Information Processing Systems (NIPS)*, pp.395-404, 2013.

[2] M. C. Hughes and E. B. Sudderth, Memorized Online Variational Inference for Dirichlet Process Mixture Models, *Conference on Neural Information Processing Systems (NIPS)*, pp.1133-1141, 2013.

[3] M. Kristan and A. Leonardis and D. Skocaj, Multivariate Online kernel density estimation with Gaussian kernels, *Pattern Recognition*, vol.44, pp.2630-2642, 2011.

[4] David Hastie, Silvia Liverani and Sylvia Richardson, Sampling from Dirichlet process mixture models with unknown concentration parameter: mixing issues in large data implementations, *Stat Comput*, vol.25, pp.1023-1037, 2015.

[5] A. Jasra, C.C. Holmes and D.A. Stephens, Markov Chain Monte Carlo Methods and the Label Switching Problem in Bayesian Mixture Modeling, *Statistical Science*, vol.20, pp.50–67, 2005.

[6] Aurore Lavigne and Silvia Liverani, Uncertainty quantification of a partition coming from a DPMM, 2018.

[7] P. Vincent, and Y. Bengio, Manifold Parzen Windows, *Neural Information Processing Systems*, pp.825-832, 2002.

[8] Y. Bengio, O. Delalleau, and N. Le Roux, N. The curse of dimensionality for local kernel machines. Technical Report 1258, Département d'informatique et de recherche opérationnelle, Université de Montréal, 2005.

[9] Timo Simila, Self-organizing Map Learning Nonlinearly Embedded Manifolds, *Information Visualization*, vol.4, pp.22-31, 2005.

[10] Hujun Yin, Data visualisation and manifold mapping using the ViSOM, *Neural Networks*, vol.15, pp.1005-1016, 2002.

[11] Hujun Yin, Learning Nonlinear Principal Manifolds by Self-Organising Maps, *Lecture Notes in Computational Science and Engineering*, pp.68-95, 2007.

[12] N. J. Mount and D. Weaver, Self-organizing maps and boundary effects: quantifying the benefits of torus wrapping for mapping SOM trajectories, *Pattern Analysis and Applications*, vol.14, pp.139-148, 2011.

[13] F. Shen, Q. Ouyang, W. Kasai, and O. Hasegawa, A general associative memory based on self-organizing incremental neural network, *Neurocomputing*, vol.104, pp.57-71, 2013.

[14] F. Shen, S. Akihito, and O. Hasegawa, An enhanced self-organizing incremental neural network for online unsupervised learning, *Neural Networks*, vol.20, pp.893-903, 2007.

[15] F. Shen, H. Yu, W. Kasai, and O. Hasegawa, An associative memory system for incremental learning and temporal sequence, *Int. Joint Conf. Neural Networks*, pp.1-8, 2010.

[16] Y. Nakamura, and O. Hasegawa, Nonparametric density estimation based on self-organizing incremental neural network for large noisy data, *IEEE Trans. Neural Networks and Learning Systems*, pp.25-32, 2016.

[17] B. Scholkopf and A. J. Smola, Learning with Kernels, MIT Press, Cambridge, MA, USA, 2002.

[18] Duda, Richard O. and Hart, Peter E. and Stork, David G., Pattern Classification (2nd Edition), Wiley-Interscience, New York, NY, USA, 2000.

[19] M. C. Jones and J. S. Marron and S. J. Sheather, A Brief Survey of Bandwidth Selection for Density Estimation. *Journal of the American Statistical Association*, vol.91, pp.401–407, 1996.

[20] C. E. Rasmussen and C. K. L. Williams, Gaussian Processes for Machine Learning, MIT Press, Cambridge, MA, USA, 2006.

[21] C. K. L. Williams and C.E. Rasmussen, Gaussian processes for regression. *Conference on Neural Information Processing Systems (NIPS)*, pp. 514-520. 1996.

[22] C. K. L. Williams and D. Barber, Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.20, pp.1342-1351, 1998.

[23] George R. Terrell and David W. Scott, Variable Kernel Density Estimation, *The Annals of Statistics*, vol.20, pp.1236-1265, 1992.

[24] Bruce E. Hansen, Uniform Convergence Rates for Kernel Estimation with Dependent Data, *Econometric Theory*, vol.24, pp.726-748, 2008.

[25] A. Bhattacharya, D. Pati, and D. Dunson, Anisotropic function estimation using multi-bandwidth Gaussian processes, *Ann. Stat.*, vol.42, pp.352-381, 2014.

[26] Aad van der Vaart, and Harry van Zanten, Information rates of nonparametric Gaussian process methods, *Journal of Machine Learning Research*, vol.12, pp.2095-2119, 2011.

[27] Sylvain Arlot and Alain Celisse, A survey of cross-validation procedures for model selection. *Statist. Surv.*, vol.4, pp.40–79, 2010.

[28] Tarn Duong and Martin L. Hazelton, Cross-validation Bandwidth Matrices for Multivariate Kernel Density Estimation, *Scandinavian Journal of Statistics*, vol.32, pp.485-506, 2005.

[29] J. B. Tenenbaum, V. de Silva, and J. C. Langford, A global geometric framework for nonlinear dimensionality reduction, *Science*, vol. 5500, pp. 2319–2323, Dec. 2000.

[30] Mikhail Belkin and Partha Niyogi, Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering, *Advances in Neural Information Processing Systems*, pp. 586–691, 2001.

[31] S. T. Roweis and L. K. Saul, Report nonlinear dimensionality reduction by locally linear embedding, *Science*, vol. 290, pp. 2323-2326, Dec. 2000.