https://doi.org/10.20965/jaciii.2021.p1011

Paper:

Multilayer Batch Learning Growing Neural Gas for Learning Multiscale Topologies

Yuichiro Toda[†], **Takayuki Matsuno, and Mamoru Minami**

Okayama University
3-1-1 Tsushima-Naka, Kita-Ku, Okayama, Okayama 700-8530, Japan
E-mail: ytoda@okayama-u.ac.jp
[†]Corresponding author

[Received October 2, 2019; accepted September 14, 2021]

Hierarchical topological structure learning methods are expected to be developed in the field of data mining for extracting multiscale topological structures from an unknown dataset. However, most methods require user-defined parameters, and it is difficult for users to determine these parameters and effectively utilize the method. In this paper, we propose a new parameter-less hierarchical topological structure learning method based on growing neural gas (GNG). First, we propose batch learning GNG (BL-GNG) to improve the learning convergence and reduce the user-designed parameters in GNG. BL-GNG uses an objective function based on fuzzy C-means to improve the learning convergence. Next, we propose multilayer BL-GNG (MBL-GNG), which is a parameter-less unsupervised learning algorithm based on hierarchical topological structure learning. In MBL-GNG, the input data of each layer uses parent nodes to learn more abstract topological structures from the dataset. Furthermore, MBL-GNG can automatically determine the number of nodes and layers according to the data distribution. Finally, we conducted several experiments to evaluate our proposed method by comparing it with other hierarchical approaches and discuss the effectiveness of our proposed method.

Keywords: growing neural gas, hierarchical competitive learning, unsupervised learning

1. Introduction

Unsupervised learning methods are expected in various fields, such as data mining. The objectives of unsupervised learning are feature extraction, clustering, and learning the topological structure of a dataset to efficiently find important information [1]. Many methods have been proposed and have shown effective results in achieving these objectives [2–6]. However, many problems still exist with unsupervised learning. In this study, we focused on two problems of unsupervised learning. First, the necessary clustering results depend on the situation or application. For example, **Fig. 1(a)** shows an example of a dataset. If

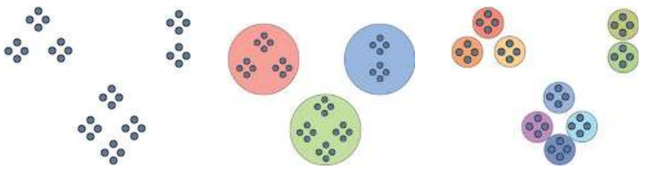


Fig. 1. An example of a dataset for clustering: A gray circle indicates an input data.

we need to obtain a rough result, the dataset should be divided into three clusters (**Fig. 1(b)**). By contrast, if we need a detailed result, the dataset should be divided into nine clusters (**Fig. 1(c)**). In this way, unsupervised learning has the problem of generating clustering results at different scales. Another problem is related to the parameter settings. As previously mentioned, many effective unsupervised learning methods have been proposed. However, most of these methods require user-designed parameters. It is difficult for a user to set the parameters because in many cases these parameters depend on the data structure [7].

In this paper, we propose a multilayer batch learning growing neural gas (MBL-GNG) approach. MBL-GNG is a parameter-less unsupervised learning algorithm based on competitive topological learning. Furthermore, MBL-GNG has hierarchical and topological structures for providing multiscale results. MBL-GNG is based on GNG proposed by Fritzke [8], and can dynamically change the topological structure based on the adjacent relation (edge), referring to the activation frequency of the adjacent node according to the error index. Therefore, this type of topological learning algorithm is applied to 3D point cloud datasets measured from sensors for the perceptual system of intelligent autonomous robots [9, 10], gesture recognition [11], and knowledge extraction [12]. However, GNG has a problem related to learning convergence because GNG is an online learning method and the learning rates do not depend on the number of iterations. First, we propose batch learning GNG (BL-GNG) to improve the learning convergence and reduce the user-designed parameters. The learning method of BL-GNG is based on fuzzy C-means (FCM) because it has a local minimum convergence property [13, 14]. Next, we describe the MBL-GNG algorithm. In MBL-GNG, the input data of each layer uses parent nodes to learn more abstract topological structures from the dataset. Furthermore, MBL-GNG automatically determines the number of nodes in each layer using a one-nearest neighbor graph composed of the input data. During the experiments, we applied our proposed method to several datasets to study the effectiveness of the proposed method.

The remainder of this paper is organized as follows. Section 2 describes previous studies related to this research. Section 3 proposes BL-GNG, and Section 4 proposes MBL-GNG. Section 5 presents the experimental results. Finally, we summarize this paper and provide a discussion regarding the proposed method in Section 6.

2. Related Work

2.1. Previous Work

One of the most popular methods of unsupervised learning is the k-means method [15]. The k-means method has *k* prototypes (cluster centers), and the position of the prototypes is iteratively determined by calculating the center of data assigned to each prototype. The k-means method is used in many applications of data mining because it is easy to implement and has a local minimum convergence property. However, k-means has certain problems, such as robustness in the distribution of the dataset [16] and the definition of the number of prototypes. In the field of soft clustering, many probabilistic methods [17, 18] and FCM-type algorithms have been proposed [19, 20]. In many cases, soft clustering methods can improve the global convergence property compared to the k-means clustering method because the soft clustering method allows each input data to have some probabilities or grades in a cluster. However, these methods also have problems related to user-defined parameters because it is difficult for the user to define the number of prototypes or the probabilistic distribution for the unknown data distribution.

To solve this problem, many clustering methods have been proposed. These methods can be divided into two types. One is to determine the number of prototypes according to any criterion, such as X-means [21], which determines the number of nodes according to Akaike's information criterion or the Bayesian information criterion. The other is online incremental unsupervised learning such as a growing cell structure (GCS) [22], GNG, and self-organizing incremental neural network (SOINN) [23]. SOINN, proposed by Shen and Hasegawa, is one of the most effective learning methods in the field of competitive learning. SOINN has a high tolerance to noise and a high classification capability. In particular, load-balancing SOINN (LB-SOINN) significantly improves the learning stability and capability [24] by using a load that indicates the learning time of each node. However, it is difficult for these methods to provide clustering results of different resolutions because these methods are basically composed of only one or two layers. Furthermore, these methods have several user-defined parameters. For example, LB-SOINN requires six such parameters, and as previously mentioned, it is difficult for the user to define them.

To provide the multiscale results, a growing hierarchical self-organizing map (GHSOM) [25] and growing hierarchical tree SOM (GHTSOM) [26] were proposed. These methods combine hierarchical clustering and competitive learning methods. With these methods, a child layer is generated if a parent node satisfies the criterion related to the amount of node activity. By composing multiple layers, these methods can provide multiscale clustering results. In particular, GHTSOM is a parameter-less algorithm that defines several parameters empirically and shows effective clustering results. However, these methods generate redundant nodes and edges because their basic topological structure is predefined.

In this way, many unsupervised learning methods have been proposed to solve the problems related to unsupervised learning. In this study, we focus on hierarchical topological learning and a parameter-less algorithm. The contributions of this study are as follows:

1) Improving the learning convergence of the conventional GNG algorithm.
2) Providing the multiscale clustering results for giving the multiple viewpoints from the original dataset.
3) Proposing the parameter-less algorithm for improving the usability of the clustering algorithm.

Therefore, we propose MBL-GNG for extracting the features of the dataset from multiscale clustering results and reducing the user's load related to determining the parameters.

2.2. Growing Neural Gas

Self-organized maps (SOMs) [27], neural gas (NG) [28], GCS, and GNG are well-known unsupervised learning methods. These methods are based on competitive learning. The number of nodes and topological structure of the network in SOM were designed beforehand [27]. In NG, the number of nodes is fixed beforehand, but the topological structure is updated according to the distribution of the sample data. By contrast, GCS and GNG can dynamically change the topological structure based on the adjacent relation (edge), referring to the ignition frequency of the adjacent node according to the error index. However, GCS does not delete nodes and edges, whereas GNG can delete nodes and edges based on the concept of age [8, 22]. In this section, we explain the GNG learning algorithm to clarify our learning rule. In general, the learning rule of the GNG is calculated as follows:

$$\begin{cases} w_{s_1} \leftarrow w_{s_1} + \eta_1 \cdot (v_i - w_{s_1}), \\ s_1 = \underset{j \in W}{\arg\min} \left\| v_i - w_j \right\|, \end{cases} \quad \cdots\cdots \quad (1)$$

$$w_j \leftarrow w_j + \eta_2 \cdot (v_i - w_j) \quad \text{if } c_{s_1,j} = 1, \quad . . . \quad (2)$$

where $W = \{w_1, w_2, \ldots, w_M\}$ is the set of nodes, $V = \{v_1, v_2, \ldots, v_N\}$ is the set of input vectors, $c_{i,j}$ is an element of the adjacency matrix of a graph C, and η_1 and η_2 are the learning rates ($\eta_1 > \eta_2$) of GNG, respectively. Thus, the nodes are updated only if the node is the winner node s_1 or connects to s_1 in the GNG. We assume that these equations are derived using a gradient descent method. Based on this assumption, we can define the objective function of the GNG as follows:

$$\begin{cases} J_{\mathrm{GNG}} = \sum_{n=1}^{N}\sum_{k=1}^{M} r_{nk}\left\|v_n - w_k\right\|^2 + \sum_{n=1}^{N}\sum_{k=1}^{M} \mu_{nk}\left\|v_n - w_k\right\|^2, \\ r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j \left\|v_n - w_j\right\|^2, \\ 0, & \text{otherwise,} \end{cases} \\ \mu_{nk} = \begin{cases} \mu, & \text{if } c_{l,k} = 1 \quad \left(l = \arg\min_j \left\|v_n - w_j\right\|^2\right), \\ 0, & \text{otherwise,} \end{cases} \end{cases}$$

$$. . . . . (3)$$

where N is the number of input data, M is the number of nodes, μ ($0 \leq \mu < 1$) is a coefficient, and we assume that the adjacent relation is static. We calculated the following equation to solve the solution minimizing J_{GNG}:

$$\frac{\partial J_{\mathrm{GNG}}}{\partial w_k} = -\left\{\sum_{n=1}^{N} r_{nk}(v_n - w_k) + \sum_{n=1}^{N} \mu_{nk}(v_n - w_k)\right\} = 0.$$

$$. . . . . (4)$$

From Eqs. (3) and (4), the online update method of the gradient descent can be defined as follows:

$$\begin{aligned} w^{(t+1)} &= w^{(t)} - \eta \frac{\partial J_{\mathrm{GNG}}}{\partial w_k^{(t)}} \\ &= w^{(t)} + \eta r_{tk}\left(v_t - w_k^{(t)}\right) + \eta \mu_{nt}\left(v_t - w_k^{(t)}\right) \\ &= w^{(t)} + \eta_1'\left(v_t - w_k^{(t)}\right) + \eta_2'\left(v_t - w_k^{(t)}\right), \quad . (5) \end{aligned}$$

where t is the number of iterations, and η is the learning rate. Eqs. (1), (2), and (5) are equivalent because $\eta_1' > \eta_2'$ from $0 \leq \mu < 1$ and $\eta_2' \neq 0$ only if node w connects to the winner node. In this way, the learning rule of the GNG can be defined as the online gradient descent. Here, the first term on the right side of Eq. (3) represents the objective function of the k-means method. By contrast, the second term is considered the optimizing term of soft clustering, which is not clear in terms of soft clustering optimization. Although the learning rule has an unclear term, GNG outperforms other soft clustering methods in many cases [29]. This is because the GNG can dynamically change the topological structure. Therefore, we consider that redefining the objective function of GNG according to the point of view of soft clustering leads to an improvement in the learning capability of GNG.

3. Batch Learning Growing Neural Gas

In this chapter, we propose BL-GNG. As mentioned before, GNG is a type of online learning, and the learning rates do not depend on the number of iterations. Therefore, GNG has a problem related to the learning stability. To improve the learning stability, we change the online learning to batch learning from the viewpoint of fuzzy C-means (FCM).

3.1. Update Adjacent Relation

To realize the batch learning algorithm, it is important to avoid a dependence on the input order. In GNG, the update rules of the edge and node depend strongly on the input order. Therefore, we describe the update rule for the adjacent relation in BL-GNG.

In the update rule, BL-GNG uses a temporary adjacent relation $g_{i,j}$ of a temporary adjacency matrix G to create and remove the edge. In the initial step, all the temporary adjacent relations are set to zero. When input vector v_i is given, the temporary adjacent relation is updated as follows:

$$\begin{cases} g_{s_1,s_2} = 1, \\ s_1 = \arg\min_{j\in W} \left\|v_i - w_j\right\|, \\ s_2 = \arg\min_{j\in W\setminus\{s_1\}} \left\|v_i - w_j\right\|. \end{cases} \quad . . . . (6)$$

After all input vectors are given, all adjacent relations $c_{i,j}$ of the adjacency matrix are updated as follows:

$$\begin{cases} c_{i,j} = 1, & \text{if } g_{i,j} = 1, \\ c_{i,j} = 0, & \text{otherwise.} \end{cases} \quad . . . . (7)$$

In batch learning, the first and second winner nodes (s_1 and s_2) are calculated for all input vectors. The temporary adjacent relation is $g_{i,j} = 0$ if the nodes are not selected as s_1 and s_2. Therefore, such an edge $c_{i,j}$ is removed because there is no input vector between nodes w_i and w_j. Thus, BL-GNG updates the adjacent relationship.

3.2. Initialization of BL-GNG

In the GNG, the number of initial nodes was 2. In BL-GNG, it takes a long time to grow the number of required nodes because a new node is inserted after all input vectors are calculated. However, BL-GNG can use any number of initial nodes by using the update rule of the adjacent relation. **Algorithm 1** presents the detailed procedure. During initialization, the adjacency matrix C is set to zero, and M_{init} nodes are generated by randomly selecting the M_{init} input vector. After assigning all nodes, the update rule enables the initialization of BL-GNG to start from any number of nodes.

Algorithm 1 Initialization

```
Input: Input dataset $V$, number of initial nodes $M_{\text{init}}$
Output: Set of node $W$ and adjacency matrix $C$
 1: for $j = 1$ to $M_{\text{init}}$ do
 2:   Select the $k$-th input vector randomly
 3:   $w_j = v_k$
 4: end for
 5: for $j = 1$ to $N$ do
 6:   $s_1 = \arg\min_{i \in A} \|v_j - w_i\|$
 7:   $s_2 = \arg\min_{i \in A \setminus \{s_1\}} \|v_j - w_i\|$
 8:   $g_{s_1,s_2} = 1$
 9: end for
10: for $i = 1$ to $M_{\text{init}}$ do
11:   for $j = 1$ to $M_{\text{init}}$ do
12:     if $g_{i,j} = 1$ then
13:       $c_{i,j} = 1$
14:     else
15:       $c_{i,j} = 0$
16:     end if
17:   end for
18: end for
```

3.3. Learning Rule of BL-GNG

In BL-GNG, the node position w_i^* can be defined using Eq. (4) as follows:

$$w_i^* = \frac{\sum_{n=1}^{N} (r_{ni} + \mu_{nk}) v_n}{\sum_{n=1}^{N} (r_{ni} + \mu_{nk})}. \quad (8)$$

Thus, we can define the node position of batch learning from the learning rule of the GNG. However, as mentioned before, the objective function of GNG includes an unclear optimization term, and this term does not depend on the number of iterations. Therefore, the learning rule of GNG is unstable. The objective function of the BL-GNG is defined as follows:

$$J_{\text{BL-GNG}} = \sum_{n=1}^{N} \sum_{k=1}^{M} (u_{nk})^2 \|v_n - w_k\|^2, \quad (9)$$

where u_{nk} indicates the grade of a membership function for the k-th node of the n-th input vector. Eq. (9) has the same objective function as the basic FCM method because GNG can be considered an online version of the soft clustering method, and FCM has the local minimum convergence property [13, 14]. From Eqs. (8) and (9), the update rule of BL-GNG can be defined as follows:

$$w_i^{(t)} = \frac{\sum_{j=1}^{N} \left({u_{ji}}^2 \cdot v_j \right)}{\sum_{j=1}^{N} {u_{ji}}^2}. \quad (10)$$

The membership function of BL-GNG must satisfy the following conditions because the learning rule is based on basic FCM.

$$\begin{cases} u_{nk} \in [0,1]; & 1 \le n \le N; \quad 1 \le k \le M; \\ \sum_{i=1}^{M} u_{ni} = 1; & 1 \le n \le N; \\ \sum_{j=1}^{N} u_{jk} > 0; & 1 \le k \le M. \end{cases} \quad (11)$$

From these conditions, the membership function of BL-GNG is defined as follows:

$$u_{ji} = \frac{1}{\left(\frac{d_{i,j}}{d_{s_1,j}} + \sum_{c_{k,s_1}=1} \frac{d_{i,j}}{d_{k,j}} \right)}. \quad (12)$$

Equation (12) indicates that the membership function influences only the winner node and nodes with an adjacent relation to the winner node. In general, FCM has a fuzziness m, which is a user-defined parameter for controlling the influence of the nodes on the input vector. BL-GNG uses a fixed fuzziness value $m = 2$ because BL-GNG can control the influence of the nodes by changing the adjacent relations dynamically. In addition, Eq. (12) occasionally generates nodes that do not satisfy condition (11). These nodes are removed from the set of nodes, W. This procedure is similar to that of GNG-U [30], which is used for non-stationary datasets. GNG-U removes nodes whose utility value is lower than a user-defined threshold value.

At this point, we consider the membership function of BL-GNG. FCM influences all nodes for one input vector and updates the node positions. However, BL-GNG influences only the winner node and nodes with an adjacent relation to the winner node for one input data. This learning behavior can be considered a partial optimization for an input vector. Thus, the learning rule of BL-GNG determines the node positions by iterating the partial optimization.

3.4. Full Procedure of BL-GNG

In this section, we describe the complete BL-GNG algorithm. BL-GNG can be divided into two types. One involves calculating the membership function and the temporary adjacent relations. **Algorithm 2** shows the algorithm of local learning, where $H^1 = \{h_1^1, h_2^1, \ldots, h_M^1\}$ and $H^2 = \{h_1^2, h_2^2, \ldots, h_M^2\}$ are the variables for calculating the positions of the nodes using Eq. (10). In **Algorithm 2**, the accumulated error E_i is calculated using a criterion in the insertion algorithm of a new node. After all input vectors are calculated, all nodes and adjacency relations are updated (**Algorithm 3**).

Algorithm 4 shows the insertion algorithm of the new node, and **Algorithm 5** shows the total procedure of BL-GNG. The insertion algorithm is basically the same as that of the GNG. However, BL-GNG does not need to decrease the accumulated errors because the accumulated errors are set to zero before applying **Algorithm 2**.

Algorithm 2 Local learning

```
Input: Set of input data $V$, set of nodes $W$ and adjacency matrix $C$
Output: $H^1$, $H^2$, temporary adjacency matrix $G$ and the accumulated error $E$
1: Set all $h_j^1 = 0$ and $h_j^2 = 0$, $g_{i,j} = 0$, $E_i = 0$
2: for $j = 1$ to $N$ do
3:     $s_1 = \arg\min_{i \in W} \|v_j - w_i\|$
4:     $s_2 = \arg\min_{i \in W \setminus \{s_1\}} \|v_j - w_i\|$
5:     Calculate the membership functions and update the variables $h_i^1$ and $h_i^2$ of the winner node $s_1$ and the neighborhood of the $s_1$
         $h_i^1 \leftarrow h_i^1 + u_{ji}^2$
         $h_i^2 \leftarrow h_i^2 + u_{ji}^2 \cdot v_j$
6:     $g_{s_1,s_2} = 1$
7:     Update the accumulated error $E_{s_1}$
       $E_{s_1} \leftarrow E_{s_1} + \|v_j - w_{s_1}\|^2$
8: end for
```

Algorithm 3 Updating procedure

```
Input: $H^1$, $H^2$ and temporary adjacency matrix $G$
Output: Set of nodes $W$ and adjacency matrix $C$
1: for $i = 1$ to $M$ do
2:     Update the node $w_i$ as following;
       $w_i^{(t)} = \frac{h_i^2}{h_i^1}$
3:     for $j = 1$ to $M$ do
4:         Update the edge $c_{i,j}$ by calculating Eq. (7)
5:     end for
6: end for
```

Algorithm 4 Insertion algorithm of a new node (IA)

```
Input: Set of accumulated error $E$, set of nodes $W$ and adjacency matrix $C$
Output: Set of nodes
1: Select the unit $q$ with the maximal accumulated error
2: Select the unit $f$ with the maximal accumulated error among the neighbors of $q$
3: Insert a new unit $r$ halfway between $q$ and $f$
   $w_r = 0.5 \cdot (w_q + w_f)$
4: Insert edges $c_{q,r} = 1$, $c_{r,f} = 1$ and remove the edge $c_{q,f} = 0$
```

Algorithm 5 BL-GNG

```
Input: Set of input data $V$, the number of initial nodes $M_{\text{init}}$ and the maximum number of nodes $M_{\max}$
Output: Set of nodes $W$ and adjacency matrix $C$
1: $M = M_{\text{init}}$
2: Initialization $(V)$
3: while the stopping criterion is not reached do
4:     $(H^1, H^2, G, E)$ = Local learning $(V, W, C)$
5:     $(W', C)$ = Update procedure $(H^1, H^2, G)$
6:     if $M \leq M_{\max}$ then
7:         $(W, C)$ = IA $(E, W, C)$
8:         $M \leftarrow M + 1$
9:     end if
10: end while
```

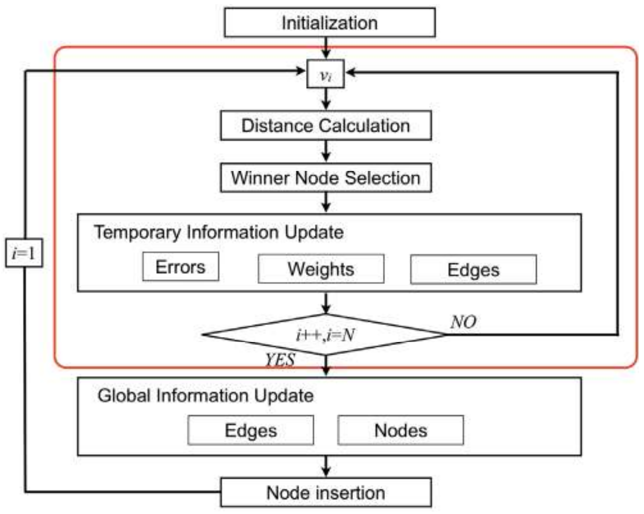


Fig. 2. Flowchart of BL-GNG.

Table 1. Feature of BL-GNG compared to GNG.

	GNG	BL-GNG
Initial node number	2	User-defined
Convergence	Low	High
Computational cost	Constant	Depending on number of data
Required memory	Small	Large
Parameters	7	2

In addition, BL-GNG does not need a parameter λ that is used as a criterion for the insertion algorithm because all input vectors are calculated in **Algorithm 2**. Therefore, the insertion algorithm was conducted using **Algorithm 3**. Thus, BL-GNG can learn the topological structure.

Figure 2 shows a flowchart of the BL-GNG. In addition, **Table 1** shows the features of BL-GNG compared to those of GNG. In **Table 1**, the convergence indicates the number of update nodes as the iterations progress. In GNG, the number does not converge because the learning process strongly depends on the input order, and the learning rate does not depend on the number of iterations. Therefore, the convergence was low, as shown in **Table 1**. Next, we considered the number of user-defined parameters. GNG requires seven parameters: the learning rate, the maximum age of edge $a_{\max}$, integer multiplier λ, two decreasing rates of the accumulated error α, β, and the maximum number of nodes $M_{\max}$. By contrast, BL-GNG requires only two user-defined parameters: the number of initial nodes M_{init} and the maximum number of nodes. Thus, BL-GNG has several advantages over GNG.

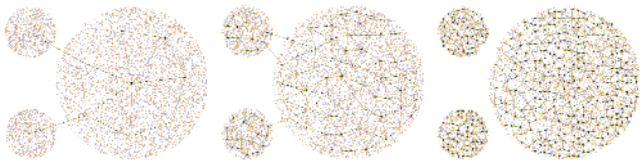


Fig. 3. Examples of BL-GNG. Green and red dots indicate input data and nodes, respectively. A red line indicates an edge.

4. Multilayer BL-GNG

BL-GNG is proposed to improve the learning convergence and reduce the user-defined parameters compared to GNG. However, BL-GNG needs to define the number of nodes for application to the unknown data distribution, as with other methods. It is difficult for users to define the number of nodes because it depends on the situation or application. **Fig. 3** shows examples of BL-GNG. These results show that the clustering results strongly depend on the number of nodes. In this way, BL-GNG has a problem of defining the number of nodes for the unknown data distribution according to the situation. In this chapter, we propose a multilayer BL-GNG (MBL-GNG). MBL-GNG has the results of different resolutions, and the number of nodes in each layer is defined automatically.

4.1. Decision Method of the Max Number of Nodes

In GNG, the topological structure with the maximum number of clusters is considered to be a 1-nearest neighbor graph [31]. In the 1-nearest neighbor graph, the distribution of clusters approximates the data distribution because each cluster is composed of the minimum element of the topological structure. Therefore, the number of nodes in the i-th layer is defined as the number of clusters of the 1-nearest neighbor graph composed of all input vectors of the i-th layer. **Algorithm 6** shows the detailed procedure, where D is the diagonal matrix whose diagonal elements D_{ii} indicates the number of edges of node w_i. In **Algorithm 6**, because the Laplacian has k zero eigenvalues if the graph has k clusters, the number of clusters is calculated by counting the number of zero eigenvalues of a Laplacian L [32].

4.2. Full Procedure of MBL-GNG

Algorithm 7 shows the complete procedure of MBL-GNG. As mentioned before, the number of initial nodes of BL-GNG can be any number. However, the result of BL-GNG depends on the initial positions of the nodes because the initial nodes are randomly assigned from the set of input vectors. Therefore, the number of initial nodes of MBL-GNG is a set to two nodes ($M_{\text{init}} = 2$) because the influence of the initial state can be reduced. MBL-GNG learns the topological structure of the $(l+1)$-th layer by using a set of input vectors composed of the set of l-th layer nodes. By doing so, MBL-GNG can learn more abstract topological structures from

Algorithm 6 Decide max number of nodes (DM)

```
Input: Set of input data $V^{(l)}$
Output: Maximum number of nodes in the $l$-th layer
 1: for $i = 1$ to $N$ do
 2:     $s_1 = \underset{j \in V^l \setminus i}{\arg\min} \left\| v_i - v_j \right\|$
 3:     $c_{s_1,i} = 1$
 4: end for
 5: Calculate graph Laplacian $L$
    $L = D - C$
 6: Calculate $N$ eigenvalues $\lambda$ from Laplacian $L$
 7: for $i = 1$ to $N$ do
 8:     if $\lambda_i = 0$ then
 9:         $M_{\max} \leftarrow M_{\max} + 1$
10:     end if
11: end for
```

Algorithm 7 MBL-GNG

```
Input: Set of input data $V^1$
 1: The number of layer $l = 1$ and the number of clusters $f^l$ sets
    to 0
 2: while $f^l \neq 1$ do
 3:     $M^l_{\max} = \mathrm{DM}(V^l)$
 4:     $(W^l, C^l) = \text{BL-GNG}(V^l, 2, M^l_{\max})$
 5:     Calculate Laplacian L
        $L = D - C$
 6:     Calculate the $M^l_{\max}$ eigenvalues $\lambda$ of $L$
 7:     for $i = 1$ to $M^l_{\max}$ do
 8:         if $\lambda_i = 0$ then
 9:             $f^l \leftarrow f^l + 1$
10:         end if
11:     end for
12:     Increase the layer number
        $l \leftarrow l + 1$
13:     $V^l = \{W^{(l-1)}\}$
14: end while
```

the original dataset as the number of layers increases. As the terminal criterion of MBL-GNG, the number of clusters f^l in the l-th layer's BL-GNG is 1. This terminal criterion is the same as that in general hierarchical clustering methods [1, 33]. In this way, MBL-GNG automatically determines the number of nodes and layers and learns the topological structures of the different resolutions.

5. Experimental Results and Discussion

5.1. Comparison Between BL-GNG and GNG

In this section, we describe an experiment conducted to show the learning capability of BL-GNG in comparison with GNG. In this experiment, we used a 2D plot dataset composed of three circles (3C) (**Fig. 4**), the centers and radii of which were C1 = $(x, y, r) = (6.0, 6.0, 0.5)$, C2 = $(6.0, 8.0, 0, 5)$, and C3 = $(9.0, 7.0, 1.5)$, respectively. The plot data were generated by a uniform random num-

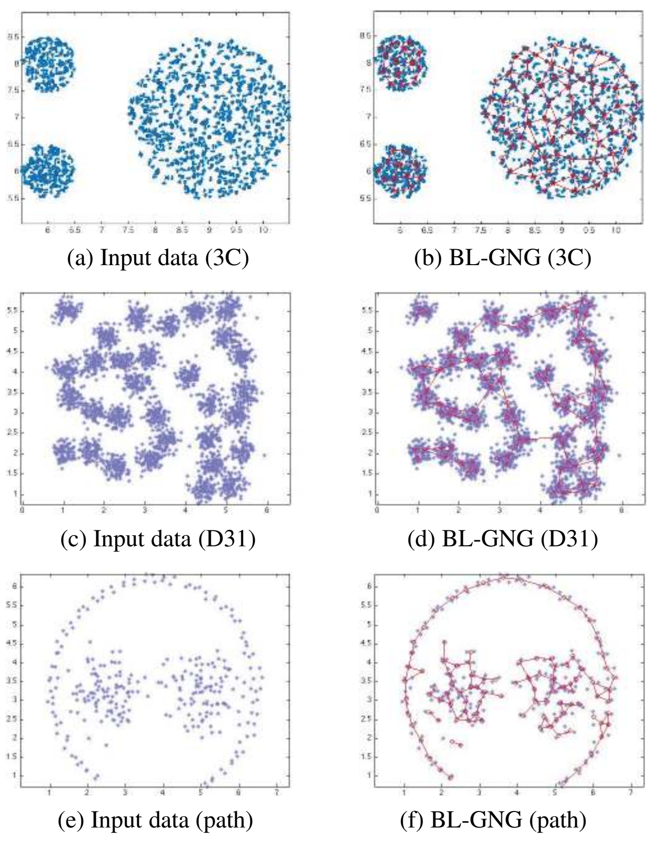
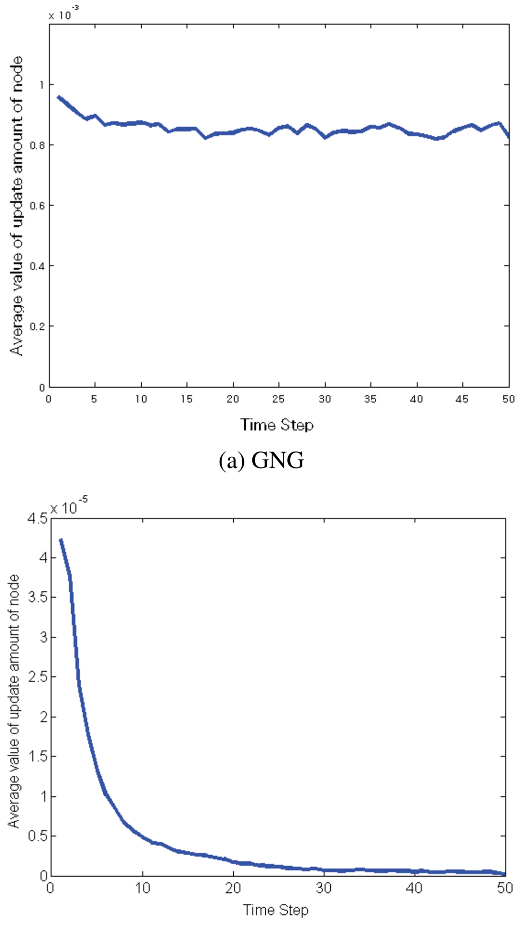

(a) Input data (3C) (b) BL-GNG (3C)
(c) Input data (D31) (d) BL-GNG (D31)
(e) Input data (path) (f) BL-GNG (path)

Fig. 4. Examples of a topological structure of BL-GNG with $M_{\text{init}} = 2$ and $M_{\max} = 100$. Blue and red dots indicate the input data and node, respectively. The red line indicates an edge.



(a) GNG
(b) BL-GNG

Fig. 5. Experiment results of average update value of the nodes. The time step is the number of iterations after $M_{\max}$ nodes are generated.

ber placed into each circle. The numbers of data for C1, C2, and C3 were 200, 200, and 300, respectively. Furthermore, we used benchmark tests such as D31 [34] and path-based (Path) tests [35]. **Fig. 5** displays a two-dimensional (2D) plot of each dataset. In BL-GNG, we defined $M_{\text{init}} = 2$ and $M_{\max} = 100$ as the parameters. In GNG, we defined $\eta_1 = 0.08$, $\eta_2 = 0.008$, $a_{\max} = 200$, $\alpha = 0.5$, $\beta = 0.995$, $\lambda = 1200$. The integer multiplier λ and the maximum age of the edge $a_{\max}$ were defined for the same conditions as BL-GNG, and the other parameters were defined empirically. In this experiment, the number of trials was 100 for each method. To evaluate the convergence of each method, we used the following evaluation criteria:

$$\begin{cases} U = \dfrac{\sum_{k=1}^{M} \|\Delta w_k\|^2}{M}, \\ \Delta w_k = w_k^t - w_k^{(t-1)}, \end{cases} \quad \dots\dots\dots\dots (13)$$

where w_k^t is the position of the k-th node at the t-th iteration, and U is the average update amount of the nodes. Therefore, the criterion is low as learning grows if the learning method is stable. In addition, we used the objective function of the k-means method to accurately evaluate the learning results of the topological structure.

$$\begin{cases} E = \sum_{n=1}^{N}\sum_{k=1}^{M} r_{nk} \|v_n - w_k\|^2, \\ r_{nk} = \begin{cases} 1, & \text{if } k = \arg\min_j \|v_n - w_j\|^2, \\ 0, & \text{otherwise.} \end{cases} \end{cases} \quad (14)$$

Figure 4 plots the average update amount using 3C dataset after the number of nodes M reaches the maximum number of nodes $M_{\max}$. In **Fig. 5(a)**, GNG did not converge because the average update amount of GNG had constant value after M reaches $M_{\max}$. This result shows that the topological learning of GNG depends on the order of input vectors since the learning rate of GNG does not depend on the number of iteration. On the other hand, BL-GNG converged to 0 (**Fig. 5(b)**) though the membership function of BL-GNG also dose not depend on the number of iteration. Because we consider that BL-GNG has the local minimum convergence property by calculating the node positions from all input vectors. Furthermore, the error of BL-GNG is almost less than that of GNG in all datasets (**Table 2**). From these results,

Table 2. Experimental results of error value.

		BL-GNG	GNG
3C	Average	14.10	15.27
	Variance	0.012	0.137
D31	Average	55.32	63.12
	Variance	0.13	0.83
Path	Average	3.71	4.19
	Variance	0.009	0.021

BL-GNG improves the learning stability of GNG as the learning accuracy of GNG remains. In addition, the user-defined parameters of BL-GNG can be reduced. In this way, BL-GNG is easy to use algorithm method compared with GNG.

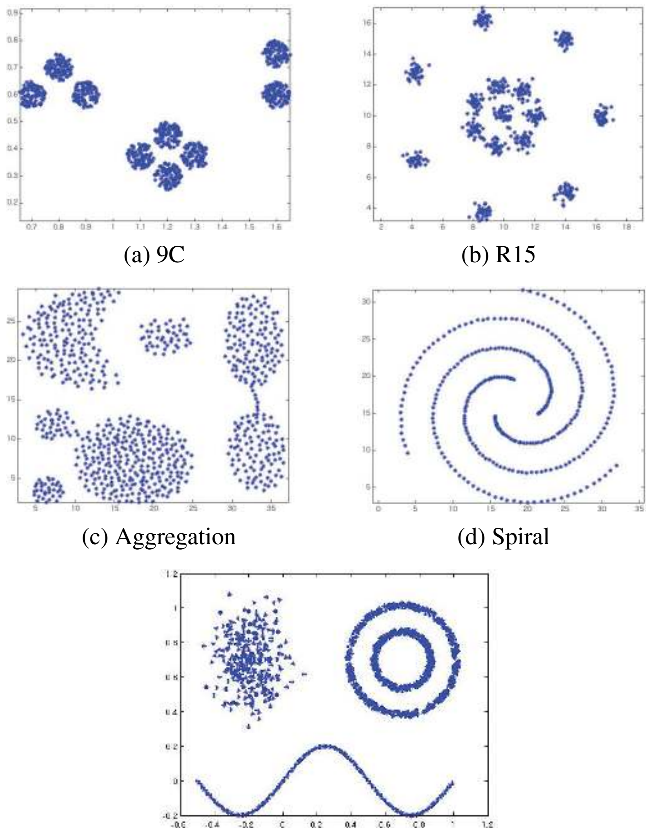

(a) 9C (b) R15 (c) Aggregation (d) Spiral (e) Artificial Dataset

Fig. 6. Data distribution of each dataset.

5.2. Performance Evaluation of MBL-GNG Using 2D Plot Data

We conducted experiments to evaluate MBL-GNG. In this experiment, we used 2D point cloud data because one of the main applications in topological clustering is learning the point cloud data measured through visual sensors used in the perceptual systems of autonomous robots [9–12]. We used the 9 circles (9C) dataset, whose centers and radii are C1 = $(0.7,0.6,0.05)$, C2 = $(0.8,0.7,0.05)$, C3 = $(0.9,0.6,0.05)$, C4 = $(1.6,0.6,0.05)$, C5 = $(1.6,0.75,0.05)$, C6 = $(1.2,0.3,0.05)$, C7 = $(1.2,0.45,0.05)$, C8 = $(1.1,0.375,0.05)$, and C9 = $(1.3,0.375,0.05)$. The plot data were generated by a uniform random number into each circle. The number of data in each circle is 100 (the total number of input data is 900). In addition, we used another artificial dataset (AD) composed of one Gaussian distribution (300 data points), two rings (900 data points), and one sine curve (150 data points). Furthermore, we used benchmark tests such as R15 [34], spiral (Sp) [36], and aggregation (Agg) [35]. **Fig. 6** shows a 2D plot of each dataset.

MBL-GNG can cluster datasets from two different viewpoints. One is to use nodes, such as the prototypes of the k-means method (node division). The other is to use the topological structure of each layer (cluster division). In this experiment, we compared these clustering methods to determine which is better for MBL-GNG. In addition, we compared MBL-GNG to other hierarchical topological clustering methods, namely, the growing hierarchical self-organizing map (GHSOM) and growing hierarchical probabilistic self-organizing graph (GHPSOG) to verify the clustering capability of MBL-GNG. The parameters τ_1 and τ_2 of GHSOM used 0.8 and 0.01, respectively. A GHPSOG model has a hierarchical architecture composed of a dynamic graph and is based on a probabilistic mixture of multivariate Gaussian components [37]. GHPSOG can also change the graphs dynamically according to the data distribution in the same manner as MBL-GNG. Therefore, we used GHPSOG for a comparison with our proposed method, and the parameter τ of GHPSOG used a value of 0.01. In this experiment, the number of trials was 100 for each method. To compare these methods, we used the Calinski–Harabasz (CH) [38] and Davies–Bouldin (DB) [39] criteria for the evaluation. CH is used to evaluate the optimal number of clusters with the maximum CH index value. DB is used to evaluate the cohesion within a cluster and the discreteness of the between clusters because the DB index value is based on the ratio of between clusters and within cluster distances. A lower DB index value indicates that the clustering result is better. Using this method, we verified the clustering capability of MBL-GNG from two different viewpoints.

First, **Tables 3** and **4** show the results of the number of nodes and clusters in each layer after the topological structure of each layer is learned. MBL-GNG can learn the multiscale topological structures in each dataset because the number of layers in each result is determined according to the data distribution. In MBL-GNG, the number of layers depends on the number of nodes in each layer, and the number of nodes is determined by a 1-nearest neighbor graph composed of the input dataset. Therefore, the variances of the number of nodes in first layer are zero (**Table 3**). By contrast, the variances of the other layers are not zero because two initial nodes are generated randomly, and the final results of the topological structure depend on the initial state of the nodes. However, the variances are not large because the learning method is based on the batch-learning method. Therefore, we consider that the influence of the initialization is

Table 3. Number of nodes in each layer.

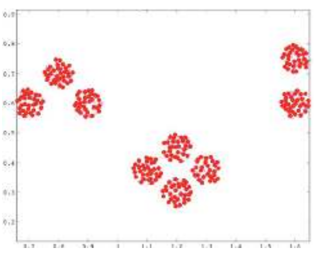
		1st	2nd	3rd	4th	5th
9C	Ave	283.0	87.2	27.7	9.4	3.2
	Var	0.00	3.85	2.46	0.36	0.29
R15	Ave	178.0	40.25	15.0	3.2	–
	Var	0.00	2.55	0	0.46	–
Agg	Ave	243.0	72.2	22.9	6.9	3.3
	Var	0	8.22	3.32	1.44	0.25
Sp	Ave	113.0	30.5	–	–	–
	Var	0.00	4.11	–	–	–
AD	Ave	381.0	106.3	27.1	9.0	3.0
	Var	0.00	4.31	2.95	1.06	0.40

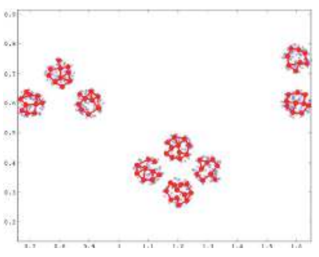
Table 4. Number of clusters in each layer.

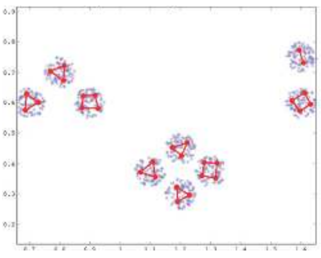
		1st	2nd	3rd	4th	5th
9C	Ave	11.3	9.0	8.7	2.9	1.0
	Var	0.96	0.02	0.39	0.08	0.00
R15	Ave	12.2	9.6	1.2	1.0	–
	Var	0.54	0.93	0.13	0.00	–
Agg	Ave	5.0	5.0	2.7	1.1	1.0
	Var	0.04	0.04	1.44	0.05	0.00
Sp	Ave	3.0	1.0	–	–	–
	Var	0.00	0.00	–	–	–
AD	Ave	20.8	4.3	3.0	1.8	1.0
	Var	2.86	0.23	0.02	0.78	0.00

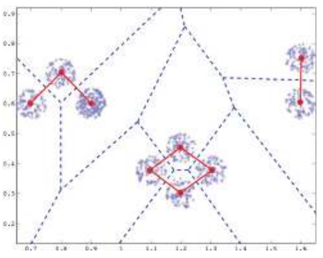
not terribly strong and that MBL-GNG can determine the number of layers stably according to each data distribution.

Next, **Figs. 7** and **8** show examples of the learning results of MBL-GNG in 9C and AD, respectively. In the result of 9C, the learning results of the second and third layers are considered almost the same from the viewpoint of clustering because the number of clusters is approximately nine (**Table 4**). However, the results of the topological structure are quite different because the number of nodes in each layer is different (**Table 3**). The topological structure of the second layer in 9C represents the detailed data distribution of each circle. However, the topological structure of the third layer indicates that each circle is composed of primitive shapes such as lines, triangles, or squares. In addition, even if the dataset has a different probability density distribution, such as AD, MBL-GNG can provide natural clusters according to different scales. For example, the components of the two rings are divided into different numbers of clusters in the second and third layers (**Figs. 8(b)** and **(c)**). In general, hierarchical topological learning methods such as GHSOM take a top-down approach, which means that the higher layer has a more detailed topological structure. However, MBL-GNG is a bottom-up approach because the input dataset of the $(t+1)$-th layer uses the topological structure of the t-th layer. Therefore, the topological structure of the $(t+1)$-th layer is a more abstract result than that of


(a) 1st layer


(b) 2nd layer


(c) 3rd layer


(d) 4th layer

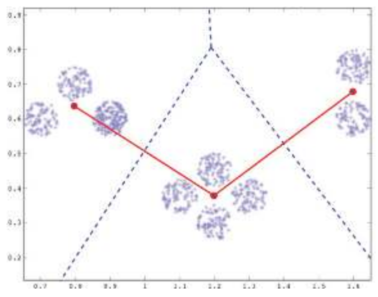
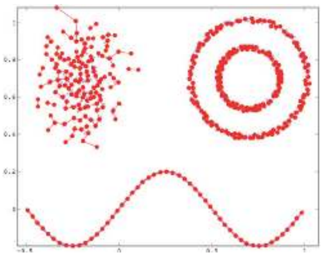
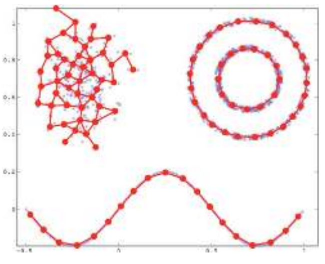
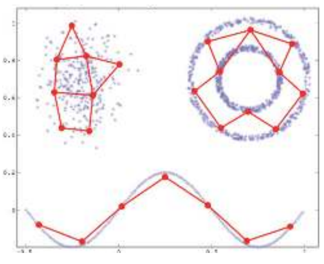

(e) 5th layer

Fig. 7. Results of MBL-GNG in 9C dataset. The blue and red dots indicate input data and nodes, respectively. The red line indicates the edge of MBL-GNG. In (d) and (e), blue dashed line indicates Voronoi edge.


(a) 1st layer


(b) 2nd layer


(c) 3rd layer

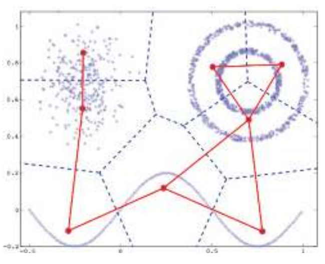

(d) 4th layer

Fig. 8. A result of MBL-GNG in AD. In (d), blue line indicates Voronoi edge.

the t-th layer. Thus, MBL-GNG can learn different scale topological structures in each layer.

In addition, the topological structures do not generate redundant nodes and edges because BL-GNG can dynamically change the topological structure according to the winner node and the second winner node for the input data. By contrast, the node positions of the highest layer do not exist in the data for both results (**Figs. 7(e)** and **8(d)**) because the higher layer represents a more primitive data distribution by using a small number of nodes, and the Voronoi edges can be correctly divided into clusters (**Figs. 7(d)**, **(e)**, and **Fig. 8(d)**). In addition, the edges indicate the relationship based on the Euclidean distance between each data distribution. However, there are no edges between the two rings and the Gaussian component in **Fig. 8(d)**, although the distances between the three components are almost the same. The most primitive topological structure of the Gaussian component is considered a vertical line in the AD because the data distribution of the Gaussian component is vertically long, and the distance between the Gaussian component and the two-ring component is longer than that between the sine curve component and each of the other components. Therefore, there are no edges between the two components. In addition, we provide another example of the highest layer in AD that changes the standard deviation of the Gaussian component into a horizontal length (**Fig. 9(a)**). In this result, the most primitive topology of the Gaussian component is represented as a horizontal line, and all components have a relation. From these results, the MBL-GNG can learn topological structures according to the data distributions. Furthermore, we consider that MBL-GNG can learn the relation between data sources according to the situation or application by selecting a suitable metric space.

However, datasets with overlapping areas, such as R15, sometimes generate different topological structures in the same layer. **Fig. 10** shows examples of topological structures obtained using R15. In **Fig. 10(a)**, MBL-GNG can learn the almost correct data distribution by using 15 nodes in the third layer. In this case, the precision of the third layer was 0.9917 (the numbers of true and false positives were 595 and 5, respectively). However, the MBL-GNG cannot learn the correct data distribution, as shown in **Fig. 10(b)**. As mentioned before, the topological structure of MBL-GNG generates two nodes at random during the initialization, and the topological structure depends on the initial state of the current layer and the final state of the parent layer. However, the number of third layers in the R15 dataset was always 15. Furthermore, we believe that MBL-GNG can improve the learning stability by improving the initialization, such as by using the k-means method [40].

Next, **Figs. 11–14** and **Table 5** show the results of the CH and DB index values in each layer. In these figures, we cannot plot the cluster results of the highest layer because the number of clusters is one in the highest layer. Therefore, these index values cannot be calculated. In the input dataset without overlapping areas such as C9 and

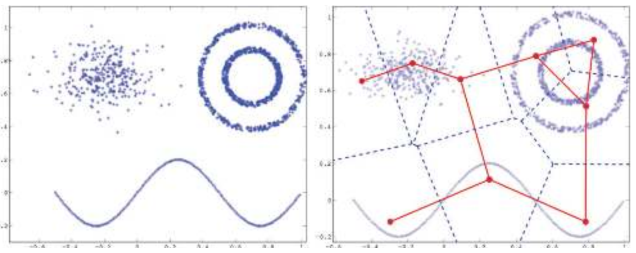
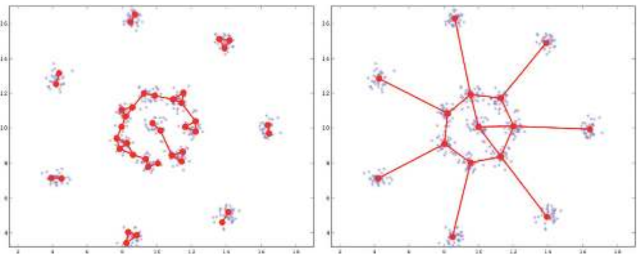

(a) Dataset (b) 4th layer

Fig. 9. A result of MBL-GNG in AD changing the standard deviation of Gaussian component.


(a) A successful example

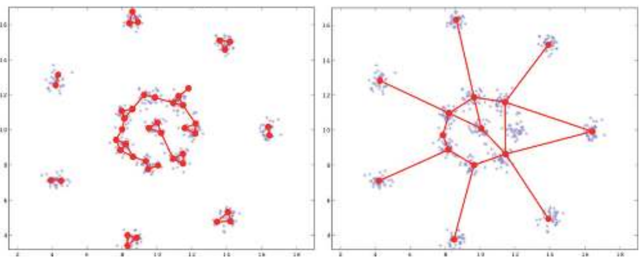
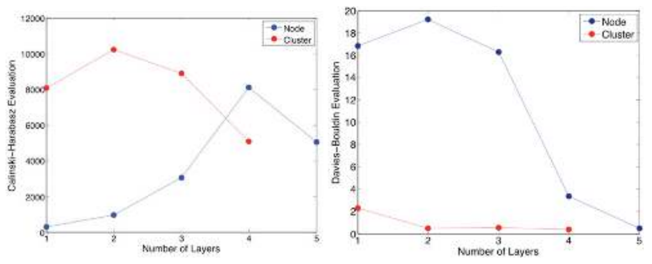

(b) A false example

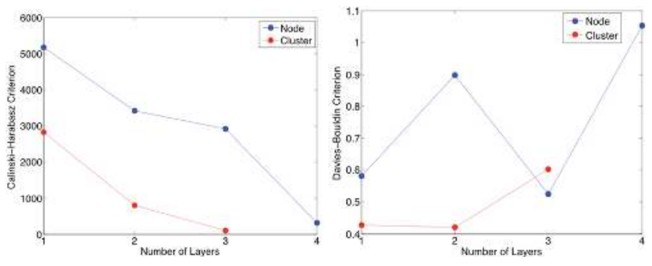
Fig. 10. Results of MBL-GNG in R15. Left side figures are the topological structures of the 2nd layer and right side figures are that of the 3rd layer.

AD (**Figs. 11** and **14**), the index values of clusters are better than those of the nodes because MBL-GNG can divide into a suitable number of clusters stably in those datasets. In 9C, the best number of clusters is nine because the layers with the highest value of CH are the second layer in the cluster division and the fourth layer in the node division, and the data distribution in each cluster is almost the same. By contrast, the layers with the lowest DB in 9C are the fourth layer in cluster division and the fifth layer in node division because the ratio of the between cluster and within cluster distance is the best balance when the number of clusters of MBL-GNG is 3. In addition, the index values of the nodes are better than those of the clusters in the input dataset with overlapped areas such as R15 and Agg. In addition, the layers that have the best index value of each index value are different (**Figs. 12** and **13**). Furthermore, the results of SP (**Table 3**) show that both index values of the node division are better than those of the cluster division. However, the number of clusters in the first layer is always three, and we consider this result to be natural (**Fig. 15**). Thus, the suitable layer and division



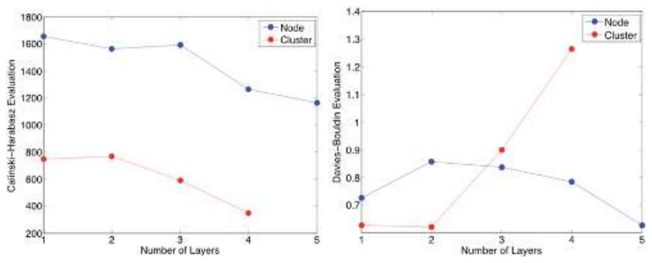
(a) Calinski–Harabasz Criterion (b) Davies–Bouldin Criterion

Fig. 11. Evaluation results of C9.



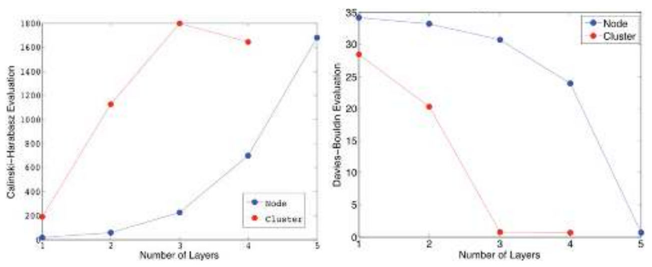
(a) Calinski–Harabasz Criterion (b) Davies–Bouldin Criterion

Fig. 12. Evaluation results of R15.



(a) Calinski–Harabasz Criterion (b) Davies–Bouldin Criterion

Fig. 13. Evaluation results of Agg.



(a) Calinski–Harabasz Criterion (b) Davies–Bouldin Criterion

Fig. 14. Evaluation results of AD.

differ according to the criteria. Therefore, it is important to select a suitable criterion for utilizing MBL-GNG effectively if one clustering result must be selected from the multiscale topological structures automatically.

In addition, **Tables 6** and **7** present the comparison results. In these tables, the index values are the best index values for all layers. MBL-GNG outperforms other methods, excluding the DB index value of 9C from the viewpoint of the clustering capabilities. Even the result of

Table 5. Evaluation results of SP.

		1st	2nd
CH	Node	994.46	321.06
	Cluster	5.80	–
DB	Node	0.476	0.604
	Cluster	5.882	–

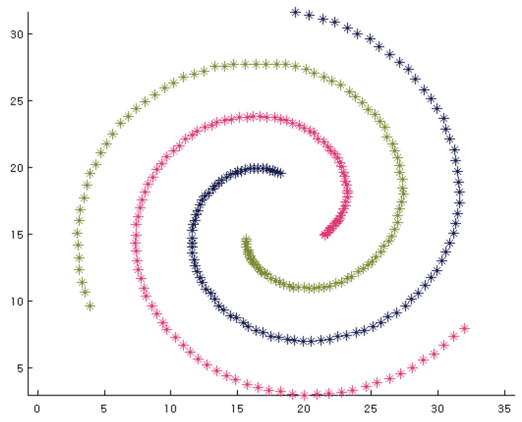


Fig. 15. Clustering result of SP in the 1st layer. Different clusters have different colors.

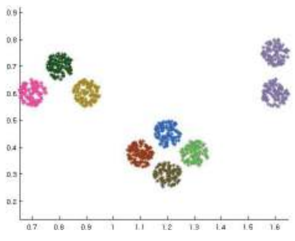
Table 6. Evaluation results of Calinski–Harabasz Criterion.

	Cluster	Node	GHSOM	GHPSOG
9C	8120.66	**10234.46**	6081.70	4370.00
R15	**5169.45**	2824.72	327.72	272.21
Agg	**1657.40**	767.07	1164.27	926.79
Sp	**994.46**	5.80	260.58	169.46
AD	1679.88	**1797.15**	776.88	954.67

Table 7. Evaluation results of Davies–Bouldin Criterion.

	Cluster	Node	GHSOM	GHPSOG
9C	0.477	0.390	**0.377**	2.310
R15	0.525	**0.420**	1.002	1.283
Agg	0.628	**0.621**	0.631	0.810
Sp	**0.476**	5.882	0.847	25.085
AD	**0.651**	0.688	17.483	1.219

MBL-GNG in 9C is almost the same as that of GHSOM. In GHSOM, some redundant nodes and edges are generated because the topological structure is always rectangular. By contrast, the GHPSOG model assumes a probabilistic mixture of multivariate Gaussian components. In this experiment, almost none of the data sources in each dataset were based on the Gaussian components. Therefore, these methods are worse than those of MBL-GNG. In addition, it is difficult to determine the user-defined parameters. **Fig. 16** shows examples of the highest layer of the GHSOM using different parameters. The clustering results were quite different. However, MBL-GNG can change the topological structure for any data distribution


(a) $\tau_1 = 0.8$, $\tau_2 = 0.01$

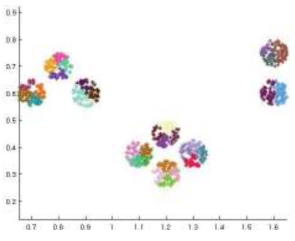

(b) $\tau_1 = 0.8$, $\tau_2 = 0.001$

Fig. 16. Examples of the clustering result of GHSOM using different parameters. In each figure, different clusters have different color. In (a), the number of clusters is 8. In (b), the number of clusters is 30.

in each layer without requiring any user-defined parameters. From these results, we consider that the clustering capabilities of MBL-GNG are higher than those of the other methods.

6. Conclusion

In this paper, we proposed a BL-GNG approach for improving the learning stability of GNG. The objective function of BL-GNG uses an FCM-type objective function because this type of objective function has a local convergence property. By changing online learning to batch learning, BL-GNG can reduce the user-defined parameters compared to GNG. Next, we proposed a MBL-GNG for learning the topological structures from the different scales of a dataset. MBL-GNG generates multiscale topological structures and determines the number of nodes and layers automatically without user-defined parameters. Several experiments were conducted to evaluate our proposed method. We demonstrated that BL-GNG has a local convergence property compared to GNG, and that MBL-GNG outperforms GHSOM and GHPSOG from the viewpoint of CH and DB index values.

However, MBL-GNG has certain limitations, such as the computational cost and dependency of the initial state. The MBL-GNG is based on a batch learning method. Therefore, the computational cost increases as the number of data increases, and it is difficult to apply MBL-GNG to big data. Furthermore, MBL-GNG cannot control the number of layers according to the situation or application because the number of nodes in each layer is determined by the 1-nearest neighbor graph. However, we consider that the number of layers can be controlled using a k-nearest neighbor graph. Therefore, to verify the effectiveness of MBL-GNG, we will improve these limitations and apply real-world applications, such as analyzing a 3D point cloud.

Acknowledgements

This work was partially supported by JSPS KAKENHI Grant Number 20K19894.

References:

[1] R. O. Duda, P. E. Hart, and D. G. Stork, "Pattern Classification," 2nd edition, John Wiley & Sons, 2012.

[2] D. V. Prokhorov, L. A. Feldkamp, and T. M. Feldkamp, "A new approach to cluster-weighted modeling," Int. Joint Conf. on Neural Networks Proc. (IJCNN'01), 2001.

[3] F. Nie, Z. Zeng, I. W. Tsang, D. Xu, and C. Zhang, "Spectral Embedded Clustering: A Framework for In-Sample and Out-of-Sample Spectral Clustering," IEEE Trans. on Neural Networks, Vol.22, No.11, pp. 1796-1808, 2011.

[4] H. Truong, L. Ngo, and L. Pham, "Interval Type-2 Fuzzy Possibilistic C-Means Clustering Based on Granular Gravitational Forces and Particle Swarm Optimization," J. Adv. Comput. Intell. Intell. Inform., Vol.23, No.3, pp. 592-601, 2019.

[5] Y. Hamasuna, S. Nakano, R. Ozaki, and Y. Endo, "Cluster Validity Measures Based Agglomerative Hierarchical Clustering for Network Data," J. Adv. Comput. Intell. Intell. Inform., Vol.23, No.3, pp. 577-583, 2019.

[6] M. Higashi, T. Kondo, and Y. Kanzawa, "Fuzzy Clustering Method for Spherical Data Based on q-Divergence," J. Adv. Comput. Intell. Intell. Inform., Vol.23, No.3, pp. 561-570, 2019.

[7] D. Heinke and F. H. Hamker, "Comparing neural networks: a benchmark on growing neural gas, growing cell structures, and fuzzy ARTMAP," IEEE Trans. on Neural Networks, Vol.9, No.6, pp. 1279-1291, 1998.

[8] B. Fritzke, "A growing neural gas network learns topologies," Advances in Neural Information Processing Systems 7: Proc. of the 1994 Conf., pp. 625-632, MIT Press, 1995.

[9] M. Saroya, G. Best, and G. A. Hollinger, "Roadmap Learning for Probabilistic Occupancy Maps With Topology-Informed Growing Neural Gas," IEEE Robotics and Automation Letters, Vol.6, No.3, pp. 4805-4812, 2021.

[10] A. A. Saputra, W. H. Chin, Y. Toda, N. Takesue, and N. Kubota, "Dynamic density topological structure generation for real-time ladder affordance detection," 2019 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 3439-3444, 2019.

[11] N. Mirehi, M. Tahmasbi, and A. T. Targhi, "Hand gesture recognition using topological features," Multimedia Tools and Applications, Vol.78, No.10, pp. 13361-13386, 2019.

[12] E. Ventocilla, R. M. Martins, F. Paulovich, and M. Riveiro, "Scaling the Growing Neural Gas for Visual Cluster Analysis," Big Data Research, Vol.26, Article No.100254, 2021.

[13] J. C. Bezdek, "A Convergence Theorem for the Fuzzy ISODATA Clustering Algorithms," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.PAMI-2, No.1, pp. 1-8, 1980.

[14] R. J. Hathaway and C. B. James, "Recent convergence results for the fuzzy c-means clustering algorithms," J. of Classification, Vol.5, pp. 237-247, 1988.

[15] J. MacQueen, "Some methods for classification and analysis of multivariate observations," Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability, Volume 1, pp. 281-297, 1967.

[16] J. Liang, L. Bai, C. Dang, and F. Cao, "The K-Means-Type Algorithms Versus Imbalanced Data Distributions," IEEE Trans. on Fuzzy Systems, Vol.20, No.4, pp. 728,745, 2012.

[17] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," Machine Learning, Vol.39, Issue 2-3, pp. 103-134, 2000.

[18] S. Goldwater and G. Tom, "A fully Bayesian approach to unsupervised part-of-speech tagging," Proc. of the 45th Annual Meeting of the Association of Computational Linguistics, Vol.45, No.1, pp. 744-751, 2007.

[19] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition. I," IEEE Trans. on Systems, Man, and Cybernetics, Part B, Vol.29, No.6, pp. 778-785, 1999.

[20] D. T. Anderson, J. C. Bezdek, M. Popescu, and J. M. Keller, "Comparing Fuzzy, Probabilistic, and Possibilistic Partitions," IEEE Trans. on Fuzzy Systems, Vol.18, No.5, pp. 906-918, 2010.

[21] D. Pelleg and A. W. Moore, "X-means: Extending K-means with Efficient Estimation of the Number of Clusters," Proc. of the 17th Int. Conf. on Machine Learning (ICML'00), pp. 727-734, 2000.

[22] B. Fritzke, "Growing cell structures – a self-organizing network for unsupervised and supervised learning," Neural Networks, Vol.7, No.9, pp. 1441-1460, 1994.

[23] F. Shen and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning," Neural Networks, Vol.19, No.1, pp. 90-106, 2006.

[24] H. Zhang, X. Xiao, and O. Hasegawa, "A Load-Balancing Self-Organizing Incremental Neural Network," IEEE Trans. on Neural Networks and Learning Systems, Vol.25, No.6, pp. 1096-1105, 2014.

[25] A. Rauber, D. Merkl, and M. Dittenbach, "The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data," IEEE Trans. on Neural Networks, Vol.13, No.6, pp. 1331-1341, 2002.

[26] A. Forti and G. L. Foresti, "Growing Hierarchical Tree SOM: An unsupervised neural network with dynamic topology," Neural networks, Vol.19, No.10, pp. 1568-1580, 2006.

[27] T. Kohonen, "Self-organizing maps," Springer Series in Information Sciences book series, Vol.30, Springer Science & Business Media, 2001.

[28] T. Martinetz and K. Schulten, "A 'neural-gas' network learns topologies," Artificial Neural Networks: Proc. of the 1991 Int. Conf. on Artificial Neural Networks, pp. 397-402, 1991.

[29] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition. II," IEEE Transactions on Systems, Man, and Cybernetics, Part B, Vol.29, No.6, pp. 786-801, 1999.

[30] B. Fritzke, "A self-organizing network that can follow non-stationary distributions," Artificial Neural Networks – Int. Conf. on Artificial Neural Networks (ICANN'97), pp. 613-618, 1997.

[31] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," Computer, Vol.32, No.8, pp. 68-75, 1999.

[32] S. Fortunato, "Community detection in graphs," Physics Reports, Vol.486, No.3-5, pp. 75-174, 2010.

[33] R. Xu and D. Wunsch, "Survey of clustering algorithms," IEEE Trans. on Neural Networks, Vol.16, No.3, pp. 645-678, 2005.

[34] C. J. Veenman, M. J. T. Reinders, and E. Backer, "A maximum variance cluster algorithm," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.24, No.9, pp. 1273-1280, 2002.

[35] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," Pattern Recognition, Vol.41, No.1, pp. 191-203, 2008.

[36] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," ACM Trans. on Knowledge Discovery from Data, Vol.1, No.1, Article 4, 2007.

[37] E. Lopez-Rubio and E. J. Palomo, "Growing Hierarchical Probabilistic Self-Organizing Graphs," IEEE Trans. on Neural Networks, Vol.22, No.7, pp. 997-1008, 2011.

[38] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," Communications in Statistics, Vol.3, No.1, pp. 1-27, 1974.

[39] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.PAMI-1, No.2, pp. 224-227, 1979.

[40] A. Likas, N. Vlassis, and J. J. Verbeek, "The global *k*-means clustering algorithm," Pattern Recognition, Vol.36, No.2, pp. 451-461, 2003.

Name:
Yuichiro Toda

Affiliation:
Okayama University

Address:
3-1-1 Tsushima-Naka, Kita-Ku, Okayama, Okayama 700-8530, Japan

Brief Biographical History:
2017- Specially Appointed Assistant Professor, Tokyo Metropolitan University
2018- Assistant Professor, Okayama University

Main Works:
• Y. Toda and N. Kubota, "Self-localization Based on Multi-resolution Map for Remote Control of Multiple Mobile Robots," IEEE Trans. on Industrial Informatics, Vol.9, No.3, pp. 1772-1781, 2013.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE)
• Japan Society for Fuzzy Theory and Intelligent Informatics (SOFT)
• The Japan Society of Naval Architects and Ocean Engineers (JSNAOE)

Name:
Takayuki Matsuno

Affiliation:
Okayama University

Address:
3-1-1 Tsushima-Naka, Kita-Ku, Okayama, Okayama 700-8530, Japan

Brief Biographical History:
2006- Assistant, Toyama Prefectural University
2008- Lecturer, Toyama Prefectural University
2011- Lecturer, Okayama University
2017- Associate Professor, Okayama University

Main Works:
• T. Matsuno, T. Kamegawa, T. Hiraki, Y. Toda, and M. Minami, "Needle Angle Offset Compensation Based on Volume CT Image for Needle Puncture Robot," 24th Int. Symp. on Artificial Life and Robotics, pp. 922-927, 2019.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE)
• The Japan Society of Mechanical Engineers (JSME)
• The Robotics Society of Japan (RSJ)

Name:
Mamoru Minami

Affiliation:
Okayama University

Address:
3-1-1 Tsushima-Naka, Kita-Ku, Okayama, Okayama 700-8530, Japan

Brief Biographical History:
1994- Associate Professor, Fukui University
2002- Professor, Fukui University
2010- Professor, Okayama University

Main Works:
• K. N. Lwin, N. Mukada, M. Myint, D. Yamada, A. Yanou, T. Matsuno, K. Saitou, W. Godou, T. Sakamoto, and M. Minami, "Visual Docking against Bubble Noise with Three-dimensional Perception Using Dual-eye Cameras," IEEE J. of Oceanic Engineering, Vol.45, No.1, pp. 247-270, 2020.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE)
• The Japan Society of Mechanical Engineers (JSME)
• The Robotics Society of Japan (RSJ)