IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 29, 2019, accepted May 28, 2019, date of publication June 10, 2019, date of current version June 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2921832

Topological Clustering via Adaptive Resonance Theory With Information Theoretic Learning

NAOKI MASUYAMA[1], (Member, IEEE), CHU KIONG LOO[2], (Senior Member, IEEE), HISAO ISHIBUCHI[3], (Fellow, IEEE), NAOYUKI KUBOTA[4], (Member, IEEE), YUSUKE NOJIMA[1], (Member, IEEE), AND YIPING LIU[1], (Member, IEEE)

[1]Graduate School of Engineering, Osaka Prefecture University, Osaka 599-8531, Japan
[2]Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur 50603, Malaysia
[3]Shenzhen Key Laboratory of Computational Intelligence, University Key Laboratory of Evolving Intelligent Systems of Guangdong Province, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China
[4]Graduate School of Systems Design, Tokyo Metropolitan University, Tokyo 191-0065, Japan

Corresponding author: Hisao Ishibuchi (hisao@sustech.edu.cn)

This work was supported in part by the Ministry of Education, Culture, Sports, Science and Technology–Japan (MEXT) Leading Initiative for Excellent Young Researchers (LEADER), in part by the Frontier Research Grant under Project FG003-17AFR from University of Malaya, in part by the ONRG Grant under Project ONRG-NICOP-N62909-18-1-2086 from Office of Naval Research Global, U.K., in part by the International Collaboration Fund under Project IF0318M1006 from MESTECC, Malaysia, in part by the National Natural Science Foundation of China under Grant 61876075, in part by the Program for Guangdong Introducing Innovative and Enterpreneurial Teams under Grant 2017ZT07X386, in part by the Shenzhen Peacock Plan under Grant KQTD2016112514355531, in part by the Science and Technology Innovation Committee Foundation of Shenzhen under Grant ZDSYS201703031748284, and in part by the Program for University Key Laboratory of Guangdong Province under Grant 2017KSYS008.

ABSTRACT This paper proposes a topological clustering algorithm by integrating topological structure and information theoretic learning, i.e., correntropy, into adaptive resonance theory (ART). Specifically, the proposed algorithm utilizes the correntropy induced metric (CIM) for defining a similarity measure, a node insertion criterion, and an edge creation criterion. Other types of the ART-based topological clustering algorithms have been developed, however, these algorithms have various drawbacks such as a large number of parameters, sensitivity to noisy data. Moreover, generated topological networks cannot represent the distribution of data. In contrast, the proposed algorithm realizes a stable computation and reduces the number of parameters compared to existing algorithms. Furthermore, improving the ability to express the data structure more appropriately by the topological network, a mechanism that adaptively controls the node insertion criterion is introduced to the proposed algorithm. The experimental results showed that the proposed algorithm has superior performance with respect to the self-organizing and the classification abilities compared with the state-of-the-art topological clustering algorithms.

INDEX TERMS Adaptive resonance theory, correntropy, information theoretic learning, topological clustering.

I. INTRODUCTION

With the growth of Internet of Things (IoT) technologies, a massive amount of information (i.e., big data) can be obtained easily from various sources. However, the obtained data usually contain not only valuable information but also irrelevant one. Thus, extracting useful information from data is a significant task.

In general, cluster analysis is one of the widely applied approaches to extract knowledge or hidden relation from data. The k-means [1] and Expectation-Maximization (EM) [2] are well-known unsupervised clustering algorithms. Although the k-means and EM need to specify the number of clusters of data in advance, they have been widely used in various studies because of their simplicity and high applicability.

Self-Organizing Map (SOM) [3], Growing Cell Structure (GCS) [4], and Growing Neural Gas (GNG) [5] are fundamental algorithms of topological clustering. A representative topological clustering algorithm is Self-Organizing Incremental Neural Network (SOINN) [6] which is successfully integrated the features of SOM and GNG. These algorithms, however, suffer from "plasticity-stability dilemma" [7]. In order to avoid the dilemma, Adaptive Resonance Theory (ART) [8], which is inspired by the learning mechanism of

The associate editor coordinating the review of this manuscript and approving it for publication was Bo Shen.

the human brain, has been developed as an unsupervised clustering algorithm. Fuzzy ART (FA) [9] is regarded as a basic version of ART-based clustering. TopoART [10] is an on-line self-organizing clustering algorithm which combines two FAs in a hierarchical manner. One superior ART-based algorithm is Kernel Bayesian ART (KBA) [11] which applies Kernel Bayes' Rule (KBR) [12] and Correntropy Induced Metric (CIM) [13] to realize the fast and stable self-organizing ability.

In regard to self-organizing clustering algorithms, SOINN and TopoART suffer from instability that the quality of their clustering results is highly dependent on the presentation order of input data points. One of the state-of-the-art algorithms that can deal with the above problem is Topological Kernel Bayesian ART (TKBA) [14]. TKBA is an ART-based topological clustering algorithm which inherits the advantages of KBA. According to [14], TKBA achieves more stable self-organizing ability and higher noise reduction ability than TopoART- and SOINN-based algorithms. The superior performance of TKBA is based on the characteristics of KBR and CIM. In TKBA, drawbacks caused by KBR and CIM also exist. The major drawbacks of TKBA are two-fold: (i) the number of parameters in TKBA is large, and one parameters (i.e., a kernel bandwidth σ) has a strong effect on its clustering performance, and (ii) since the CIM-based node insertion criterion is fixed during a cluster learning process, the generated nodes tend to be uniformly distributed. The former problem means the difficulty of appropriate parameter specification, which reduces the applicability of the algorithm. The latter problem means the difficulty of generating a topological network which appropriately reflects the characteristics of the data such as density information.

In summary of the existing algorithms, classical algorithms such as k-means, EM, and SOM require knowledge of data in advance. Since GNG and SOINN suffer from the plasticity-stability dilemma, it is difficult to perform stable continuous learning required for clustering algorithms. TKBA successfully overcomes the problems of GNG and SOINN. However, TKBA has several parameters that greatly affect its clustering ability. In addition, it is difficult for the topological network generated by TKBA to appropriately represent the features of data such as density information.

The main purpose of this paper is to tackle the above-mentioned problems of the existing clustering algorithms. Specifically, we propose a Topological CIM-based ART (TCA) which is inspired by TKBA, since TKBA achieves the state-of-the-art performance among the topological clustering algorithms. In TKBA, CIM is used for the similarity measurement whereas KBR is used for the nearest neighbor node selection. However, in the proposed TCA, CIM is used for both of them. As a result, the reduction in the number of parameters is achieved. Furthermore, the node insertion criterion (i.e., vigilance parameter) of an individual node is adaptively regulated based on the distribution of data points. Consequently, the insertion of nodes is suppressed in a sparse data region and more nodes are inserted in a dense data region, so that the topological network generated by TCA can represent the distributions of data points appropriately. TCA has less parameters and higher self-organizing ability (to represent the distribution of data points) compared to the state-of-the-art GNG-based and ART-based topological clustering algorithms.

The paper is organized as follows: Section II presents a literature review for some representative unsupervised clustering algorithms. Section III describes details of the proposed TCA algorithm. Section IV presents simulation experiments to examine the self-organizing ability of TCA. Concluding remarks are presented in Section V.

II. LITERATURE REVIEW

Supervised learning algorithms have superior classification performance such as support vector machine [15] and deep learning [16]–[18]. In general, however, supervised learning algorithms require well-structured labeled data to maximize their classification ability. As mentioned earlier, different types of huge information are generated at any moment based on IoT technologies. In this situation, it is difficult to prepare well-structured labeled data for each learning task.

On the other hand, unsupervised learning algorithms do not require such well-structured labeled information during learning. Especially, it is considered that the significance of cluster analysis will further increase in the growing IoT society, where huge data without structured labels are generated day by day [19]. Therefore, the importance of unsupervised learning algorithms become increasing, and related research has become more active both in theoretical and practical perspectives [20], [21].

Typical types of unsupervised clustering algorithms are the EM [2] and the k-means [1]. The SOM [3] is a classical topological clustering algorithm which is mainly utilized for data visualization. However, the k-means and EM generate only a predefined number of clusters even if its number of clusters is not appropriate. Moreover, SOM tries to generate multiple separated clusters by a predefined single topological network architecture. GCS [4] and GNG [5] have successfully solved the problem of SOM by introducing a growing topological network architecture. Due to superior performance of the growing topological network architecture, GNG-based topological clustering algorithms have been integrated with other learning algorithms such as semi-supervised learning [22] and hierarchical clustering [23]. One well-known problem of GNG is the excessive cluster creation process that inserts a new node to the region with the maximum error until a predefined maximum number of nodes are generated. In order to handle this problem, Grow When Required (GWR) [24] shows an effective solution. GWR inserts nodes whenever the state of the current network does not sufficiently match the input data. Another effective approach is integrating GNG and CIM [25], which is called GNG-CIM [26]. CIM is correntropy-based similarity measurement which is proposed from the information theoretic learning perspective. CIM can be considered as an alternative criterion realizing more stable

computation than the Euclidean distance-based similarity measurement.

SOINN [6] is one of the representative topological clustering algorithms which successfully integrates the features of SOM and GNG. SOINN can grow a topological network incrementally based on the Euclidean distance-based similarity measurement and an error-based node insertion criterion. Several types of SOINN-based algorithms have also been proposed in [27]–[29]. Enhanced SOINN (ESOINN) [27] simplifies the structure of SOINN to a single-layer model and reduces the number of parameters to be specified. In addition, ESOINN has the ability to separate overlapping clusters based on their node distributions. Adjusted SOINN (ASOINN) [28] further reduces the number of parameters from SOINN, however, the overlap separation process in ESOINN has omitted. To estimate density information by ASOINN, SOINN combined with Kernel Density Estimation (KDE) [29] (KDESOINN) is proposed in [30]. Although these algorithms are computationally efficient, the learned network distributions for high-dimensional data are unstable because of their Euclidean distance-based network construction mechanisms [31].

The topological clustering algorithms have successfully realized the ability to represent new knowledge into their topological structure. However, since these algorithms permanently insert new nodes into their network for memorizing new knowledge, they have a potential to forget learned knowledge (i.e., catastrophic forgetting). This trade-off is called the plasticity-stability dilemma. A typical successful approach to solve the plasticity-stability dilemma is ART [8] inspired by the human brain. FA [9] is considered to be a representative of ART-based clustering algorithms. To improve the self-organizing ability of FA, TopoART [10] has been developed. TopoART is an ART-based self-organizing incremental clustering algorithm which hierarchically combines two FAs (i.e., TopoARTa and TopoARTb). TopoARTa performs clustering by FA using all data points, while TopoARTb generates clusters using only the data points contributing to the cluster generation in TopoARTa. TopoART has the same advantages as FA which enables the match-based fast stable learning and intrinsic self-organization. Although TopoART has various advantages, it also has a major problem associated with the FA learning process, i.e., high sensitivity to statistical overlapping between the generated categories [32]. This sensitivity problem results in category proliferation (i.e., disordered generation of categories), which leads to a high computational cost, and deterioration in clustering ability.

To tackle the category proliferation problem, Bayesian ART (BA) [33] has been proposed. BA successfully integrates Bayes' theorem to the ART architecture which selects the nearest neighbor node for a data point. However, BA applies a covariance-based similarity measurement when updating the status of clusters (i.e., vigilance test). Thus, if BA attempts to process a large number of data points in a high-dimensional feature space, it is difficult to achieve (i) stable likelihood calculation and (ii) high global convergence performance within feasible computational time [34]. Kernel Bayesian ART (KBA) [11] can handle the problems of BA by applying Kernel Bayes' Rule (KBR) [12] for selecting the nearest neighbor node instead of the general Bayes' theorem in BA, and a correntropy [25]-based alternative similarity measurement called CIM [13]. As an extension of KBA, Topological Kernel Bayesian ART (TKBA) [14] has been proposed. In TKBA, KBR is utilized for selecting the nearest neighbor node and CIM is utilized for the node insertion criterion. Moreover, CIM is also utilized as a criterion to construct topological networks, which contributes to their stability. Even though there are two major problems, namely the number of parameters is large and the generated nodes tend to be uniformly distributed, TKBA is a superior topological clustering algorithm compared to SOINN, KDESOINN, and TopoART from the viewpoint of the self-organizing ability.

III. TOPOLOGICAL CIM-BASED ART

In this section, first the theoretical background of CIM is briefly described, then the learning algorithm of TCA is presented in detail.

A. CORRENTROPY-INDUCED METRIC

1) DEFINITION

Correntropy [25], which is a generalized similarity measure between two variables, is defined as follows:

$$C_\sigma(\boldsymbol{X}, \boldsymbol{Y}) = E[\kappa_\sigma(X - Y)], \tag{1}$$

where $\boldsymbol{X} = (x_1, x_2, \ldots, x_L)$ $(x_l \in \Re^d)$ and $\boldsymbol{Y} = (y_1, y_2, \ldots, y_K)$ $(y_k \in \Re^d)$ are arbitrary vectors. κ_σ is a kernel function that satisfies the Mercer's Theorem [15].

A nonlinear metric called CIM is derived from correntropy. CIM quantifies the similarity between two probability distributions as follows:

$$\mathrm{CIM}(x_i, y_k, \sigma) = [\{\kappa_\sigma(0) - \kappa_\sigma(x_i - y_k)\}]^{\frac{1}{2}}. \tag{2}$$

Here, the kernel bandwidth σ affects the sensitivity of CIM, which is discussed in Section IV-B.

2) KERNEL FUNCTION IN CIM

In general, correntropy frequently applies the Gaussian kernel, which is desirable due to its smoothness and strict positive-definiteness. Thanks to the features of the Gaussian kernel, CIM is desirably defined as a nonlinear metric by correntropy. CIM performs like an L_2 norm if data points are densely distributed compared to the kernel bandwidth sigma, like an L_1 norm if data points are widely distributed, and like an L_0 norm when data points are far away from the $\kappa_\sigma(0)$ in (2) [35]. In this study, because of the features of the Gaussian kernel function, it is utilized as the kernel function κ_σ in (2), i.e., $\exp(-\|x - y\|^2/2\sigma^2)$.

B. LEARNING PROCEDURE OF TCA

The learning procedure of TCA is divided into five parts, namely (i) winner node selection, (ii) vigilance test, (iii) node

learning, (iv) topology construction, and (v) kernel bandwidth adaptation. In the following, let us suppose data points $\boldsymbol{X} = (x_1, x_2, \ldots, x_L)$ ($x_l \in \Re^d$) are given to the TCA network which generates nodes $\boldsymbol{Y} = (y_1, y_2, \ldots, y_K)$ ($y_k \in \Re^d$). Furthermore, each node in $\boldsymbol{Y}$ has an individual kernel bandwidth $S = (\sigma_1, \sigma_2, \ldots, \sigma_K)$ for CIM.

In the beginning of the learning where the TCA has no node, x_1 becomes a new node, i.e., $y_1 = x_1$, which has a predefined arbitrary kernel bandwidth σ_{init}.

1) WINNER NODE SELECTION

Once data point x_l is input to the network, a node which has the similar state to data point x_l is selected, i.e., a winner node. As one option to reduce the computational cost, candidates of the winner node can be extracted from nodes $\boldsymbol{Y}$. Specifically, CIM between x_l and $\boldsymbol{Y}$ is calculated, and nodes having smaller values than the mean of kernel bandwidths (i.e., mean(S)) of $\boldsymbol{Y}$ are extracted as $\boldsymbol{Y}_{\text{extracted}}$:

$$\boldsymbol{Y}_{\text{extracted}} = \left\{ y_k \,|\, \underset{k \in K}{\text{CIM}}(x_l, y_k, \text{mean}(S)) \leq \text{mean}(S) \right\}. \quad (3)$$

As a result, only the neighbor nodes in $\boldsymbol{Y}_{\text{extracted}}$ around x_l are considered for the following learning process. Hereafter, for simplicity, the extracted nodes $\boldsymbol{Y}_{\text{extracted}}$ are referred to as $\boldsymbol{Y}$.

The index k of winner node for x_l is selected from $\boldsymbol{Y}$ based on CIM which is calculated by (2) as follows:

$$k = \arg\min_{k \in K} \left[\text{CIM}\left(x_l, \boldsymbol{Y}, S\right) \right]. \quad (4)$$

2) VIGILANCE TEST

After determined the winning node, a vigilance test is performed. The vigilance test evaluates whether x_l belongs to y_k, which is defined as follows:

$$V_k \leq \text{CIM}\left(x_l, y_k, \max(S)\right), \quad (5)$$

where V_k denotes the similarity between x_l and the winner node y_k, which is defined by CIM as follows:

$$V_k = \kappa_{\sigma_k}(0) - \kappa_{\sigma_k}\left(\|x_l - y_k\|\right), \quad (6)$$

where κ denotes a kernel function, and σ_k denotes an individual kernel bandwidth of y_k. In this study, the Gaussian kernel is utilized for CIM.

In case that the condition in (5) is not satisfied by y_k, searching for the next candidate of the winning node. If the vigilance test fails with all the existing nodes $\boldsymbol{Y}$, a new node is defined as $y_{K+1} = x_l$.

The kernel bandwidth of the new node y_{K+1} should have a similar state to its neighbors. In this study, therefore, the average of the kernel bandwidths of the three neighbor nodes is utilized as σ_{K+1} for the new node y_{K+1}.

3) NODE LEARNING

Once the vigilance test is satisfied, the state of the winner node y_k is updated as follows:

$$y_k = \frac{M_k}{M_k + 1} y_k + \frac{1}{M_k + 1} x_l, \quad (7)$$

where M_k denotes the number of data points that have accumulated by the node y_k.

In addition, the number of data points that have accumulated by the node y_k is incremented as follows:

$$M_k \leftarrow M_k + 1, \quad (8)$$

$$N_k \leftarrow N_k + 1. \quad (9)$$

M_k is utilized for the node learning described in Section III-B.3. N_k is utilized for the kernel bandwidth adaptation process described in Section III-B.5. M_k increases continuously whereas N_k is initialized to zero during a kernel bandwidth adaptation process, thus we need the two counters M_k and N_k.

4) TOPOLOGY CONSTRUCTION

In general, the ideal topological network satisfies a Delaunay diagram. However, similar to TKBA [14], a topology construction process in TCA does not guarantee to generate the Delaunay diagram. The topology construction process consists of node deletion and edge creation/deletion as follows:

a: EDGE CREATION

Once the vigilance test succeeds and the 2nd winner node also satisfies the match criterion in (5), the 1st and 2nd winner nodes are connected by an edge. The 1st and 2nd winner nodes are determined by (4) during the winner node selection. Unlike GNG, the edges in TCA do not have any age factor.

For a stable topology construction, node deletion and edge deletion processes are performed by a predefined cycle λ (i.e., every λ iterations).

b: NODE DELETION

As a node deletion criterion, the similarity between y_k which satisfies the vigilance test (i.e., $V_k \leq \text{mean}(S)$) and x_l is defined by CIM as an error E_{CIM} ([0, 1]) corresponding to y_k. For the error E_{CIM}, the initial value of the error $E_{\text{CIM}} = 1$ is given to a newly generated node, and E_{CIM} becomes zero when $V_k = 0$. The error E_{CIM} is updated by the following rule during the vigilance test:

$$E_{\text{CIM}}^{y_k} = \min\left(E_{\text{CIM}}^{y_k}, V_k\right), \quad (10)$$

where V_k is calculated by CIM.

If the error E_{CIM} is large, it means that there is no data point near the node. In TCA, the node deletion is executed if the node has an error E_{CIM} larger than the square of mean(S), namely:

$$E_{\text{CIM}} > \text{mean}(S)^2. \quad (11)$$

Here, all nodes that satisfy the above condition are deleted from $\boldsymbol{Y}$. In addition, isolated nodes, which do not have an emanating edge, are also removed from $\boldsymbol{Y}$.

c: EDGE DELETION

The TCA does not have an age factor for each edge, thus, the edge deletion is performed only when an edge intersection is detected. Intersections are detected by the cross

product-based detection algorithm [36], which is applied to all the nodes that have emanating edges. If an intersection is detected, the edge which has the maximum CIM is removed.

5) KERNEL BANDWIDTH ADAPTATION

The kernel bandwidth σ of CIM in TCA is adaptively changed based on a frequency of data point input, i.e., the nodes in a region with frequent input tend to have a small kernel bandwidth value, and the nodes in a region with infrequent input tend to have a large kernel bandwidth value. Here, the "*region*" is defined by nodes connected by edges, that is, a cluster. This means that the kernel bandwidth value of the node is separately adjusted within each cluster. As a result, it is possible to adjust the kernel bandwidth value appropriately taking into consideration the individual local distribution of data points corresponding to each cluster.

Let us suppose that a set of clusters $Z = (z_1, z_2, \ldots, z_R)$ has been generated in TCA, and each cluster z_r consists of connected nodes $y_k \in \boldsymbol{Y}$. The adaptation process is executed when a new node (i.e., y_{K+1}) has not been generated in $\boldsymbol{Y}$ during a period $J = \arg\min_{k \in K}[N_k] \times 10$, where N_k is the number of data points that have been accumulated by $y_k \in \boldsymbol{Y}$ as defined in (9). In the following, the adaptation process for $y_k \in z_r$, which hold the kernel bandwidth σ_k and the counter N_k, is considered.

First of all, the uniform degree α_{z_r} of the cluster z_r is calculated based on the counters N of each node in the cluster z_r as follows:

$$\alpha_{z_r} = \frac{\text{median}\,(N)}{\text{median}\,(N) + \text{mean}\,(N)}. \tag{12}$$

Here, $\alpha_{z_r} = 0.5$ means that the data points are normally distributed. In other words, the nodes in the cluster z_r should also be normally distributed.

The kernel bandwidth σ_k of $y_k \in z_r$ is updated as follows:

[Case 1] When $0.05 < (0.5 - \alpha_{z_r})$,

(i) $N_k \geq \text{mean}(N)$

$$\sigma_k = \begin{cases} \sigma_k - \dfrac{1}{10}\|\alpha_{z_r}\| \cdot \text{mean}(N), & \text{if } \sigma_k \leq 0.5 \cdot \sigma_{\text{init}} \\ 0.5 \cdot \sigma_{\text{init}}, & \text{else.} \end{cases} \tag{13}$$

(ii) $N_k < \text{mean}(N)$

$$\sigma_k = \begin{cases} \sigma_k + \dfrac{1}{10}\|\alpha_{z_r}\| \cdot \text{mean}(N), & \text{if } \sigma_k \geq 1.5 \cdot \sigma_{\text{init}} \\ 1.5 \cdot \sigma_{\text{init}}, & \text{else.} \end{cases} \tag{14}$$

[Case 2] When $(0.5 - \alpha_{z_r}) < -0.05$,

(i) $N_k \geq \text{mean}(N)$

$$\sigma_k = \begin{cases} \sigma_k + \dfrac{1}{10}\|\alpha_{z_r}\| \cdot \text{mean}(N), & \text{if } \sigma_k \geq 1.5 \cdot \sigma_{\text{init}} \\ 1.5 \cdot \sigma_{\text{init}}, & \text{else.} \end{cases} \tag{15}$$

(ii) $N_k < \text{mean}(N)$

$$\sigma_k = \begin{cases} \sigma_k - \dfrac{1}{10}\|\alpha_{z_r}\| \cdot \text{mean}(N_k), & \text{if } \sigma_k \leq 0.5 \cdot \sigma_{\text{init}} \\ 0.5 \cdot \sigma_{\text{init}}, & \text{else.} \end{cases} \tag{16}$$

Here, σ_{init} denotes a predefined arbitrary kernel bandwidth value for the first generated node. Once the above adaptation has occurred, the counters N of each node in the cluster z_r are initialized to zero.

In order to avoid excessive contraction and expansion of the kernel bandwidth, upper and lower limits are defined. Although there are several conditional branches, the updating rules of the kernel bandwidth are quite simple. The summary of the learning procedure of TCA is presented in Algorithm 1.

As described above, the learning algorithm of TCA is divided into five parts, namely (i) winner node selection, (ii) vigilance test, (iii) node learning, (iv) topology construction, and (v) kernel bandwidth adaptation, which are indicated in line 7, line 9, lines 10-12, lines 21-27, and lines 28-33 of Algorithm 1, respectively.

C. COMPUTATIONAL COMPLEXITY

The complexity of an algorithm is one of the significant factors to compare learning algorithms. In general, the complexity of an algorithm can generally be evaluated from two aspects, i.e., (i) the computational complexity which shows how long the algorithm is executed, and (ii) the space complexity which shows how much memory is used by the algorithm. In this study, we consider the computational complexity which is typically represented by a Big-$\mathcal{O}$ notation. In the following, the symbols N, D, C, I, and L denote the number of data points, the dimension of the data points, the number of nodes, the number of iterations, and the size of batch data points, respectively.

In [14], we discussed the computational complexity of TKBA, ASOINN, and TopoART as $\mathcal{O}(NC^2I) + \mathcal{O}(NDCI)$, $\mathcal{O}(NC^2I)$, and $\mathcal{O}(NDC^3I) + \mathcal{O}(ND^2C^2I)$, respectively. In TKBA, $\mathcal{O}(NC^2I)$ is needed for the winner node selection process, and $\mathcal{O}(NDCI)$ is needed for the recursive calculation in the vigilance test process.

The learning procedure of TCA is similar to TKBA, i.e., $\mathcal{O}(NC^2I)$ is needed for the winner node selection process, and $\mathcal{O}(NDCI)$ is needed for the recursive calculation in the vigilance test process. The complexity of the kernel bandwidth adaptation process is lower than the winner node selection and the recursive calculation processes. As a result, the computational complexity of TCA is derived as $\mathcal{O}(NC^2I) + \mathcal{O}(NDCI)$, which is the same as TKBA.

The computational complexity of each algorithm is summarized in Table 1.

IV. SIMULATION EXPERIMENTS

This section presents simulation experiments to evaluate the ability of TCA. First, the effectiveness of the kernel bandwidth adaptation is demonstrated. Next, the effect of

Algorithm 1 Learning Algorithm of TCA

Input:
the data points: $\boldsymbol{X} = (x_1, x_2, \ldots, x_L)$ $(x_l \in \Re^d)$,
the existing nodes: $\boldsymbol{Y} = (y_1, y_2, \ldots, y_K)$ $(y_k \in \Re^d, K \geq 0)$,
the kernel bandwidth of $\boldsymbol{Y}$: $S = (\sigma_1, \sigma_2, \ldots, \sigma_K)$,
and the number of data points that have accumulated by the node y_k: $M = (M_1, M_2, \ldots, M_K)$, and $N = (N_1, N_2, \ldots, N_K)$.
Output:
the updated nodes: $\boldsymbol{Y}$
the updated kernel bandwidth of $\boldsymbol{Y}$: S,
and the updated number of data points that have accumulated by the node y_k: M, and N.

1 **function** LearnTCA($\boldsymbol{X}$, $\boldsymbol{Y}$, S, M, N)
2 Input a data point x_l to network.
3 **if** *There is no node in TCA network* **then**
4 Create the new node as $y_{K+1} = x_l$.
5 Assign the kernel bandwidth $\sigma_{K+1} = \sigma_{\text{init}}$ for y_{K+1}.
6 **else**
7 Compute the 1st and 2nd winner nodes by (4).
8 Compute the similarity V_k of the winner node as (6).
9 **if** *The 1st winner node satisfies* $V_k \leq mean(S)$ *as (5)* **then**
10 Update the state of y_k by (7).
11 Update counters M_k and N_k by (8) and (9).
12 Update the error $E_{\text{CIM}}^{y_k}$ by (10).
13 **if** *The 2nd winner node satisfies* $V_k \leq mean(S)$ *as (5)* **then**
14 Create a new edge between 1st and 2nd winner nodes.
15 **else**
16 **if** *All the nodes are failed with the vigilance test* **then**
17 Create the new node as $y_{K+1} = x_l$.
18 Assigne the average of the kernel bandwidth of the three neighbor nodes to σ_{K+1} for y_{K+1}.
19 **else**
20 Remove the 1st winner node y_k from the winner node selection, and continue from step 9 with the next candidate node.
21 **if** *The instant* l *is multiple of a topology construction cycle* λ **then**
22 **forall the** $k \in 1, \ldots, K$ **do**
23 **if** y_k *does not have an emanating edge* **then**
24 Remove y_k from $\boldsymbol{Y}$.
25 **if** y_k *satisfies* $E_{CIM}^{y_k} > mean(S)^2$ *as (11)* **then**
26 Remove y_k from $\boldsymbol{Y}$.
27 Remove the longest edge from y_k which makes an intersection.
28 **if** *A new node has not been generated during a period* $J = \arg\min_{k \in K}[N_k] \times 10$ **then**
29 Compute the uniform degree α_{z_r} by (12).
30 **if** *Satisfy* $0.05 < (0.5 - \alpha_{z_r})$ **then**
31 Update the kernel bandwidth σ_k by (13) and (14).
32 **else if** *Satisfy* $(0.5 - \alpha_{z_r}) < -0.05$ **then**
33 Update the kernel bandwidth σ_k by (15) and (16).
34 **if** $l < L$ **then**
35 Continue from step 2 with $l \leftarrow l + 1$.
36 **return** $\boldsymbol{Y}$, S, M, *and* N.

parameters in TCA is investigated. Then, the self-organizing and classification abilities are evaluated in comparison with TKBA [14], ASOINN [28], and TopoART [10].

A. ADAPTATION OF KERNEL BANDWIDTH IN CIM

To visualize the effectiveness of the kernel bandwidth adaptation, a 2D synthetic dataset is utilized. The dataset consists

TABLE 1. Comparison of computational complexity. *N*, *D*, *C*, *I*, and *L* denote the number of data points, the dimension of data points, the number of clusters, the number of iterations, and the size of batches, respectively.

Model	Order	Defined in
TCA	$\mathcal{O}(NC^2I) + \mathcal{O}(NDCI)$	–
TKBA	$\mathcal{O}(NC^2I) + \mathcal{O}(NDCI)$	[14]
ASOINN	$\mathcal{O}(NC^2I)$	[14]
TopoART	$\mathcal{O}(NDC^3I) + \mathcal{O}(ND^2C^2I)$	[14]

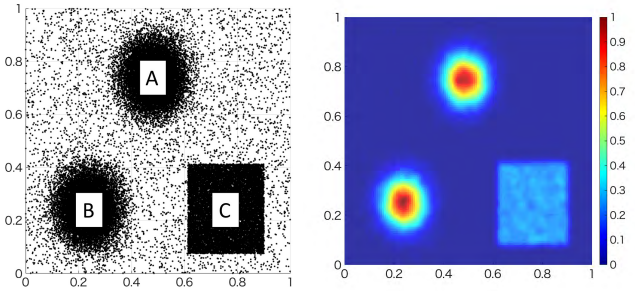

FIGURE 1. 2D synthetic dataset with 10% uniform random noise for kernel bandwidth adaptation. The dataset is divided into 3 parts as two Gaussian distributions (A and B), and a rectangle uniform distribution (C). Each distribution consists of 15k data points without noise. (a): entire data points, and (b): density estimation results utilizing data points by kernel density estimation.

of 45k data points which are generated from two Gaussian distributions (15k data points in each distribution), and a rectangle uniform distribution (15k data points). Furthermore, 10% random uniform noise is added to each distribution (i.e., 1.5k data points). Fig. 1a shows the entire data points. Fig. 1b shows a density estimation result from the data points in Fig. 1a by Kernel Density Estimation (KDE) [37].

During experiment, the entire data points are given to the network five times. In addition, our simulation experiments are conducted in two environments, i.e., stationary and non-stationary environments. In the stationary environment, the data points are randomly selected from the entire dataset (49.5k data points). In the non-stationary environment, the data points are randomly selected from a specific distribution with 10% noise (16.5k data points), and the distribution is sequentially shifted from A to C. In this section,

TABLE 2. Parameter settings for kernel bandwidth adaptation experiment.

Algorithm	Parameter	Value
TCA	topology construction cycle λ	400
	initial kernel bandwidth for CIM σ_{init}	0.07
TKBA	topology construction cycle λ	400
	hypervolume V_{MAX}	0.25
	kernel bandwidth for CIM σ_{CIM}	0.035
	kernel bandwidth for KBR σ_{kbr}	1.0
	Tikhonov regularization constant ϵ_K	$0.01/K$
	Tikhonov regularization constant δ_K	$2K$
	weighting factor γ	$1/d$
ASOINN	node insert cycle λ	400
	maximum age of edge age_{MAX}	25
	parameter for node deletion c	0.5
TopoART	node insert cycle τ	(300, 300)
(a, b)	learning rate β	(1.00, 0.65)
	parameter for node deletion ϕ	(2, 2)
	vigilance parameter ρ	(0.90, 0.95)

the parameters of each algorithm are set as in Table 2. The parameters are selected to generate a similar number of nodes in the network of each algorithm.

Figures 2 and 3 show self-organizing results, and Figs. 4 and 5 show density estimation results generated by KDE [37] utilizing nodes in the obtained topological network of each algorithm. In Figs. 2 and 3, a circle (or a rectangle) plot represents a node and a black line denotes an edge of the topological network.

Focusing on Figs. 2 and 3, it is clearly observed that TopoART is sensitive to the random noise data points. In addition, the generated topological network in ASOINN is quite different in the stationary and the non-stationary environments. Since ASOINN tends to generate a different network structure depending on the environment of the input sequence, a reproducibility of topological network formation is low. In contrast, TCA and TKBA can stably generate similar topological network regardless of the input order of data points and noisy environment.

The results in Figs. 2 and 3 further emphasize the superiority of TCA. Although TKBA shows an excellent self-organization ability, it can be seen that the characteristics of

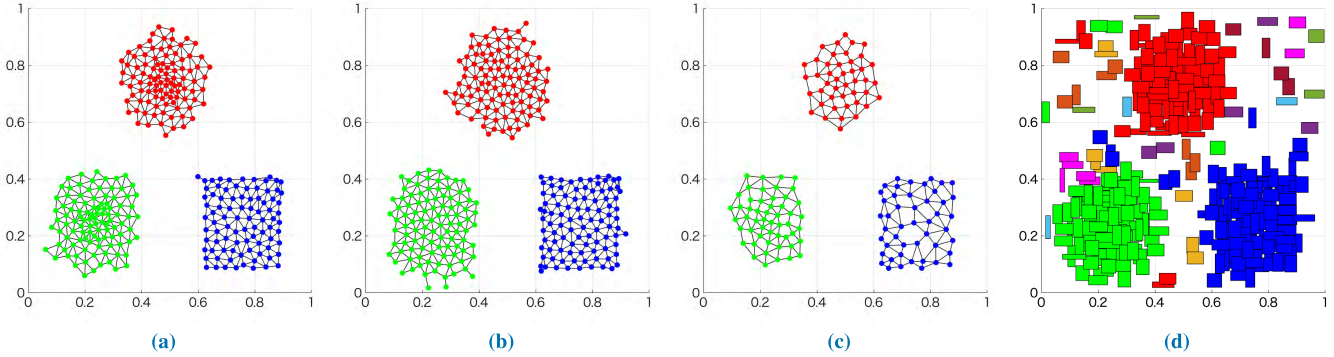

FIGURE 2. Effect of kernel bandwidth adaptation in the stationary environment (Self-organizing results). (a) TCA. (b) TKBA. (c) ASOINN. (d) TopoART.

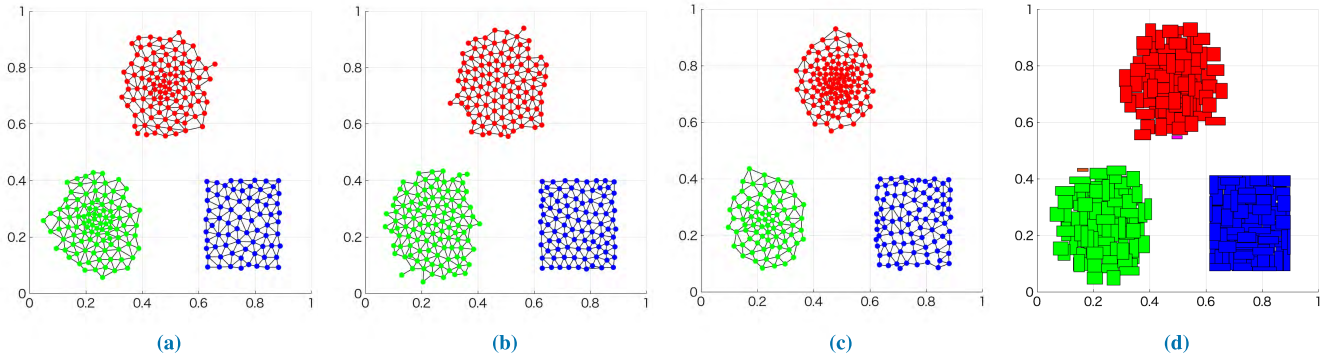

FIGURE 3. Effect of kernel bandwidth adaptation in the non-stationary environment (Self-organizing results). (a) TCA. (b) TKBA. (c) ASOINN. (d) TopoART.

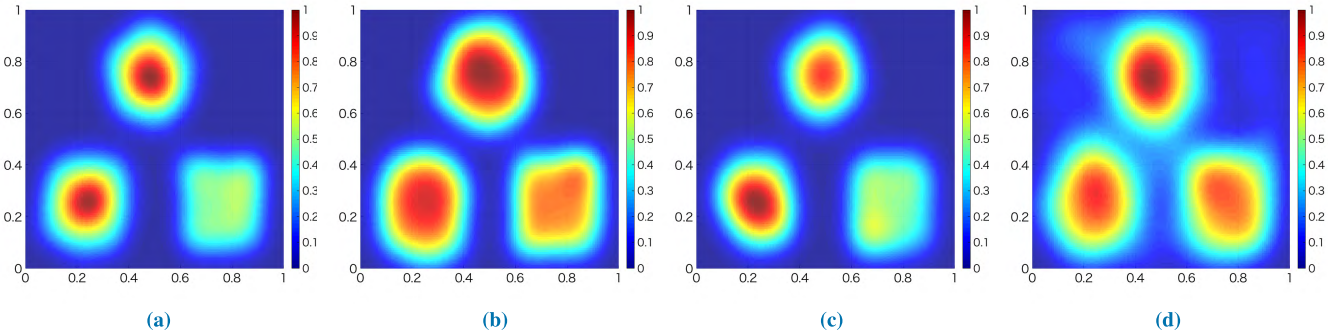

FIGURE 4. Effect of kernel bandwidth adaptation in the stationary environment (Density estimation results). (a) TCA. (b) TKBA. (c) ASOINN. (d) TopoART.

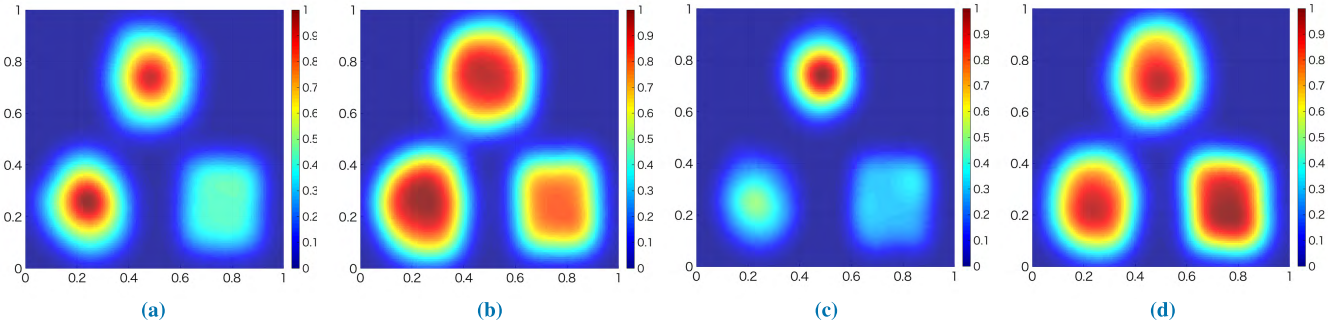

FIGURE 5. Effect of kernel bandwidth adaptation in the non-stationary environment (Density estimation results). (a) TCA. (b) TKBA. (c) ASOINN. (d) TopoART.

the data points cannot be represented by the generated topological networks. On the other hand, the density estimation results by the nodes in TCA clearly show the two Gaussian distributions and the uniform distribution compared to the results of TKBA.

B. EFFECT OF PARAMETERS IN TCA

It is essential to examine the influence of parameters in an algorithm for understanding its characteristics. This subsection further examines the characteristics of TCA, i.e., the effect of the topology construction cycle λ and the initial kernel bandwidth for CIM σ_{init} on the generated network. The dataset utilized in this subsection consists of a 2D synthetic dataset as shown in Fig. 6. The dataset is divided into six distributions as A, B, C, D, E and F, which consists of 90k data points in total (15k data points in each distribution). Here, A and B satisfy the 2D Gaussian distribution. C and D are concentric-ring distributions. E and F are sinusoidal distributions. In this experiment, the dataset shown in Fig. 6b (10% of uniform random noise is added to Fig. 6a) is utilized under the stationary environment where each data point is selected in a random order from the entire dataset only once. In regard to the parameter settings, we examine 25 combinations of five specifications of the topology construction

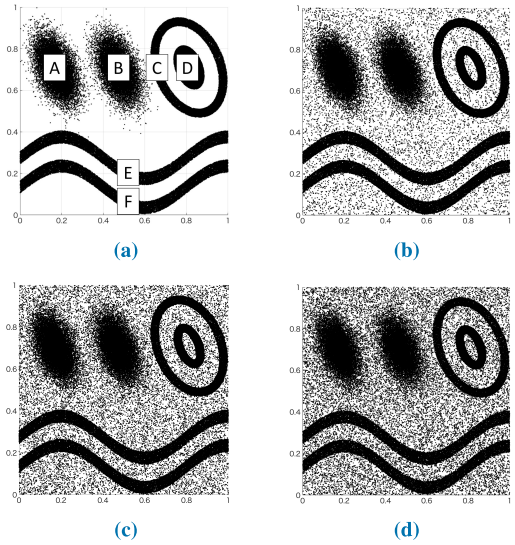

FIGURE 6. 2D synthetic dataset for self-organizing experiments. The dataset is divided into six parts as two Gaussian distributions (A and B), two concentric-ring distributions (C and D), and two sinusoidal distributions (E and F). Each distribution consists of 15k data points without noise. (a)-(d): different levels of additive noise. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

cycle $\lambda = \{100, 200, 400, 600, 1,000\}$ and the initial kernel bandwidths for CIM as $\sigma_{\text{init}} = \{0.025, 0.050, 0.075, 0.100, 0.125\}$.

Figures 7-11 show topological networks generated by TCA with the 25 settings of the two parameters. Red circle denotes generated nodes and black line represents edges between nodes. As an overall trend, with the increase in the topology construction cycle λ increases, the number of nodes also increases. On the other hand, with the increase in the initial kernel bandwidth for CIM σ_{init}, the number of nodes in the topological network decreases, the node distribution becomes more uniformly, and nodes are more easily to be connected by edges. From Figs. 7-11, we can see that σ_{init} has stronger effects than λ on the obtained topological networks. Changing the value of λ does not significantly change the obtained networks in Figs. 7-11 (except for the case of $\sigma_{\text{init}} = 0.025$).

From the above discussions, it can be concluded that σ_{init} has a greater influence than λ on the self-organizing ability of TCA.

C. SELF-ORGANIZING ABILITY

This section evaluates the performance of the self-organizing ability from subjective and quantitative perspectives. In regard to the subjective perspective, the generated topological networks are graphically presented, and visually evaluate whether the six distributions in the dataset can properly be represented by the topological network. Subjective evaluation results are presented in Sections IV-C.1 and IV-C.2. With respect to the quantitative perspective, the generated topological networks are assessed by Normalized Mutual Information (NMI) [38], Micro and Macro F-measures [39], and Adjusted Rand Index (ARI) [40]. Quantitative evaluation results are presented in Section IV-C.3.

Throughout this subsection, the parameters of TCA are set as in Table 3. According to the results in Figs. 7-11, TCA with these parameter settings can organize the topological network that properly represents each distribution without excessive nodes, and also without being affected by noise.

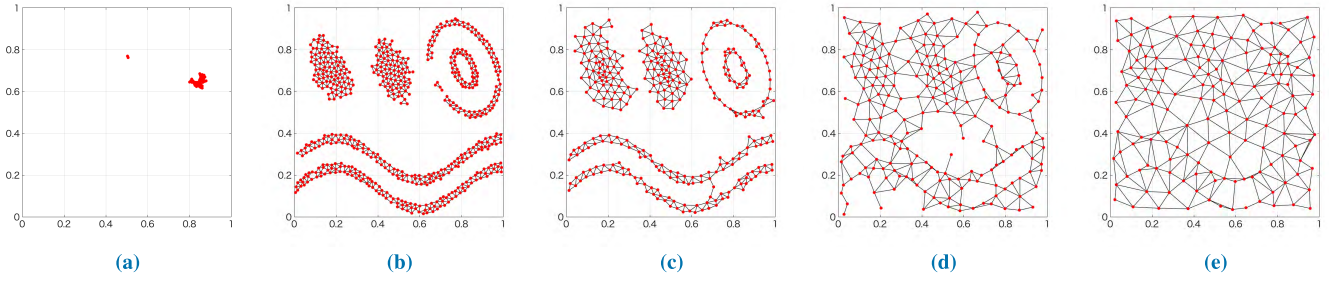

FIGURE 7. Effect of parameters in TCA (Topology construction cycle $\lambda = 100$). (a) $\sigma_{init} = 0.025$. (b) $\sigma_{init} = 0.050$. (c) $\sigma_{init} = 0.075$. (d) $\sigma_{init} = 0.100$. (e) $\sigma_{init} = 0.125$.

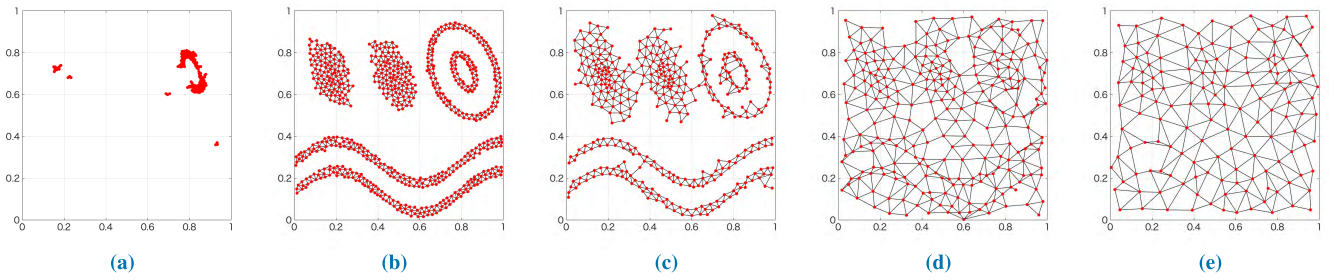

FIGURE 8. Effect of parameters in TCA (Topology construction cycle $\lambda = 200$). (a) $\sigma_{init} = 0.025$. (b) $\sigma_{init} = 0.050$. (c) $\sigma_{init} = 0.075$. (d) $\sigma_{init} = 0.100$. (e) $\sigma_{init} = 0.125$.

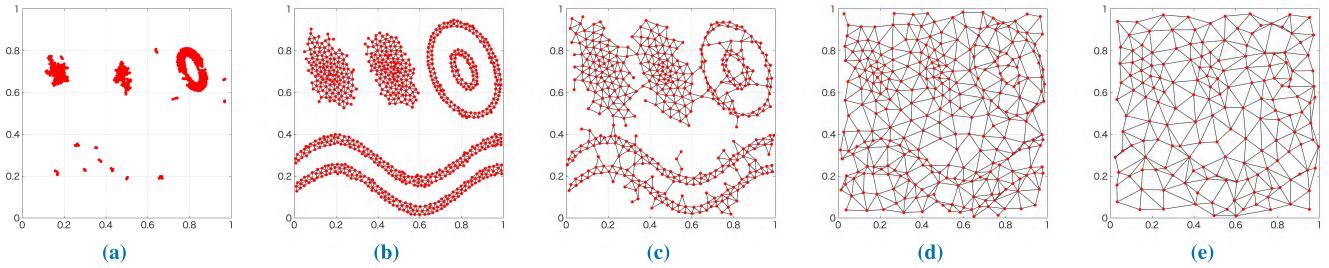

FIGURE 9. Effect of parameters in TCA (Topology construction cycle $\lambda = 400$). (a) $\sigma_{init} = 0.025$. (b) $\sigma_{init} = 0.050$. (c) $\sigma_{init} = 0.075$. (d) $\sigma_{init} = 0.100$. (e) $\sigma_{init} = 0.125$.

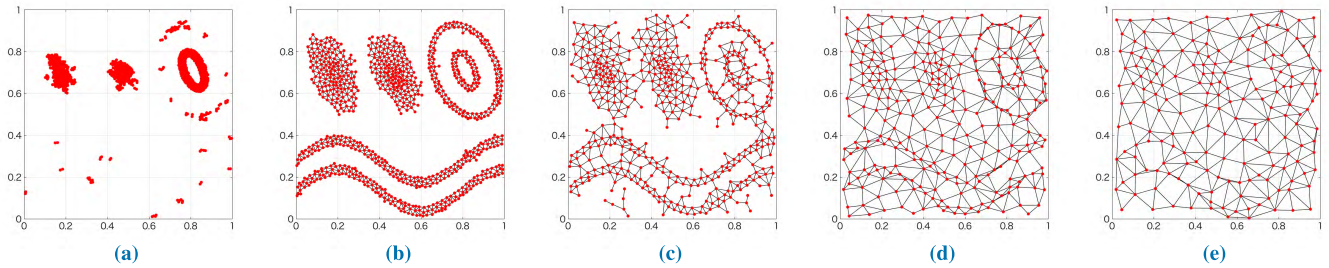

FIGURE 10. Effect of parameters in TCA (Topology construction cycle $\lambda = 600$). (a) $\sigma_{init} = 0.025$. (b) $\sigma_{init} = 0.050$. (c) $\sigma_{init} = 0.075$. (d) $\sigma_{init} = 0.100$. (e) $\sigma_{init} = 0.125$.

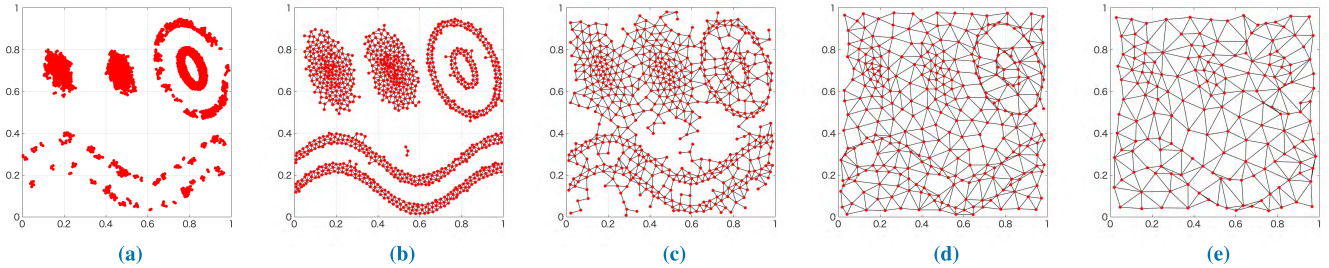

FIGURE 11. Effect of parameters in TCA (Topology construction cycle $\lambda = 1,000$). (a) $\sigma_{init} = 0.025$. (b) $\sigma_{init} = 0.050$. (c) $\sigma_{init} = 0.075$. (d) $\sigma_{init} = 0.100$. (e) $\sigma_{init} = 0.125$.

TABLE 3. Parameter settings for self-organizing ability experiment. The rest of parameters in TKBA, and the parameters of ASOINN and TopoART are the same as in Table 2.

Algorithm	Parameter	Value
TCA	topology construction cycle λ	200
	initial kernel bandwidth for CIM σ_{init}	0.05
TKBA	kernel bandwidth for CIM σ_{CIM}	0.025

Moreover, the kernel bandwidth σ_{CIM} for CIM in TKBA is set as in Table 3, which is the same setting as in [14]. The other parameters in TKBA, and the parameters in ASOINN and TopoART are the same as in Table 2.

The self-organizing ability is evaluated with the same 2D synthetic dataset as Fig. 6. In the experimental setting in [14], the data points are given to the topological network only once. However, since the ability of continuous learning is one of the significant factors in growing network algorithms, thus, in this experiment, the entire data points with noise are given to the network five times during the self-organizing, i.e., $(90\text{k} + \text{noise}) \times 5$ data points in total. In this experiment, both the stationary and non-stationary environments are considered. As explained, the data points are randomly selected from the whole dataset in the stationary environment, whereas the data points in distributions of A to F are sequentially shown to the non-stationary environment.

1) SELF-ORGANIZING ABILITY IN STATIONARY ENVIRONMENT

Figures 12-15 show generated topological networks in TCA, TKBA, ASOINN, and TopoART in the stationary environment, respectively.

As shown in Figs. 14 and 15, ASOINN and TopoART tend to generate useless nodes in the topological networks from the noise data. As a result, their topological networks cannot express data structures correctly. Focusing on Fig. 13, TKBA has better noise reduction ability than ASOINN and TopoART. However, as the noise ratio increases, the topological network in TKBA collapses. In [14], TKBA has

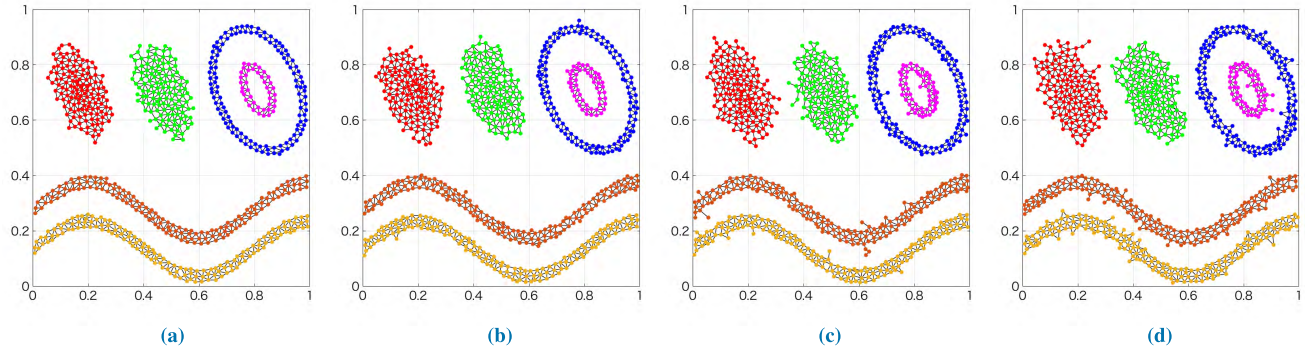

FIGURE 12. Topology construction of TCA with the stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

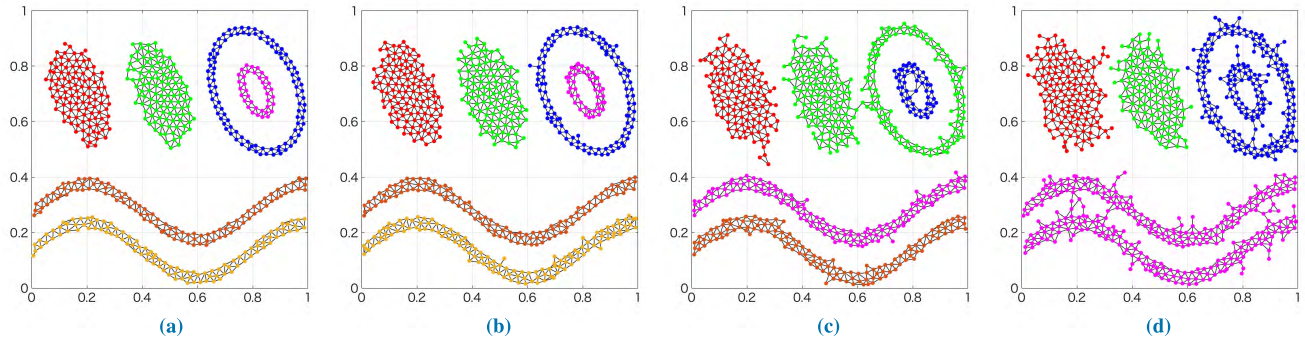

FIGURE 13. Topology construction of TKBA with the stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

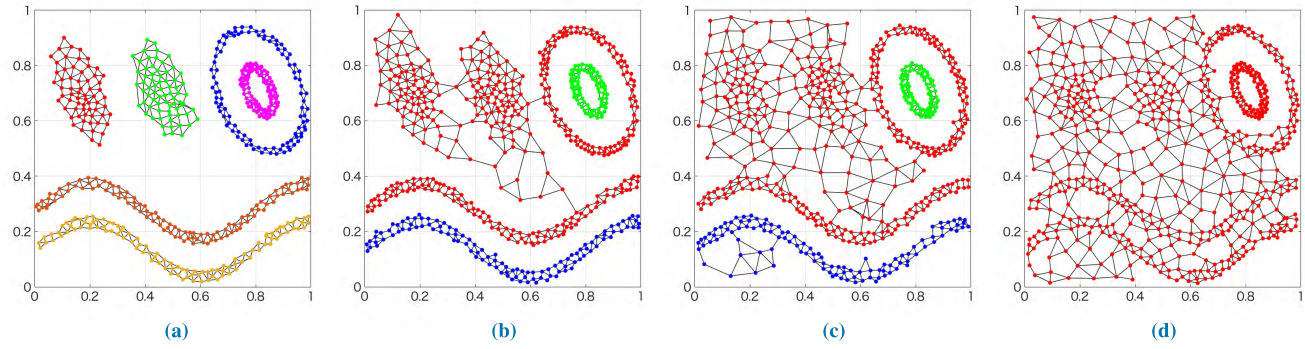

FIGURE 14. Topology construction of ASOINN with the stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

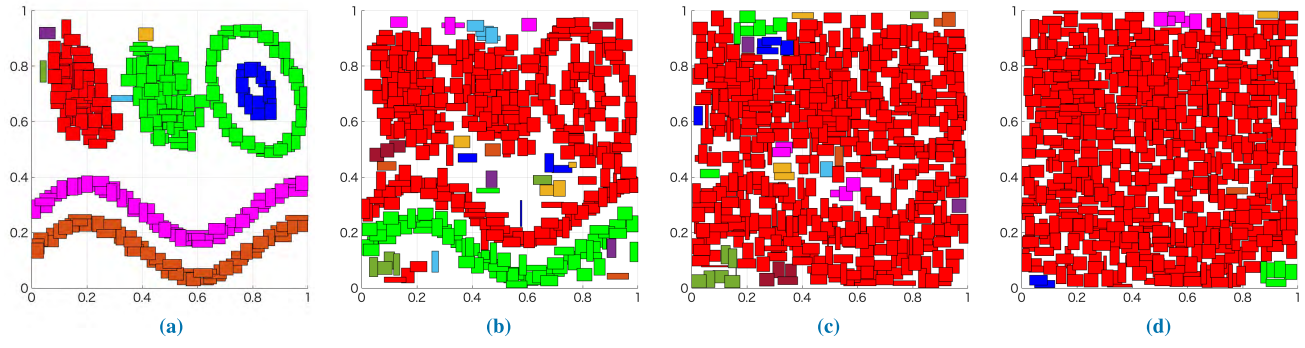

FIGURE 15. Topology construction of TopoART with the stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

successfully generated a well-organized topological network even in the environment with 30% noise. As mentioned earlier, the data points were given to the topological network only once during the experiment in [14]. In this paper, the whole data points are given five times during the experiment. As a result, TKBA generates excessive nodes from

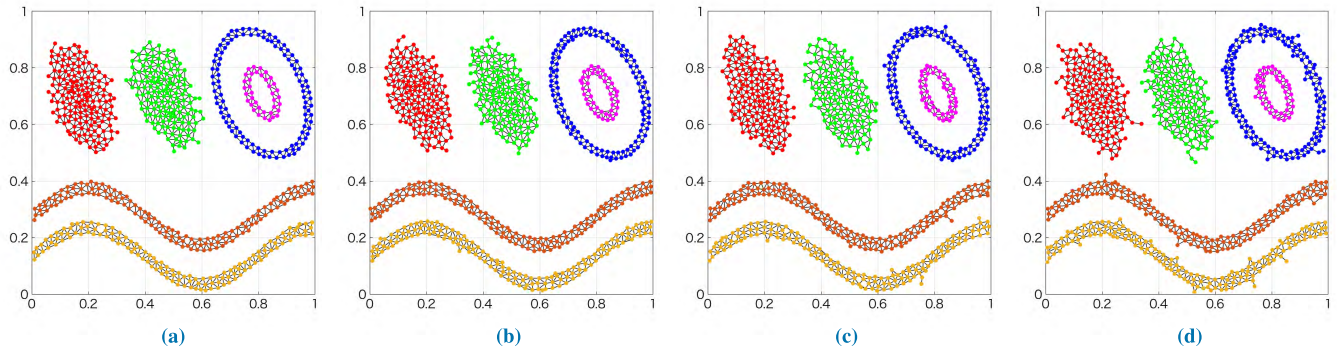

FIGURE 16. Topology construction of TCA with the non-stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

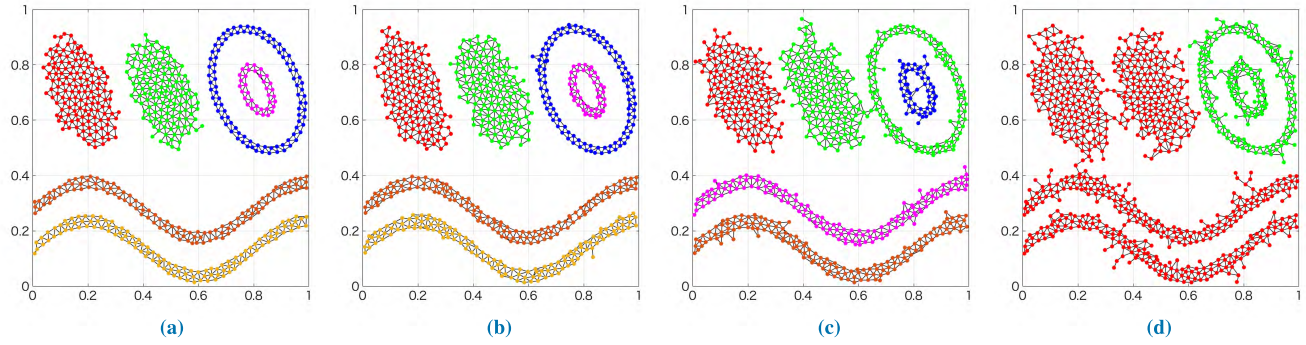

FIGURE 17. Topology construction of TKBA with the non-stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

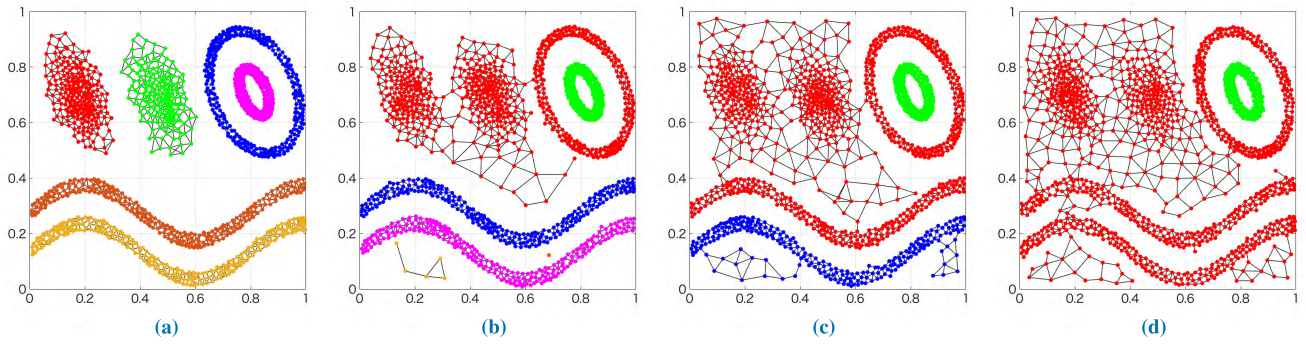

FIGURE 18. Topology construction of ASOINN with the non-stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

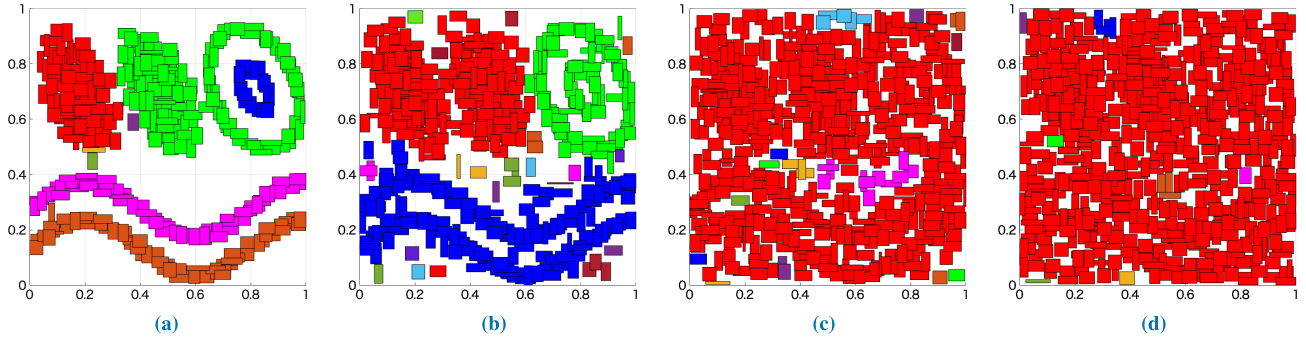

FIGURE 19. Topology construction of TopoART with the non-stationary environment. (a) Noise 0%. (b) Noise 10%. (c) Noise 20%. (d) Noise 30%.

noise information. This is one of the drawbacks of TKBA which has not discussed in [14]. On the other hand, the topological network in TCA still maintains the six distributions without serious collapses even if there are similar characteristics (e.g., generates excessive nodes) to TBKA in the noised environment.

TABLE 4. Quantitative analysis of self-organizing ability on the 2D synthetic dataset. The best value in each metric are indicated by bold. A symbol † represents that TCA has a statistically significant difference ($p < 0.05$) by the Wilcoxon signed-rank test on NMI, Micro-/Macro- F-measure, and ARI.

Input Sequence	Noise Rate	Measurement	TCA	TKBA	ASOINN	TopoART
Stationary	0 [%]	# of Nodes (SD)	655.30 (8.12)	563.46 (8.27)	544.72 (14.18)	299.28 (7.49)
		# of Clusters (SD)	6.02 (0.06)	6.00 (0.00)	5.90 (0.27)	8.70 (2.60)
		NMI (SD)	0.9981 (0.0008)	**0.9986**†(0.0008)	0.9917 (0.0184)	0.8772†(0.0845)
		Micro F-measure (SD)	0.9996 (0.0002)	**0.9997**†(0.0002)	0.9830 (0.0456)	0.7478†(0.1513)
		Macro F-measure (SD)	0.9996 (0.0002)	**0.9997**†(0.0002)	0.9774 (0.0609)	0.6756†(0.1893)
		ARI (SD)	0.9989 (0.0005)	**0.9992**†(0.0005)	0.9815 (0.0483)	0.7201†(0.1728)
	10 [%]	# of Nodes (SD)	681.10 (10.54)	629.02 (9.64)	558.70 (19.95)	421.88 (9.25)
		# of Clusters (SD)	6.02 (0.06)	6.08 (0.30)	4.10 (0.70)	35.06 (5.26)
		NMI (SD)	**0.9981** (0.0008)	0.9971 (0.0047)	0.8319†(0.0877)	0.5342†(0.2962)
		Micro F-measure (SD)	**0.9996** (0.0002)	0.9963 (0.0107)	0.6799†(0.1202)	0.3901†(0.1753)
		Macro F-measure (SD)	**0.9996** (0.0002)	0.9895 (0.0322)	0.6012†(0.1364)	0.2233†(0.1588)
		ARI (SD)	**0.9989** (0.0005)	0.9956 (0.0114)	0.5951†(0.1751)	0.3190†(0.2321)
	20 [%]	# of Nodes (SD)	703.78 (11.76)	699.32 (10.73)	585.30 (20.39)	517.44 (8.25)
		# of Clusters (SD)	6.08 (0.25)	5.42 (0.90)	2.30 (0.50)	22.18 (4.36)
		NMI (SD)	**0.9981** (0.0008)	0.9375†(0.0550)	0.5481†(0.0850)	0.0983†(0.1793)
		Micro F-measure (SD)	**0.9996** (0.0002)	0.8530†(0.1273)	0.3733†(0.0729)	0.1899†(0.0520)
		Macro F-measure (SD)	**0.9967** (0.0092)	0.7986†(0.1691)	0.2649†(0.0778)	0.0568†(0.0348)
		ARI (SD)	**0.9989** (0.0005)	0.8476†(0.1320)	0.1885†(0.0833)	0.0387†(0.0849)
	30 [%]	# of Nodes (SD)	728.50 (10.33)	778.52 (13.87)	597.56 (19.3818)	576.68 (9.10)
		# of Clusters (SD)	6.02 (0.06)	4.44 (1.35)	1.86 (0.40)	12.60 (3.41)
		NMI (SD)	**0.9981** (0.0009)	0.7516†(0.1393)	0.4212†(0.1859)	0.0000†(0.0016)
		Micro F-measure (SD)	**0.9995** (0.0002)	0.5331†(0.1403)	0.1400†(0.0618)	0.0000†(0.0000)
		Macro F-measure (SD)	**0.9967** (0.0092)	0.3861†(0.1544)	0.1200†(0.0530)	0.0000†(0.0000)
		ARI (SD)	**0.9989** (0.0005)	0.5168†(0.1778)	0.1199†(0.0529)	0.0000†(0.0000)
Non-stationary	0 [%]	# of Nodes (SD)	755.52 (26.64)	651.72 (7.20)	1482.40 (39.52)	355.46 (6.80)
		# of Clusters (SD)	6.02 (0.06)	6.00 (0.00)	5.46 (0.52)	7.58 (1.63)
		NMI (SD)	0.9988 (0.0006)	**0.9990**†(0.0006)	0.9630†(0.0344)	0.9543†(0.0539)
		Micro F-measure (SD)	0.9997 (0.0001)	**0.9998**†(0.0001)	0.9098†(0.0863)	0.8993†(0.1232)
		Macro F-measure (SD)	0.9997 (0.0001)	**0.9998**†(0.0001)	0.8798†(0.1151)	0.8673†(0.1622)
		ARI (SD)	0.9993 (0.0003)	**0.9994**†(0.0003)	0.9042†(0.0913)	0.8943†(0.1268)
	10 [%]	# of Nodes (SD)	774.64 (21.90)	695.18 (9.62)	1414.90 (42.89)	439.14 (7.74)
		# of Clusters (SD)	6.00 (0.00)	5.86 (0.34)	5.32 (0.82)	30.08 (4.62)
		NMI (SD)	**0.9987** (0.0009)	0.9870†(0.0225)	0.8689†(0.0580)	0.5746†(0.2633)
		Micro F-measure (SD)	**0.9997** (0.0002)	0.9698†(0.0559)	0.7300†(0.0912)	0.4038†(0.1637)
		Macro F-measure (SD)	**0.9997** (0.0002)	0.9576†(0.0815)	0.6573†(0.1055)	0.2297†(0.1482)
		ARI (SD)	**0.9993** (0.0005)	0.9677†(0.0592)	0.6701†(0.1378)	0.3655†(0.2242)
	20 [%]	# of Nodes (SD)	788.04 (15.99)	749.72 (11.11)	1372.30 (30.61)	543.14 (9.62)
		# of Clusters (SD)	5.96 (0.18)	5.01 (1.09)	4.52 (1.12)	22.84 (4.53)
		NMI (SD)	**0.9947** (0.0103)	0.9081†(0.0742)	0.7433†(0.1029)	0.0418†(0.0814)
		Micro F-measure (SD)	**0.9897** (0.0247)	0.7932†(0.1538)	0.5633†(0.1170)	0.1728†(0.0218)
		Macro F-measure (SD)	**0.9864** (0.0329)	0.7245†(0.1965)	0.4692†(0.1291)	0.0453†(0.0143)
		ARI (SD)	**0.9887** (0.0263)	0.7826†(0.1595)	0.4332†(0.1634)	0.0111†(0.0348)
	30 [%]	# of Nodes (SD)	820.26 (17.27)	824.18 (13.61)	1349.60 (35.38)	593.66 (9.87)
		# of Clusters (SD)	5.94 (0.41)	3.22 (1.14)	3.06 (0.99)	8.02 (2.68)
		NMI (SD)	**0.9907** (0.0192)	0.6404†(0.1732)	0.5936†(0.0988)	0.0003†(0.0010)
		Micro F-measure (SD)	**0.9797** (0.0477)	0.4233†(0.1173)	0.2567†(0.1033)	0.0000†(0.0000)
		Macro F-measure (SD)	**0.9730** (0.0636)	0.2813†(0.1222)	0.2191†(0.0872)	0.0000†(0.0001)
		ARI (SD)	**0.9781** (0.0506)	0.0373†(0.1633)	0.2351†(0.1012)	0.0000†(0.0000)

2) SELF-ORGANIZING ABILITY IN NON-STATIONARY ENVIRONMENT

Figures 16-19 show generated topological networks in TCA, TKBA, ASOINN, and TopoART in the non-stationary environment, respectively.

The overall trends of self-organizing results by each algorithm are the same as the one in the stationary environment, i.e., TKBA tends to generate excessive nodes in the noised environment, and ASOINN and TopoART suffer from their sensitivity to noise information. In regard to TCA, similar results to the case of the stationary environment are obtained.

From the comparison of the generated topological networks between the stationary environment (Figs. 12-15) and the non-stationary environment (Figs. 16-19), we can see large differences in the density of nodes in the generated networks by AOINN (i.e., Figs. 14 and 18).

The results in Figs. 14 and 18 show high sensitivity to the self-organizing ability of ASOINN to the input order of data points. In general, high sensitivity of the input order to data points is one of a well-known problems [41]. In contrast to ASOINN, TCA and TKBA can generate similar topological networks in both environments. High reproducibility of

self-organizing results is one of the significant advantages of TCA and TKBA.

3) QUANTITATIVE ANALYSIS

The quantitative analysis of self-organizing ability is assessed by NMI [38], Micro and Macro F-measures [39], and ARI [40]. In order to calculate the above metrics, topological networks are utilized as classifiers to realize classification tasks. Specifically, the label information is assigned to data points of each distribution A to F in Fig. 6 as class 1 to 6. Then, 90% of the data points in Fig. 6 are randomly selected to generate the topological network and perform the classification task for the remaining 10% of data points. The label information of each cluster is determined by the majority vote from the labeled data points assigned to the nodes in the cluster. To reduce the bias resulting from the random sampling of data points, 10-fold cross-validation is utilized. Moreover, all the experiments are conducted five times to obtain the consistent averaging results (i.e., five times of the 10-fold cross-validation procedure). Finally, to confirm the validity of the obtained results, the Wilcoxon signed-rank test [42] is employed to validate whether the performance of TCA has a statistically significant difference from that of the following algorithms: TKBA, ASOINN, and TopoART. Here, the null hypothesis is rejected at the significant level of 0.05.

The results of the quantitative analysis are summarized in Table 4. Since TKBA has the ability to generate considerably stable topological networks, TKBA shows remarkable results in NMI, Micro/Macro F-measure, and ARI when no noise is added in the environment. As shown in Figs. 12 to 19, however, TKBA, ASOINN, and TopoART cannot generate appropriate topological networks when the noise rate is high. As a result, the performance of these algorithms is dramatically dropped. Focusing on TCA, TCA also shows comparable performance to TKBA in the stationary and non-stationary environments with 0% noise although the statistical test shows that TKBA is significantly better than TCA. It is noteworthy that TCA shows outstanding performance when a large amount of noise is added in the environments.

With respect to the number of nodes constituting the topological network of each algorithm, as the noise rate increases, the frequency of node generation due to noise also increases. As a result, the total number of nodes in the network increases. Even though the noise rate is 0%, TCA tends to generate a large number of nodes. However, the number of added nodes, due to the increase of the noise rate, is much smaller than TKBA, ASOINN, and TopoART. This observation clearly indicates the superior stability and reproducibility of TCA.

In our self-organizing ability experiments, TCA realized better noise reduction ability while reducing the number of parameters compared to the representative topological clustering algorithms. Moreover, TCA successfully shows the ability to express density information of data.

TABLE 5. Configurations of UCI dataset.

Dataset	# of Attributes	# of Classes	# of Samples
Skin	3	2	245,057
Iris	4	3	150
Shuttle	9	7	58,000
Wine	13	3	178
Zoo	16	7	101
Letter	16	26	20,000
Sonar	60	2	208

TABLE 6. Parameter settings for classification experiment on UCI datasets. The rest of parameters of each algorithm are the same as Table 2.

Dataset	TCA σ_{init}	TKBA σ_{CIM}	ASOINN c	TopoART $(\rho_{\text{a}}, \rho_{\text{b}})$
Skin	0.080	0.100	0.500	(0.500, 0.750)
Iris	0.400	0.500	1.000	(0.550, 0.775)
Shuttle	0.350	0.370	0.400	(0.980, 0.990)
Wine	0.870	2.500	0.500	(0.200, 0.600)
Zoo	0.915	3.500	0.300	(0.550, 0.775)
Letter	0.740	1.900	0.110	(0.540, 0.770)
Sonar	0.980	5.500	1.000	(0.010, 0.505)

D. CLASSIFICATION ABILITY

This subsection presents the classification performance of TCA, TKBA, ASOINN, and TopoART by utilizing seven real-world datasets from the UCI repository of machine learning databases [43]. The datasets in this experiment are listed in Table 5.

The parameters of each algorithm are summarized in Table 6. The rest of the parameters are the same as those in Table 2. The parameters of TKBA, ASOINN, and TopoART in Table 6 are the same settings in [14]. In regards to TCA, the parameters are tuned by preliminary experiments utilizing 50% of samples in each dataset. We repeatedly calculate with different parameter settings and adopt the parameters that obtained the highest NMI. This procedure is exactly the same manner as [14]. However, the parameter settings are changed in an arbitrarily fixed range when changing the parameter condition, there is a possibility of finding an even better parameter setting by sophisticated optimization algorithms.

Similar to the self-organizing ability assessment, the label information of each cluster is determined by the majority vote from the labeled data points assigned to the nodes in the cluster, and 10-fold cross-validation method is utilized with 20 trials for obtaining consistent averaging results. In addition, during the learning sequence, each data point is shown 100 times to each algorithm. Although the several datasets provide training and test data independently, both data are integrated randomly and cross-validation applied to the entire data. Furthermore, the Wilcoxon signed-rank test [42] is employed to validate whether the performance of

TABLE 7. Classification performance of TCA, TKBA, ASOINN, and TopoART on the UCI datasets. The best value in each metric are indicated by bold. A symbol † represents that TCA has a statistically significant difference ($p < 0.05$) by the Wilcoxon signed-rank test on NMI, Micro-/Macro- F-measure, and ARI.

Dataset	Measurement	TCA	TKBA	ASOINN	TopoART
Skin	# of Nodes (SD)	1063.30 (35.13)	906.95 (20.45)	254.20 (13.28)	92.45 (5.32)
	# of Clusters (SD)	26.90 (3.42)	28.05 (3.65)	17.80 (2.97)	2.30 (0.93)
	NMI (SD)	0.839 (0.025)	**0.886**†(0.030)	0.676†(0.048)	0.000†(0.000)
	Micro F-measure (SD)	0.980 (0.004)	**0.986**†(0.005)	0.946†(0.013)	0.791†(0.005)
	Macro F-measure (SD)	0.971 (0.005)	**0.980**†(0.007)	0.925†(0.017)	0.442†(0.002)
	ARI (SD)	0.913 (0.016)	**0.940**†(0.021)	0.779†(0.048)	0.000†(0.000)
Iris	# of Nodes (SD)	25.80 (3.77)	22.06 (1.89)	44.99 (3.31)	18.36 (1.41)
	# of Clusters (SD)	6.98 (1.44)	6.84 (1.30)	5.42 (1.96)	4.40 (1.40)
	NMI (SD)	**0.805** (0.144)	0.800 (0.135)	0.748†(0.125)	0.679†(0.168)
	Micro F-measure (SD)	**0.890** (0.094)	0.882 (0.098)	0.761†(0.146)	0.624†(0.143)
	Macro F-measure (SD)	**0.877** (0.103)	0.864 (0.114)	0.707†(0.167)	0.555†(0.146)
	ARI (SD)	**0.734** (0.213)	0.725 (0.205)	0.615†(0.206)	0.517†(0.205)
Shuttle	# of Nodes (SD)	223.25 (23.71)	387.50 (6.11)	260.05 (10.68)	334.50 (13.40)
	# of Clusters (SD)	11.05 (1.72)	7.20 (0.96)	13.50 (3.63)	18.30 (2.80)
	NMI (SD)	0.724 (0.109)	0.678 (0.014)	**0.733** (0.067)	0.490†(0.097)
	Micro F-measure (SD)	0.943 (0.027)	0.926†(0.005)	**0.954** (0.018)	0.807†(0.110)
	Macro F-measure (SD)	**0.453** (0.058)	0.437†(0.051)	0.446 (0.068)	0.335†(0.075)
	ARI (SD)	0.781 (0.109)	0.709†(0.017)	**0.827** (0.077)	0.395†(0.157)
Wine	# of Nodes (SD)	40.08 (7.08)	22.77 (2.27)	53.76 (3.55)	27.81 (2.11)
	# of Clusters (SD)	11.17 (1.92)	5.50 (1.05)	8.83 (2.94)	5.78 (2.21)
	NMI (SD)	0.803 (0.152)	**0.810** (0.128)	0.652†(0.192)	0.098†(0.117)
	Micro F-measure (SD)	0.908 (0.083)	**0.915** (0.066)	0.788†(0.143)	0.401†(0.123)
	Macro F-measure (SD)	0.900 (0.099)	**0.907** (0.077)	0.748†(0.178)	0.245†(0.109)
	ARI (SD)	0.752 (0.196)	**0.765** (0.170)	0.556†(0.246)	0.033†(0.066)
Zoo	# of Nodes (SD)	15.34 (4.88)	12.63 (1.67)	32.75 (3.07)	20.35 (1.30)
	# of Clusters (SD)	6.43 (1.21)	5.89 (0.81)	7.66 (1.87)	11.45 (1.82)
	NMI (SD)	0.858 (0.120)	**0.908**†(0.084)	0.841 (0.165)	0.883†(0.105)
	Micro F-measure (SD)	0.780 (0.156)	**0.781** (0.143)	0.768 (0.158)	0.783 (0.159)
	Macro F-measure (SD)	**0.586** (0.223)	0.580 (0.198)	0.584 (0.230)	0.628†(0.212)
	ARI (SD)	0.713 (0.247)	**0.807**†(0.196)	0.701 (0.270)	0.754†(0.233)
Letter	# of Nodes (SD)	1184.95 (17.21)	574.20 (15.43)	483.50 (20.30)	896.10 (17.86)
	# of Clusters (SD)	77.20 (9.25)	56.05 (3.98)	25.80 (3.56)	39.10 (6.51)
	NMI (SD)	0.520 (0.039)	**0.536** (0.025)	0.375†(0.041)	0.061†(0.012)
	Micro F-measure (SD)	0.389 (0.041)	**0.437**†(0.032)	0.232†(0.034)	0.039†(0.006)
	Macro F-measure (SD)	0.419 (0.033)	**0.443**†(0.032)	0.267†(0.038)	0.012†(0.006)
	ARI (SD)	0.084 (0.027)	**0.159**†(0.040)	0.020†(0.007)	0.000†(0.000)
Sonar	# of Nodes (SD)	17.14 (27.12)	12.56 (2.01)	38.19 (3.74)	29.15 (1.19)
	# of Clusters (SD)	13.65 (26.86)	5.67 (1.11)	9.53 (2.49)	15.66 (3.86)
	NMI (SD)	0.113 (0.121)	0.107†(0.109)	**0.131** (0.120)	0.065†(0.071)
	Micro F-measure (SD)	0.611 (0.124)	0.614 (0.116)	**0.645**†(0.106)	0.540†(0.106)
	Macro F-measure (SD)	0.543 (0.143)	0.559 (0.127)	**0.611**†(0.122)	0.489 (0.116)
	ARI (SD)	0.077 (0.114)	0.069 (0.096)	**0.098**†(0.126)	0.028†(0.046)

TCA has a statistically significant difference from that of each comparison algorithms. Here, the null hypothesis is rejected at the significant level of 0.05.

Table 7 shows the results of classification performance. As an overall trend, TCA and TKBA show better performance than ASOINN and TopoART. Compared to TKBA, TCA shows an equivalent classification performance with respect to Iris, Wine, and Sonar datasets. TCA shows superior classification performance to Shuttle dataset, and that of Skin and Letter datasets are inferior to TKBA with the statistically significant difference. Thus, we can regard that TCA and TKBA have comparable classification ability.

V. CONCLUSION

This paper introduced a new ART-based topological clustering algorithm with the information theoretic learning in order to tackle the several problems of the existing clustering algorithms. The proposed algorithm, i.e., TCA, utilized CIM for defining a similarity measure, a node insertion criterion, and an edge creation criterion. As a result, TCA successfully reduced the number of parameters. The experimental results of the self-organizing ability showed that TCA has high noise reduction ability and stable topological network creation ability. In summary, TCA not only overcame the weakness of the state-of-the-art topological clustering algorithm TKBA, but also improved its self-organizing ability (especially the noise reduction ability). Although the classification ability of TCA for real-world data is not clearly better than TKBA, TCA is a more useful and highly applicable algorithm because of the number of parameters in TCA is much less than TKBA as shown in Table 2 (i.e., two parameters in TCA and seven parameters in TKBA).

In summary, the contributions of TCA are as follows;

1) TCA acquired the high noise reduction ability and successfully reduced the number of parameters compared to existing algorithms.
2) The topological network by TCA can properly represent the structure of data such as density information based on the mechanism that adaptively controls the node insertion criterion.
3) The above-mentioned features are achieved with maintaining the superior self-organizing and classification abilities compared to existing algorithms.

With the above contributions, the typical problems of the self-organizing growing network algorithms can be dealt with.

One future research direction is to examine the use of a hierarchical structure in TCA for improving its self-organizing ability and for extending its functionality from a structural perspective. Another future research direction is to examine an alternative kernel bandwidth adaptation mechanism to further reduce the number of parameters in TCA to be pre-specified. In regards to further improvement of CIM, a generalized CIM is one of the successful approaches. The generalized CIM [44] is induced from correntropy defined by a generalized Gaussian density function [45] (instead of the Gaussian kernel function). It is worth noting that in the generalized CIM, the kernel function does not necessarily satisfy the Mercer's condition. Thus, there is a possibility to improve the performance of TCA by utilizing the generalized CIM.

REFERENCES

[1] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.
[2] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the *EM* algorithm," *J. Roy. Stat. Soc. B, Methodol.*, vol. 39, no. 1, pp. 1–22, 1977.
[3] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.*, vol. 43, no. 1, pp. 59–69, 1982.
[4] B. Fritzke, "Growing cell structures—A self-organizing network for unsupervised and supervised learning," *Neural Netw.*, vol. 7, no. 9, pp. 1441–1460, 1994.
[5] B. Fritzke, "A growing neural gas network learns topologies," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 7. 1995, pp. 625–632.
[6] S. Furao and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning," *Neural Netw.*, vol. 19, no. 1, pp. 90–106, Jan. 2006.
[7] G. A. Carpenter and S. Grossberg, "The ART of adaptive pattern recognition by a self-organizing neural network," *Computer*, vol. 21, no. 3, pp. 77–88, Mar. 1988.
[8] S. Grossberg, "Competitive learning: From interactive activation to adaptive resonance," *Cognit. Sci.*, vol. 11, no. 1, pp. 23–63, 1987.
[9] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Netw.*, vol. 4, no. 6, pp. 759–771, 1991.
[10] M. Tscherepanow, "TopoART: A topology learning hierarchical ART network," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2010, pp. 157–167.
[11] N. Masuyama, C. K. Loo, and F. Dawood, "Kernel Bayesian ART and ARTMAP," *Neural Netw.*, vol. 98, pp. 76–86, Feb. 2018.
[12] K. Fukumizu, L. Song, and A. Gretton, "Kernel Bayes' rule: Bayesian inference with positive definite kernels," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 3753–3783, 2013.
[13] R. Chalasani and J. C. Príncipe, "Self-organizing maps with information theoretic learning," *Neurocomputing*, vol. 147, pp. 3–14, Jan. 2015.
[14] N. Masuyama, C. K. Loo, and S. Wermter, "A kernel Bayesian adaptive resonance theory with a topological structure," *Int. J. Neural Syst.*, vol. 29, Oct. 2018, Art. no. 1850052. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/S0129065718500521
[15] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 2013.
[16] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.
[17] K. Michał and B. Cyganek, "Image recognition with deep neural networks in presence of noise—Dealing with and taking advantage of distortions," *Integr. Comput.-Aided Eng.*, vol. 24, no. 4, pp. 337–349, 2017.
[18] U. R. Acharya, S. L. Oh, Y. Hagiwara, J. H. Tan, and H. Adeli, "Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals," *Comput. Biol. Med.*, vol. 100, pp. 270–278, Sep. 2017.
[19] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 3, pp. 267–279, Sep. 2014.
[20] I. Khan, J. Z. Huang, Z. Luo, and M. A. Masud, "CPLP: An algorithm for tracking the changes of power consumption patterns in load profile data over time," *Inf. Sci.*, vol. 429, pp. 332–348, Mar. 2018.
[21] I. Khan and Z. Luo, "Nonnegative matrix factorization based consensus for clusterings with a variable number of clusters," *IEEE Access*, vol. 6, pp. 73158–73169, 2018.
[22] O. Beyer and P. Cimiano, "Online semi-supervised growing neural gas," *Int. J. Neural Syst.*, vol. 22, no. 5, 2012, Art. no. 1250023.
[23] E. López-Rubio, E. J. Palomo, and E. Domínguez, "Bregman divergences for growing hierarchical self-organizing networks," *Int. J. Neural Syst.*, vol. 24, no. 4, 2014, Art. no. 1450016.
[24] S. Marsland, J. Shapiro, and U. Nehmzow, "A self-organising network that grows when required," *Neural Netw.*, vol. 15, nos. 8–9, pp. 1041–1058, 2002.
[25] W. Liu, P. P. Pokharel, and J. C. Príncipe, "Correntropy: Properties and applications in non-Gaussian signal processing," *IEEE Trans. Signal Process.*, vol. 55, no. 11, pp. 5286–5298, Nov. 2007.
[26] N. Masuyama and C. K. Loo, "Growing neural gas with correntropy induced metric," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2016, pp. 1–7.
[27] S. Furao, T. Ogura, and O. Hasegawa, "An enhanced self-organizing incremental neural network for online unsupervised learning," *Neural Netw.*, vol. 20, no. 8, pp. 893–903, Oct. 2007.
[28] F. Shen and O. Hasegawa, "A fast nearest neighbor classifier based on self-organizing incremental neural network," *Neural Netw.*, vol. 21, no. 10, pp. 1537–1547, Dec. 2008.
[29] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, no. 3, pp. 1065–1076, Sep. 1962.
[30] Y. Nakamura and O. Hasegawa, "Nonparametric density estimation based on self-organizing incremental neural network for large noisy data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 1, pp. 8–17, Jan. 2017.
[31] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *Proc. Int. Conf. Database Theory*. Berlin, Germany: Springer, 2001, pp. 420–434.
[32] S. Marriott and R. F. Harrison, "A modified fuzzy ARTMAP architecture for the approximation of noisy mappings," *Neural Netw.*, vol. 8, no. 4, pp. 619–641, 1995.
[33] B. Vigdor and B. Lerner, "The Bayesian ARTMAP," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1628–1644, Nov. 2007.
[34] S. Anatolyev, R. Khabibullin, and A. Prokhorov, "An algorithm for constructing high dimensional distributions from distributions of lower dimension," *Econ. Lett.*, vol. 123, no. 3, pp. 257–261, 2014.
[35] S. Seth and J. C. Príncipe, "Compressed signal reconstruction using the correntropy induced metric," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar./Apr. 2008, pp. 3845–3848.
[36] T. H. Cormen, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2009.
[37] M. P. Wand, "Fast computation of multivariate kernel estimators," *J. Comput. Graph. Statist.*, vol. 3, no. 4, pp. 433–445, 1994.
[38] A. Strehl and J. Ghosh, "Cluster ensembles—A knowledge reuse framework for combining multiple partitions," *J. Mach. Learn. Res.*, vol. 3, pp. 583–617, Dec. 2002.

[39] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009.
[40] L. Hubert and P. Arabie, "Comparing partitions," *J. Classification*, vol. 2, no. 1, pp. 193–218, 1985.
[41] W. A. van der Kloot, A. M. Spaans, and W. J. Heiser, "Instability of hierarchical cluster analysis due to input order of the data: The permucluster solution," *Psychol. Methods*, vol. 10, no. 4, pp. 468–476, 2005.
[42] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.
[43] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, Irvine, CA, USA, 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[44] B. Chen, L. Xing, H. Zhao, N. Zheng, and J. C. Príncipe, "Generalized correntropy for robust adaptive filtering," *IEEE Trans. Signal Process.*, vol. 64, no. 13, pp. 3376–3387, 2016.
[45] B. Chen, J. C. Príncipe, J. Hu, and Y. Zhu, "Stochastic information gradient algorithm with generalized Gaussian distribution model," *J. Circuits, Syst., Comput.*, vol. 21, no. 01, 2012, Art. no. 1250006.

NAOKI MASUYAMA (S'12–M'16) received the degree from Nihon University, in 2010, the M.E. degree from Tokyo Metropolitan University, in 2012, and the Ph.D. degree from the Faculty of Computer Science and Information Technology, University of Malaya, Malaysia, in 2016.

He is currently an Assistant Professor with the Graduate School of Engineering, Osaka Prefecture University, Japan. His research interests include data mining and machine learning.

CHU KIONG LOO (SM'14) received the B.Eng. degree (Hons.) in mechanical engineering from the University of Malaya, in 1996, and the Ph.D. degree from University Sains Malaysia, in 2004, specializing in neurorobotics.

He is currently a Full Professor with the Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya. His main research interest includes neuroscience inspired machine intelligence. He is also a Reviewer of several international technical journals, which include the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. He was the IEEE Systems, Man and Cybernetics SMC Society Vice-Chairman for Malaysia Chapter, from 2013 to 2014. He was also the President of Asia Pacific Neural Network Assembly (APNNA), in 2014.

HISAO ISHIBUCHI (M'93–SM'10–F'14) received the B.S. and M.S. degrees in precision mechanics from Kyoto University, Kyoto, Japan, in 1985 and 1987, respectively, and the Ph.D. degree in computer science from Osaka Prefecture University, Sakai, Osaka, Japan, in 1992, where he has been since 1987. Since 2017, he has been a Chair Professor with the Southern University of Science and Technology, China. His research interests include fuzzy rule-based classifiers, evolutionary multi-objective and many-objective optimization, memetic algorithms, and evolutionary games.

Dr. Ishibuchi was the IEEE Computational Intelligence Society (CIS) Vice-President for Technical Activities, from 2010 to 2013. He is currently an AdCom Member of the IEEE CIS, from 2014 to 2019, and the Editor-in-Chief of the *IEEE Computational Intelligence Magazine*, from 2014 to 2019.

NAOYUKI KUBOTA (M'01) received the M.Eng. degree from Hokkaido University, Hokkaido, Japan, in 1994, and the D.E. degree from Nagoya University, Nagoya, Japan, in 1997.

He joined the Osaka Institute of Technology, Osaka, Japan, in 1997. In 2000, he joined the Department of Human and Artificial Intelligence Systems, Fukui University, Fukui, Japan, as an Associate Professor. He joined the Department of Mechanical Engineering, Tokyo Metropolitan University, Tokyo, Japan, in 2004, where he is a Professor with the Department of System Design.

YUSUKE NOJIMA received the B.S. and M.S. degrees in mechanical engineering from the Osaka Institute of Technology, Osaka, Japan, in 1999 and 2001, respectively, and the Ph.D. degree in system function science from Kobe University, Hyogo, Japan, in 2004.

Since 2004, he has been with Osaka Prefecture University, Osaka, where he was a Research Associate and is currently an Associate Professor with Department of Computer Science and Intelligent Systems. His research interests include evolutionary machine learning and evolutionary multiobjective optimization.

YIPING LIU (M'18) received the Ph.D. degree in control theory and control engineering from China University of Mining and Technology, China in 2017. He was a joint Ph.D. student financed by the China Scholarship Council during 2016–2017 in the School of Electrical and Computer Engineering, Oklahoma State University, USA.

He is currently a Research Assistant Professor with the Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Japan. His research interests include computational intelligence, evolutionary computation, multi-objective optimization, and machine learning.