Structural Data Recognition with Graph Model Boosting

doi: 10.1109/ACCESS.2018.2876860

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON SUSTAINABLE INFRASTRUCTURES, PROTOCOLS, AND RESEARCH CHALLENGES FOR FOG COMPUTING

Received August 22, 2018, accepted September 26, 2018, date of publication October 22, 2018,
date of current version November 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2876860

Structural Data Recognition With Graph Model Boosting

TOMO MIYAZAKI, (Member, IEEE), AND SHINICHIRO OMACHI, (Senior Member, IEEE)
Graduate School of Engineering, Tohoku University, Sendai 9808579, Japan

Corresponding author: Tomo Miyazaki (tomo@iic.ecei.tohoku.ac.jp)

This work was supported by JSPS KAKENHI under Grants 15H06009 and 16K00259.

ABSTRACT This paper presents a novel method for structural data recognition using a large number of graph models. In general, prevalent methods for structural data recognition have two shortcomings: 1) only a single model is used to capture structural variation and 2) naive classifiers are used, such as the nearest neighbor method. In this paper, we propose strengthening the recognition performance of these models as well as their ability to capture structural variation. The main contribution of this paper is a novel approach to structural data recognition: graph model boosting. We construct a large number of graph models and train a strong classifier using the models in a boosting framework. Comprehensive structural variation is captured with a large number of graph models. Consequently, we can perform structural data recognition with powerful recognition capability in the face of comprehensive structural variation. The experiments using IAM graph database repository show that the proposed method achieves impressive results and outperforms existing methods.

INDEX TERMS Pattern recognition, machine intelligence, structural data recognition.

I. INTRODUCTION

Structural data represented by graphs are a general and powerful representation of objects and concepts. A molecule of water can be represented as a graph with three vertices and two edges, where the vertices represent hydrogen and oxygen, and the relation is intuitively described by the edges. Structural data recognition is used in a wide range of applications; for example, in handwritten characters, symbols from architectural and electronic drawings, images, bioinformatics, and chemical compounds need to be recognized.

Graph recognition is not straightforward. Even measuring the distance between graphs requires various techniques. The problem of graph recognition has recently been actively studied [1]–[6]. Past research has led to two notable progress in two aspects. First, graph models have been developed to capture structural variations. Second, the embedding of graphs into Euclidean space has been used to apply sophisticated classifiers in the vector domain. However, both these aspects have drawbacks. The drawback of the former is that only naive classifiers are applicable, such as the nearest neighbor (NN) or the k-nearest neighbor (k-NN) methods. The drawback of the second aspect above is the loss of structural variation in the classifiers through embedding process.[1]

Our aim in this paper is to overcome the drawbacks of the previous methods. The challenge is how to integrate structural variation into a sophisticated classifier. Inspired by boosting algorithms, we construct a strong classifier by aggregating naive classifiers which use graph models. Hence, the strong classifier can be equipped with comprehensive structural variations because it includes a large number of graph models. Specifically, we construct graph models with weighted training graphs and then train a naive classifier using the models. We update the weight so that we can focus on various graphs. Finally, we construct a strong classifier by aggregating the naive classifiers. We call this novel approach *graph model boosting* (GMB).

The main contribution of this paper is a novel approach to simultaneously strengthen capability of classification and capturing structural variation. In order to capture structural variation comprehensively, we construct a large number of models in a boosting framework so that the models can contain different structural variations and compensate one another. The capability of classification is strengthened by aggregating naive classifiers constructed with graph models. Consequently, we can equip the classifier with comprehensive structural variation and a powerful recognition capability.

In experiments, we demonstrated structural data recognition using GMB on eight graph datasets that were publicly available. We confirmed that accuracy of GMB notably

[1]The embedding process typically involves information loss, local and structural. In contrast, we fully exploit structural information of graphs.

increased as the boosting process continued. Consequently, the accuracy was comparable with the state-of-the-art methods. The experimental results thus show the effectiveness of the GMB.

A preliminary version of the work reported here was first presented in a conference paper [7]. We consolidate and expand our previous description and results. Firstly, we provide additional technical details concerning the graph model and GMB. Our contributions are highlighted clearly in the Abstract and Introduction. Secondly, we carried out a wider survey about related work to clarify the significance of the proposed method. Lastly, additional experimental results are presented: time complexity to evaluate practicality, the impact of the parameters for robustness assessment to various datasets, and using other graph model to show the capability of GMB. The number of datasets used in the experiments is expanded from five to eight by adding datasets in bioinformatics.

II. RELATED WORK

Existing methods for graph recognition can be broadly categorized into three approaches: the one-vs-one approach, the model-based approach, and the embedding approach.

A. ONE-VS-ONE APPROACH

Methods in the one-vs-one approach attempt to classify graphs according to a criterion that can be measured for two graphs, such as graph isomorphism, subgraph isomorphism, and graph edit distance. Two graphs are graph isomorphic if the mapping of their vertices and edges is bijective. Subgraph isomorphism is the case where subgraphs of two graphs satisfy graph isomorphism. Isomorphism is determined by tree search [8] or backtracking [9]. However, it is difficult to determine subgraph isomorphism in case of noisy vertices and edges. Therefore, methods using graph edit distance have been developed. The graph edit distance is defined as the minimum sum of the costs of edit operations that transform the source graph into the target graph by substitution, deletion, and insertion [10], [11]. Since it is expensive to search every combination of edit operations, approximate edit operations are searched. The method developed in [12] applies the A-star algorithm [13], the one in [4] exploits Munkres's algorithm [14], the one in [15] is based on a Hungarian algorithm, and the one in [16] uses simulated annealing. The literature [17], [18] will help readers find more details related to graph matching and edit distance. The advantage of this approach is that the calculation is completed in polynomial time. However, the methods in this approach only adopt naive classifiers, such as the NN method and the k-NN method. The performance of NN and k-NN depend on a metric between graphs, hence we need to define it carefully. However, this approach focuses on two graphs. Consequently, the methods measure a metric without structural variation; only two graphs are considered, whereas other graphs are ignored.

B. MODEL-BASED APPROACH

Methods in the model-based approach attempt to capture structural variation and classify graphs using a model. The median graph [19] is a model that captures global information concerning graphs. The median graph minimizes the total distance between training graphs. A random graph [20] is a model that specializes in capturing attribute variations at each vertex and edge. The random graph contains variables ranging from 0 to 1, which are associated with attribute values. The variables represent the probabilities of the attribute that the vertices take. However, numerous variables are required when attributes are continuous values. Improved models [21]–[23] based on random graph have been developed as well. There are three such models: First-order Gaussian graph, or FOGG [21], function-described graph, or FDG [22], and second-order random graphs, or SORGs [23]. The FOGG is a model designed to avoid increasing number of variables by replacing those of a random graph with parameters of a Gaussian distribution. FDG and SORG incorporate joint probabilities among the vertices and the edges to describe structural information. The difference between FDG and SORG is the numbers of vertices and edges at the calculation of joint probability. FDG uses pairs of vertices or edges, whereas multiple vertices and edges are used in SORG. Recently, models exploiting unsupervised learning methods have been developed [24]. Torsello and Hancock presented a framework to integrate tree graphs into one model by minimizing the minimum description length [25]. Furthermore, Torsello [26] expanded tree graphs to graphs and adopted a sampling strategy [27] to improve calculation efficiency. The EM algorithm has been applied to construct a model [28]. The methods in [24], [26], and [28] concentrate on capturing variations in vertex and edge composition. For computational efficiency, a closure tree [29] has been developed. Each vertex of the tree contains information concerning its descendants, so that effective pruning can be carried out. The model-based approach can measure distance based on structural variation. The drawback of this approach is to use NN and k-NN classifiers using only a single model. A metric is important for this approach. However, the methods use only a single model to measure a metric. We stress that minor structural information is lost in a single model. Consequently, a metric is measured by considering only major variation, whereas minor variation is ignored.

C. EMBEDDING APPROACH

Methods in the embedding approach attempt to apply sophisticated classifiers that are widely used in the vector domain. The main obstacle to embedding graphs is the lack of a straightforward and unique transformation from a graph to a vector. For example, a graph with N vertices can be transformed into $N!$ adjacency matrices, since there are $N!$ permutations of vertex labeling. Jain and Wysotzki [30] embedded graphs into Euclidean space using the Schur–Hadamard inner product and

performed k-means clustering by utilizing neural networks. Bunke and Riesen [5] used the graph edit algorithm to embed graphs into a vector space and applied a support vector machine [31]. Some methods [3], [32], [33] involve applying AdaBoost to graphs. We categorize them as part of this approach because they use sophisticated classifiers. Kudo *et al.* [32] developed a decision stump that responds to whether a query graph includes a specific subgraph and constructs a classifier using stumps in AdaBoost. Nowozin *et al.* [33] developed a classifier based on LPBoost. Zhang *et al.* [3] improved classifiers by incorporating an error-correcting coding matrix method [34] into boosting. Both [3] and [33] adopted the decision stump as a weak classifier. The difference between GMB and these methods [3], [32], [33] is the structural variation in the classifiers. Comprehensive structural variation is incorporated into the GMB, whereas local structural variation is used in [3], [32], and [33] because of the subgraphs. The advantage of the embedding approach is that it can exploit powerful classifiers. However, a disadvantage is that it ignores structural variation, since the graphs cannot be recovered from the vectors. In addition, embedding graphs with structural variation is a challenging task.

D. BRIEF SUMMARY

Summarizing the one-vs-one approach and the model-based approach require a powerful classifier. The methods in the embedding approach need to incorporate structural variations. The advantages of the model-based and the embedding approaches can complement each other to mitigate their disadvantages. Therefore, our strategy for overcoming the disadvantages is to integrate the model-based approach with a sophisticated classifier. In this paper, we build several graph models and incorporate them into AdaBoost to construct a graph classifier that can consider comprehensive structural variation.

III. GRAPH MODEL

We propose a graph model that is fast constructible so that GMB can construct models repeatedly in a feasible amount of time. Generally, we need vertex correspondence among graphs for model construction, but it is challenging to obtain such correspondences. Most existing methods consider correspondence between two graphs [35]–[38]. Therefore, We propose searching vertex correspondences among multiple graphs by assigning labels to vertices with median graphs [19] since the calculation of the median graph is fairly fast.

A. DEFINITION OF GRAPH MODEL

We propose a graph model that captures four types of structural variation in graphs: vertex composition, edge composition, vertex attributes, and edge attributes. The graph model P is defined as

$$P = (V, E, B, \Theta). \tag{1}$$

We call this graph model *probabilistic attribute graph generation model, PAGGM*. V and E are the sets of vertices and edges, respectively. B is a set of probabilities of the vertices and edges, $B = \{b_i, b_{ij} \mid v_i \in V,\ e_{ij} \in E, 0 \leq b_i, b_{ij} \leq 1\}$. Θ is a set of parameters of a probability density function in attributes at the vertices and edges, $\Theta = \{\theta_i, \theta_{ij} \mid v_i \in V,\ e_{ij} \in E\}$. For instance, θ_i will consist of a mean and a standard deviation if we use a normal distribution to describe attributes at the node v_i. The compositions and attribute variations are captured by B and Θ, respectively. We use probability density function f_{pdf} to calculate the probability that vertex v_i takes attribute a as $f_{\mathrm{pdf}}(a|\theta_i)$. We give an example of how the model describes variation in vertex composition. Let $\{v_1, v_2, v_3\}$ and $\{b_1, b_2, b_3\}$ be elements of V and B, respectively. The vertex compositions and probabilities are as follows: $\{v_1\}$ at $b_1(1-b_2)(1-b_3)$, $\{v_2\}$ at $(1-b_1)b_2(1-b_3)$, $\{v_3\}$ at $(1-b_1)(1-b_2)b_3$, $\{v_1, v_2\}$ at $b_1b_2(1-b_3)$, $\{v_1, v_3\}$ at $b_1(1-b_2)b_3$, $\{v_2, v_3\}$ at $(1-b_1)b_2b_3$, and $\{v_1, v_2, v_3\}$ at $b_1b_2b_3$. We can include attribute variations by multiplying $f_{\mathrm{pdf}}(a|\theta)$.

We define likelihood function f_L whereby a model generates an attributed graph $G' = (V', E', A')$, A' is a set of attributes of the nodes and edges, $A' = \{a'_i, a'_{ij} \mid v'_i \in V', e'_{ij} \in E'\}$. Let $Y \in \{0,1\}^{|V|\times|V'|}$ be a matching matrix between P and G', where Y is subject to $\sum_{i=1}^{|V|} Y_{ik} \leq 1$ and $\sum_{k=1}^{|V'|} Y_{ik} \leq 1$ for any vertex v_i in V and v'_k in V'. If two vertices v_i and v'_k are matched, $Y_{ik} = 1$, $Y_{ik} = 0$ otherwise. Therefore, Y provides the correspondence between P and G'. We use function π to refer to the corresponding vertex $\pi : v \text{ in } V_{\mathrm{match}} \rightarrow v' \text{ in } V'_{\mathrm{match}}$, where V_{match} and V'_{match} are sets of corresponding vertices. Specific form is described in Eq. (3). We calculate the likelihood of G' as

$$f_L(G', P) = \max_{Y} \prod_{v_i \in V_{\mathrm{match}}} b_i f_{\mathrm{pdf}}(a'_{\pi(v_i)} \mid \theta_i) \prod_{\bar{v_i} \in V_{\mathrm{miss}}} (1 - b_{\bar{i}}) \prod_{e_{ij} \in E_{\mathrm{match}}} b_{ij} f_{\mathrm{pdf}}(a'_{\pi(v_i)\pi(v_j)} \mid \theta_{ij}) \prod_{e_{\bar{i}\bar{j}} \in E_{\mathrm{miss}}} (1 - b_{\bar{i}\bar{j}}), \tag{2}$$

$$V_{\mathrm{match}} = \{v_i \mid v_i \in V, v'_k \in V', Y_{ik} = 1\}, \tag{3}$$

$$V_{\mathrm{miss}} = V \setminus V_{\mathrm{match}}, \tag{4}$$

$$E_{\mathrm{match}} = \{e_{ij} \mid e_{ij} \in E, e'_{kl} \in E', Y_{ik} = Y_{jl} = 1\}, \tag{5}$$

$$E_{\mathrm{miss}} = E \setminus E_{\mathrm{match}}. \tag{6}$$

How to find Y maximizing Eq. (2) is the crucial step of the proposed method. In order to construct the model, we need a set of correspondences $\boldsymbol{Y} = \{Y^1, \cdots, Y^n\}$ between P and training data $\boldsymbol{G} = \{G_1, \cdots G_n\}$. However, searching $\boldsymbol{Y}$ is a difficult problem. Attempts have been made to estimate $\boldsymbol{Y}$ by minimizing objective functions, such as minimum description length [24], [39] and entropy [40]. Unfortunately, they are not applicable to our study because we search $\boldsymbol{Y}$ for over thousands of subsets of $\boldsymbol{G}$.

We propose a method to quickly calculate the correspondence $\boldsymbol{Y}$. The key idea is to convert searching for $\boldsymbol{Y}$ into labeling vertices, where vertices in $\boldsymbol{G}$ corresponding to the same vertex in P share the same label. We exploit median graphs for labeling. The procedure for the assignment of labels is illustrated in Algorithm 1. $\boldsymbol{M}$ is used for calculating correspondences between P and a graph. We calculate median graph M as

$$M = \underset{G \in \boldsymbol{G}}{\operatorname{argmin}} \sum_{G_i \in \boldsymbol{G}} f_d(G, G_i), \tag{7}$$

Algorithm 1 Vertex Label Assignment

1: Assign null label to every vertex of all graphs in $\boldsymbol{G}$.
2: Calculate the median graph M of $\boldsymbol{G}$.
3: Assign a new label to each null-labeled vertex of M.
4: Calculate matching matrices between M and all graphs in $\boldsymbol{G}$.
  4.1: Calculate edit operations (substitution, deletion, insertion) from M to $\boldsymbol{G}$ by [12].
  4.2: Match substituted vertices between M and $\boldsymbol{G}$.
5: Assign vertex labels of M to null-labeled vertices of graphs in $\boldsymbol{G}$ according to the matching matrices.
6: Go to 9 if $\boldsymbol{G}$ is empty, otherwise go to 7.
7: Remove a graph from $\boldsymbol{G}$ if all of its vertices have labels.
8: Put M into a set of median graphs $\boldsymbol{M}$.
9: Stop if $\boldsymbol{G}$ is empty; otherwise, go to Step 2.

where f_d is a function used to calculate the distance between graphs. We adopt an edit distance algorithm [12] as f_d in this paper.

B. MODEL CONSTRUCTION

The construction of P involves the use of training data $\boldsymbol{G}$ and correspondences $\boldsymbol{Y}$. Specifically, we calculate B and Θ from vertices and attributes that correspond using $\boldsymbol{Y}$.

We create as many vertices and edges of PAGGM as the number of labels assigned by Algorithm 1. It is straightforward to organize the labels of edges in $\boldsymbol{G}$ when the vertex labels of $\boldsymbol{G}$ are available. Edges in $\boldsymbol{G}$ have the same label if the connected vertices of the edges have the same label. Let $\mathcal{L}$ be a function that refers to the labels of vertices; two edges e_{ij} and e_{kl} belong to the same labels if $\{\mathcal{L}(v_i), \mathcal{L}(v_j)\}$ is equal to $\{\mathcal{L}(v_k), \mathcal{L}(v_l)\}$.

We calculate b_i and b_{ij} in B using the labels as

$$b_i = \frac{1}{|\boldsymbol{G}|} \sum_{G_k \in \boldsymbol{G}} \sum_{v \in V_k} \mathbb{I}\left(\mathcal{L}(v) = i\right), \tag{8}$$

$$b_{ij} = \frac{1}{|\boldsymbol{G}|} \sum_{G_k \in \boldsymbol{G}} \sum_{e_{vv'} \in E_k} \mathbb{I}\left(\{\mathcal{L}(v), \mathcal{L}(v')\} = \{i, j\}\right), \tag{9}$$

where $\mathbb{I}$ is a function of proposition p and takes 1 if p is true, and otherwise takes 0. We can accumulate the attributes of

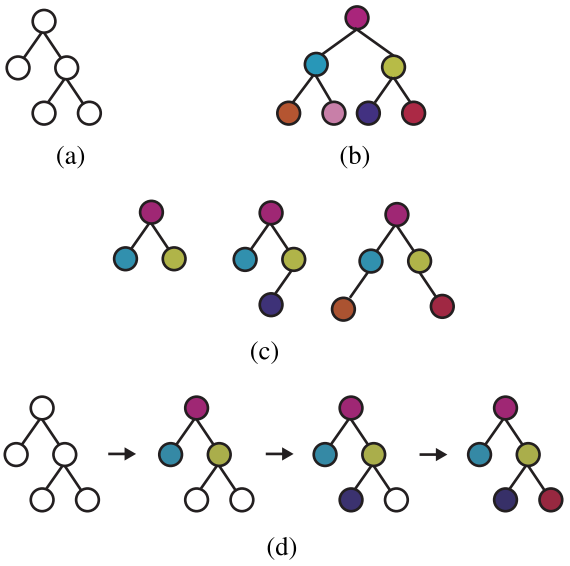

FIGURE 1. Calculation of matching matrix between graph (a) and PAGGM (b). The colors represent labels. The correspondence develops from left to right in (c).

vertex v_i and edge e_{ij} in P as

$$A_i = \bigcup_{G_k \in \boldsymbol{G}} \bigcup_{\substack{v_l \in V_k \\ \mathcal{L}(v_l) = i}} \{a_l\}, \tag{10}$$

$$A_{ij} = \bigcup_{G_k \in \boldsymbol{G}} \bigcup_{\substack{e_{lm} \in E_k \\ \{\mathcal{L}(v_l), \mathcal{L}(v_m)\} = \{i,j\}}} \{a_{lm}\}. \tag{11}$$

Then, we calculate Θ using A_i and A_{ij}. However, we need to determine the type of the probability density function before calculating Θ. There is a miscellany of types of attributes, such as continuous or discrete values, finite or infinite set, etc. A suitable function is determined by a type of attribute. In this paper, we adopt a normal density function and a discrete density function for continuous and discrete values, respectively.

C. SIMILARITY

We describe similarity between graphs and the PAGGM. An intuitive choice for similarity is the logarithm of Eq. (2). This choice is favorable when every vertex and edge in G' matches that in P. However, unmatched vertices and edges in G' occur often. In such cases, the logarithm of Eq. (2) is unsuitable because the unmatched vertices and edges in G' are not accounted for in Eq. (2). Therefore, we impose penalty η on unmatched vertices and edges. We define similarity function f_s of graph G' to a PAGGM P as

$$f_s(G', P) = \log\left(f_L(G', P)\right) - \eta\left(|\bar{V}'| + |\bar{E}'|\right), \tag{12}$$

where $\bar{V}'$ and $\bar{E}'$ represent sets of unmatched vertices and edges of G', respectively. In Section V-B, we provided the experimental results for the effects of η so that the optimal η can be obtained.

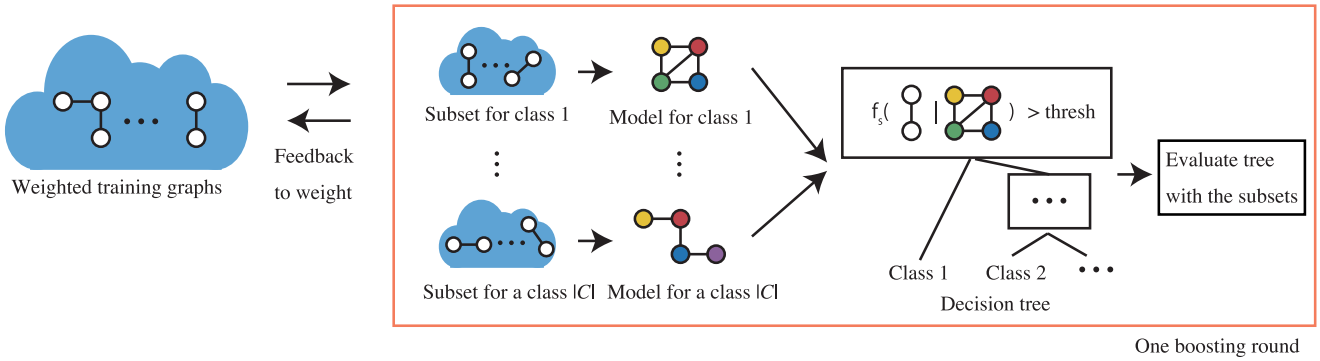

FIGURE 2. Overview of graph model boosting.

Y should be an correspondence maximizing $f_L(G', P)$. However, Y is not easily obtained because the number of vertices of P is large. Therefore, we propose calculating Y by a set of median graphs $\boldsymbol{M} = \{M_i \mid i = 1, \cdots\}$, where i represents the order constructed in Algorithm 1. Specifically, we take M_1 and assign labels to G' using a correspondence between M_1 and G'. We repeat this assignment by taking next M_i until all vertices in G' corresponding to P or all median graphs have been used. Finally, we obtain Y. Note that Y may change if the order of M_i changes. We keep the order so that Y is consistent with the correspondences that P is constructed with the training graphs.

IV. GRAPH MODEL BOOSTING

The aim of GMB is to capture comprehensive variation and construct a powerful classifier with a large number of graph models. Although the PAGGM can contain structural variations, the PAGGM is single and tends to capture major variations in the training graphs. Hence, minor but important variations are lost. Consequently, recognition errors occur on graphs in minor structures. To overcome this problem, we adopt a boosting algorithm to generate a large number of PAGGMs to complement one another. We provide an overview of the GMB in Fig. 2. In each boosting round, a PAGGM is constructed with a subset of weighted training data for each class. Note that the PAGGM can focus on minor graphs with large weights. Then, we form a decision tree that classifies a graph by recursively branching left or right down the tree according to the similarity between the graph and a PAGGM. Finally, we form a strong classifier by integrating the decision trees.

A. MODEL EXTENSION FOR WEIGHTED TRAINING DATA

We extend the PAGGM for weighted training data. We begin by extending the median graph to a *weighted median graph*. Let W be a set of weights for graphs in $\boldsymbol{G}$. We define the weighted median graph $\hat{M}$ as

$$\hat{M} = \underset{G \in \boldsymbol{G}}{\operatorname{argmin}} \sum_{G_i \in \boldsymbol{G}} w_i d(G, G_i). \tag{13}$$

We replace the median graph in Algorithm 1 with the weighted median graph. Therefore, $\boldsymbol{M}$ of P is composed of $\hat{M}$. Subsequently, we extend the calculation of B and Θ to incorporate W as

$$\hat{b}_i = \frac{1}{|\boldsymbol{G}|} \sum_{G_k \in \boldsymbol{G}} \sum_{v \in V_k} w_k \mathbb{I}\left(\mathcal{L}(v) = i\right), \tag{14}$$

$$\hat{b}_{ij} = \frac{1}{|\boldsymbol{G}|} \sum_{G_k \in \boldsymbol{G}} \sum_{e_{vv'} \in E_k} w_k \mathbb{I}\left(\{\mathcal{L}(v), \mathcal{L}(v')\} = \{i, j\}\right), \tag{15}$$

$$\hat{A}_i = \bigcup_{G_k \in \boldsymbol{G}} \bigcup_{\substack{v_l \in V_k \\ \mathcal{L}(v_l) = i}} \{a_l\} \diamond \mathcal{O}(w_l), \tag{16}$$

$$\hat{A}_{ij} = \bigcup_{G_k \in \boldsymbol{G}} \bigcup_{\substack{e_{lm} \in E_k \\ \{\mathcal{L}(v_l), \mathcal{L}(v_m)\} = \{i,j\}}} \{a_{lm}\} \diamond \mathcal{O}(w_{lm}), \tag{17}$$

where $\mathcal{O}$ is a function that refers to the position of w_k in ascending order of W. For example, given $W = \{3, 6, 1, 9\}$, the ascending order of W is $\{1, 3, 6, 9\}$. Therefore, $\mathcal{O}(6) = 3$. Let $\diamond$ represent a duplication operator, such as $\{x\} \diamond N = \{\underbrace{x, x, \cdots, x}_{N \text{ times}}\}$. We calculate $\hat{\Theta}$ with $\hat{A}$. We replace B and Θ with $\hat{B}$ and $\hat{\Theta}$, respectively, when W is given.

B. BOOSTING FRAMEWORK

We propose GMB as shown in Algorithm 2, where C represents a set of class labels of $\boldsymbol{G}$. GMB is based on the AdaBoost algorithms [41]–[43], which are a suitable choice for model construction because they provide different subsets of training data. We construct PAGGMs using the subsets. In addition to the different subsets, the weight can diversify the PAGGMs. In the AdaBoost framework, the weights of error-prone graphs become larger than those of recognized graphs. Hence, we can focus on such error-prone graphs by constructing PAGGMs using weights.

We use decision trees and NN as weak classifiers $h_t : G \to c \in C$. The trees are trained by searching branching rules composed of the PAGGM of a class and a threshold of similarity. We use the CART algorithm [44] to set the threshold. Specifically, we minimize Gini impurity, $1 - \sum_{c \in C} p(c)$, where $p(c) = \frac{n_c}{|\boldsymbol{G}|}$. n_c is the number of samples in class c. The trees have different PAGGMs and branching rules because of randomness in the GMB. Subsequently, we use all trained

TABLE 1. Summary of characteristics of datasets used in the experiments. The top row indicates the number of training data items, validation data, test data, classes, attribute types of vertex and edge, average numbers of vertices and edges, maximum number of vertices and edges, and the number of graphs in a class.

dataset	#train	#valid	#test	#class	vertex attribute	edge attribute	ave(v)	ave(e)	max(v)	max(e)	#a class
LOW	750	750	750	15	(x,y)	none	4.7	3.1	8	6	50
MED	750	750	750	15	(x,y)	none	4.7	3.2	9	7	50
HIGH	750	750	750	15	(x,y)	none	4.7	4.5	9	9	50
GREC	216	218	403	22	(x,y)	Line type	11.6	11.9	24	26	15
COIL	2 400	500	1 000	100	RGB histogram	Line boundary	3.0	3.0	11	13	24
AIDS	250	250	1 500	2	Chemical symbol	Valence	15.7	16.2	95	103	50
Mutagenicity	1 500	500	2 337	2	Chemical symbol	Valence	30.3	30.8	417	112	670
Protein	200	200	200	6	Amino acid sequence	Type	32.6	62.1	126	149	33

Algorithm 2 Graph Model Boosting, GMB

Initialize weights as $w_i = \frac{1}{|\boldsymbol{G}|}$, $i = 1, \ldots, |W|$.
FOR $t = 1$ to T
1: Extract a subset for each class from $\boldsymbol{G}$.
2: Construct a model for each class using the subset.
3: Train a weak classifier h_t using the models.
4: Calculate $err_t = \frac{1}{\sum_{w \in W} w} \sum_{G_i \in \boldsymbol{G}} w_i \mathbb{I}(h_t(G) \neq c_i)$.
5: Calculate $\alpha_t = \log \frac{1-err_t}{err_t} + \log(|C|)$.
6: Renew $w_i \in W$ as $w_i \leftarrow w_i \exp(\alpha_t \cdot \mathbb{I}(h_t(G_i) \neq c_i))$.
7: Normalize W subjected to $\sum_{w_i \in W} w_i = 1$.
End For

trees to construct the strong classifier H as

$$H(G) = \underset{c \in C}{\operatorname{argmax}} \sum_{t=1}^{T} \alpha_t \mathbb{I}\left(h_t(G) = c\right). \tag{18}$$

In recognition, H evaluates test graphs using the trees and recognizes them by majority voting on the trees. Hence, the test graphs are comprehensively analyzed by measuring similarities to the PAGGMs and using more sophisticated classifiers than NN and k-NN. An alternative weak classifier is the NN. We construct PAGGMs at each boosting round. Hence, test graphs are analyzed with various PAGGMs as well as the decision trees.

V. EXPERIMENTS

We carried out the experiments below to verify the effectiveness of the proposed method. We measured recognition accuracy for PAGGM and GMB on eight datasets.

A. DATASETS

For the experiments, we used eight datasets from the IAM graph database repository [45], which is widely used for graph-based pattern recognition and machine learning. We used the following datasets: Letter-LOW (LOW), Letter-MED (MED), Letter-HIGH (HIGH), GREC, COIL-RAG (COIL), AIDS, Mutagenicity, and Protein.

The LOW, MED, and HIGH contained graphs representing distorted letters, which were the 15 capital letters of the Roman alphabet composed of straight lines only, such as A, E, F, H, I, K, L, M, N, T, V, W, X, Y, and Z. The terms "LOW," "MED," and "HIGH" represent the levels of distortion.

The GREC consisted of graphs representing symbols from architectural and electronic drawings. The symbols were converted into graphs by assigning vertices to corners, intersections, and circles on the symbols. The lines between symbols were edges. There were 22 classes in the GREC. As suggested in [46], we used graphs consisting of only straight lines.

The COIL was composed of 100 classes of graphs extracted from images of different objects of the COIL-100 database [47]. The images were segmented into regions according to color by the mean shift algorithm [48]. The vertices were assigned to the segmented regions, and adjacent regions were connected by edges. The attributes were color histograms.

The AIDS was composed of graphs representing molecular compounds of AIDS Antiviral Screen Data.[2] The compounds were active against human immunodeficiency virus or not. We assigned the vertices to the atoms, and covalent bounds were the edges.

The Mutagenicity consisted of graphs representing molecular compounds used in [49]. The compounds were divided into two classes: compounds that have potential to become a marketable drugs, or not. The vertices and edges were assigned as same way as AIDS.

The Protein contained graphs representing proteins developed in [50]. The proteins were classified into six classes according to enzyme classes. The vertices were assigned to the secondary structure elements of a protein. The edges were assigned to three nearest elements in space.

We summarize the characteristics of the datasets in Table 1. We calculated Θ in the following probability density functions: normal distribution for continuous values such as (x,y), discrete distribution function for discontinuous values such as line type, chemical symbol, and amino acid sequence.

B. EXPERIMENTS WITH PAGGM

We carried out experiments with PAGGM. In order to evaluate PAGGM, we constructed one PAGGM for each class by using all training data. We show the constructed PAGGMs for the three classes "A", "E", and "F" in Fig. 3.

[2]https://wiki.nci.nih.gov/display/NCIDTPdata/AIDS+Antiviral+Screen+Data

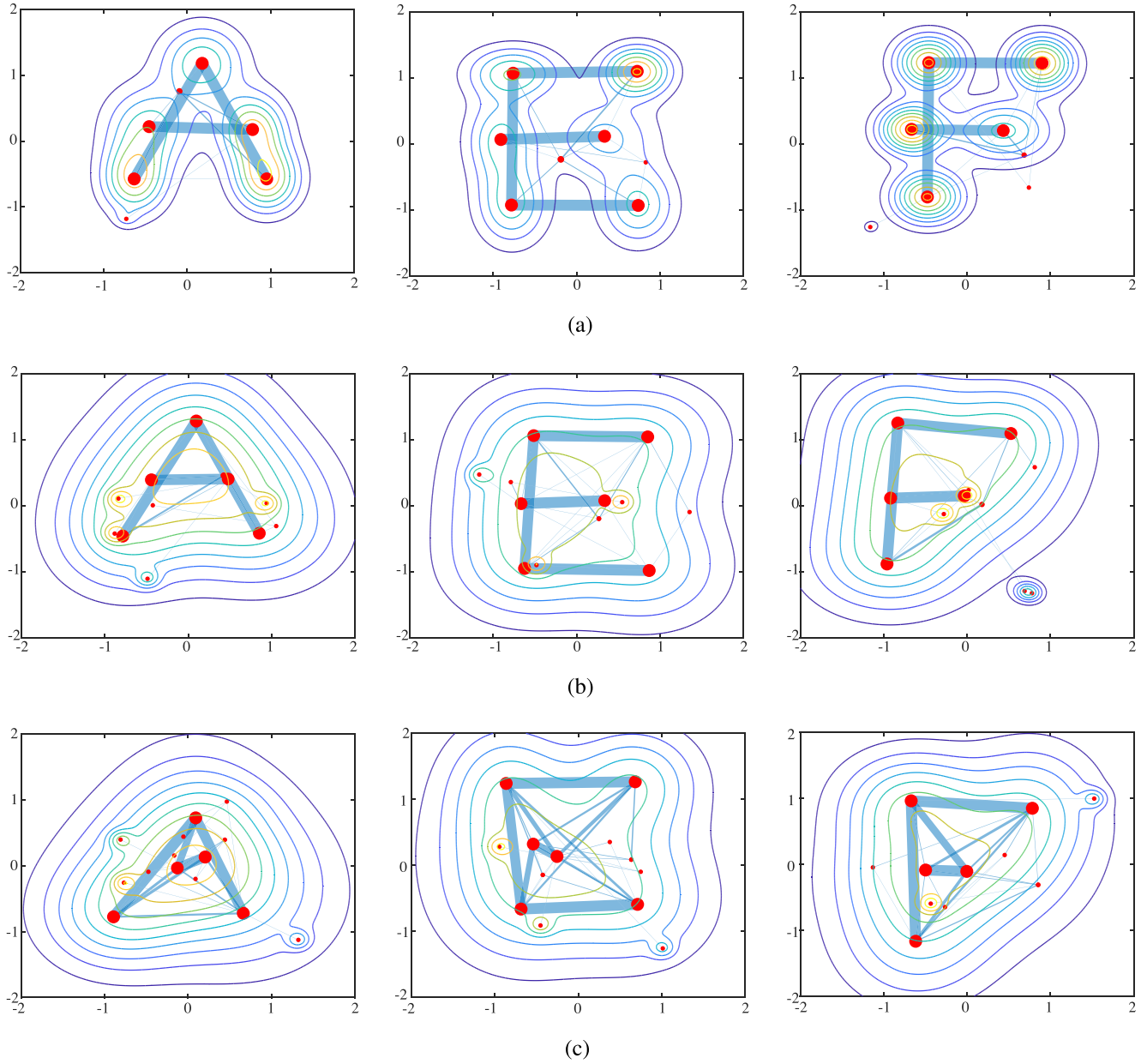

FIGURE 3. Visualization of PAGGM. Red dots and blue lines represent vertices and edges, respectively. The sizes of the dots and thicknesses of the lines represent probabilities B of vertices and edges, respectively; bigger and thicker are higher. The axes represent attribute values of the vertices: x- and y-coordinates. The contours represent probabilities of attributes. We calculated probabilities with $p(x, y) = \sum_{v_i \in V} b_i f_{\mathrm{pdf}}(x, y \mid \theta_i)$, where probability density functions for the normal distribution are used for f_{pdf}. (a) PAGGMs in LOW. (b) PAGGMs in MED. (c) PAGGMs in HIGH.

The vertices and edges compositions in PAGGMs became more complex from LOW to MED, and to HIGH. The captured compositions are consistent with the distortion levels of the dataset. Most importantly, the PAGGMs successfully contain core structures of the classes with high probabilities.

We carried out experiments to show the effects of η in Eq. (12). The effects are shown in Fig. 4. We randomly extracted subsets of the training data and trained PAGGM with them. Specifically, the numbers of training data for a class are 25 in LOW, MED, HIGH and AIDS, 6 in GREC, 12 in COIL, 360 in Mutagenicity, and 15 in Protein. We classified the validation data to the nearest class with similarity obtained by Eq. (12). We tried 10 times and averaged recognition rates because the subsets were randomly extracted. The effects show that the lowest average recognition rates were obtained when $\eta = 0$, i.e., no penalty. The rates were increased when η increased. However, higher η, such as over 20, will decrease the rates. Therefore, we adopted following η: 6 in LOW, 4 in MED, HIGH and COIL, 10 in GREC, AIDS, and Mutagenicity, 8 in Protein.

We then classified the test data by applying the NN method with similarity obtained by Eq. (12). For comparison,

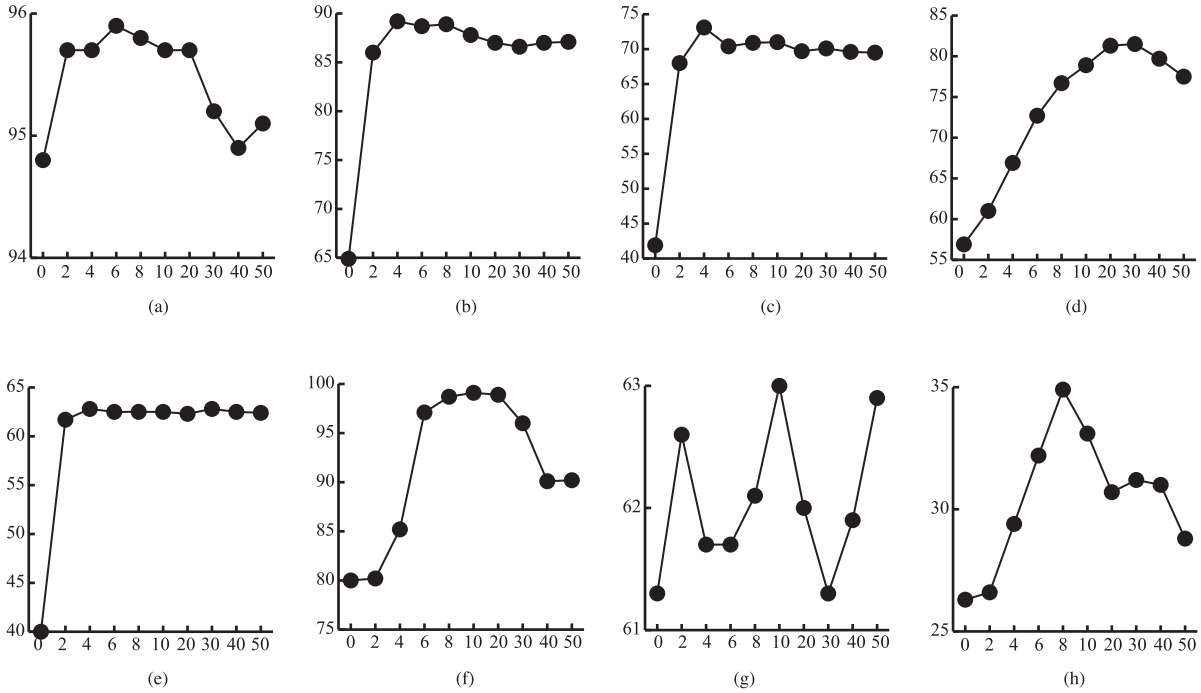

FIGURE 4. The effects of η in Eq. (12) on the validation data. The x- and y-axes represent η and average recognition rates (%) over 10 trials, respectively. (a) LOW. (b) MED. (c) HIGH. (d) GREC. (e) COIL. (f) AIDS. (g) Mutagenicity. (h) Protein.

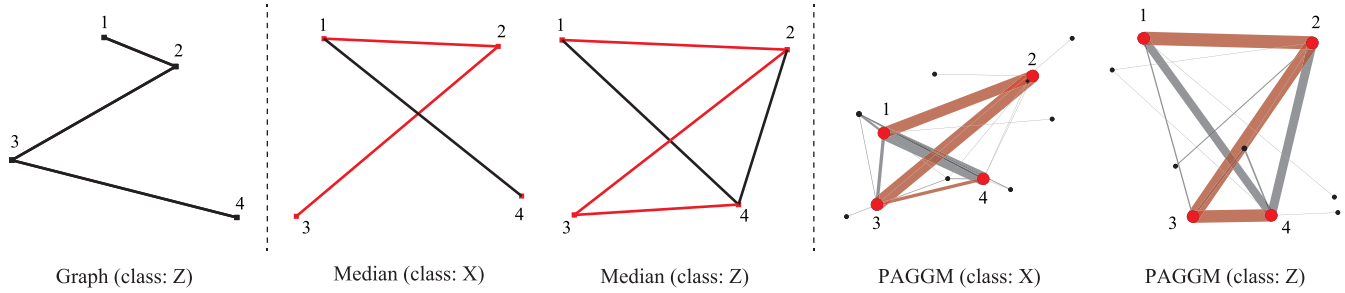

FIGURE 5. The graph of "Z" in HIGH, which was misclassified to "X" using the medians, whereas the PAGGMs successfully classified it to "Z." The red vertices and edges in the medians and PAGGMs matched to the graph. The numbers represent the correspondences. The vertex and edge composition with probabilities are only shown in the PAGGMs. See text for details.

TABLE 2. Recognition rates (%) of median graph and PAGGM using the NN method on the test data.

dataset	Median	PAGGM
LOW	84.3	96.1
MED	77.3	90.9
HIGH	60.0	76.4
GREC	74.4	84.9
COIL	56.9	65.6
AIDS	98.4	98.9
Mutagenicity	61.0	62.2
Protein	22.5	39.5

we calculated the median graphs [19] and classified test data by the edit distance algorithm [12]. The recognition rates are shown in Table 2. The experimental results showed the importance of structural variation. The recognition rates of the PAGGM outperformed the median graph for every dataset. Although both the PAGGM and the median graph were categorized into the model-based approach, the main difference is whether it contains structural variation.

We show the case that the classification was failed in the median graphs whereas PAGGMs succeeded; see Fig. 5. The graph in HIGH was misclassified to "X" because the median "X" has only one edge that did not match but two edges in the median "Z." On the other hand, the graph was successfully classified by the PAGGMs. Although the whole graph was contained in both the PAGGMs, the vertices and edges with high probability were matched in PAGGM "Z" but mismatched in "X;" see the edges (1,4) in "X."

C. EXPERIMENTS WITH GRAPH MODEL BOOSTING

In this experiment, we demonstrated GMB. We set the cardinality of subset n_c to 5 in LOW, MED, and HIGH, 9 in GREC, 6 in COIL, 5 in AIDS, 60 in Mutagenicity, and 27 in

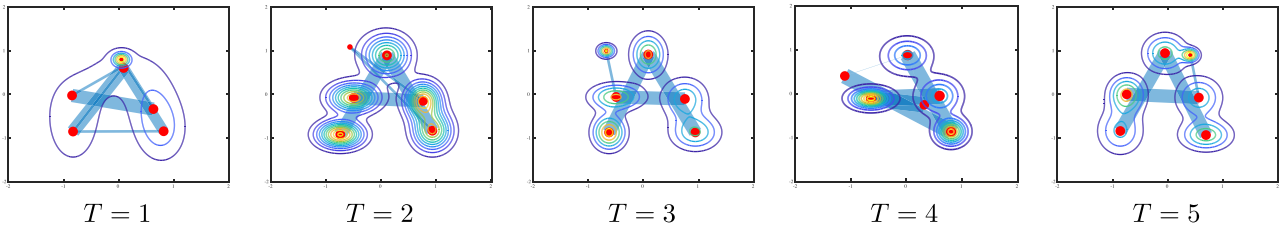

FIGURE 6. PAGGMs constructed by GMB for "A" in LOW.

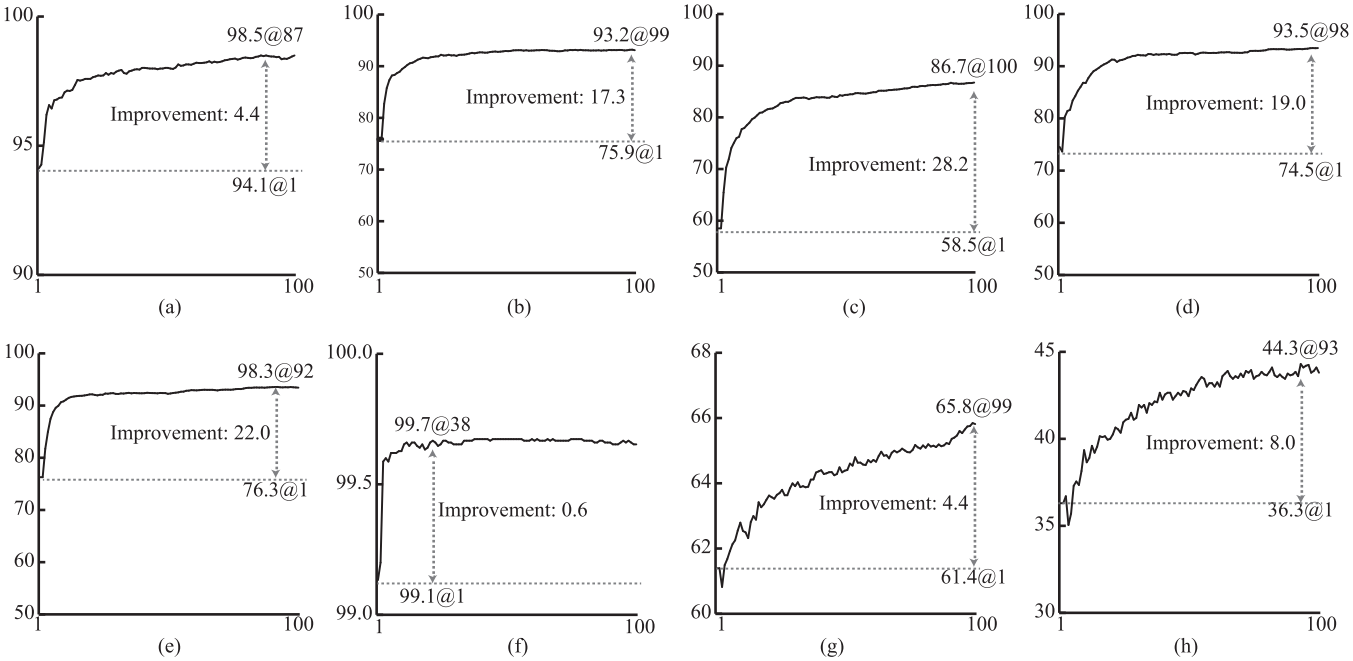

FIGURE 7. Evolutions of average recognition rates of GMB using PAGGM in 100 boosting rounds. The x- and y-axes represent boosting round and average recognition rates, respectively. (a) LOW. (b) MED. (c) HIGH. (d) GREC. (e) COIL. (f) AIDS. (g) Mutagenicity. (h) Protein.

Protein, where the number of boosting rounds T was 100. We adopted decision trees as weak classifiers in LOW, MED, HIGH, GREC, COIL, and AIDS. NN weak classifiers are used in Mutagenicity and Protein. In each round, the PAGGM was constructed with a different subset and weights, hence, we obtained various PAGGMs as shown in Fig. 6. They have unique vertex and edge compositions that are different from the core structure of "A" because we focused to the subset.

We proved the effectiveness of GMB by showing the evolution of its recognition rate in Fig. 7. The recognition rate fluctuated during the trials due to random sampling. We repeated the experiments 10 times and calculated the average recognition rate. The evolutions showed that the average recognition rates increased steadily on every dataset. The improvements were 4.4 in LOW, 17.3 in MED, 28.2 in HIGH, 19.0 in GREC, and 22.0 in COIL, 0.6 in AIDS, 4.4 in Mutagenicity, and 8.0 in Protein. The average recognition rates with $T = 1$ were lower than the results with the single PAGGM, shown in Table 2. At the beginning, GMB only captured the structural variation in subsets of the training data, resulting in poor results. However, the accuracy of GMB increased as the process continued and, finally, exceeded the results of the single PAGGM. This phenomenon signifies that comprehensive structural variations were successfully incorporated into the classifier of the GMB.

We investigated the inference results by counting the number of weak classifiers voting to correct classes and others. The purpose of this investigation is to reveal how many PAGGMs of weak classifiers contributed to the inference results. As shown in Eq. (18), the GMB votes confidence α_t to inference classes using weak classifiers h_t. We counted the number of votes to the classes, and averaged the numbers for each class. The maximum number is 100 since GMB proceeded 100 rounds. Table 3 shows the average numbers. Generally, the correct classes obtained larger votes than other classes in most of datasets. In Mutagenicity, the numbers of votes are comparable. These results show that most PAGGMs in weak classifiers contributed to the inference results.

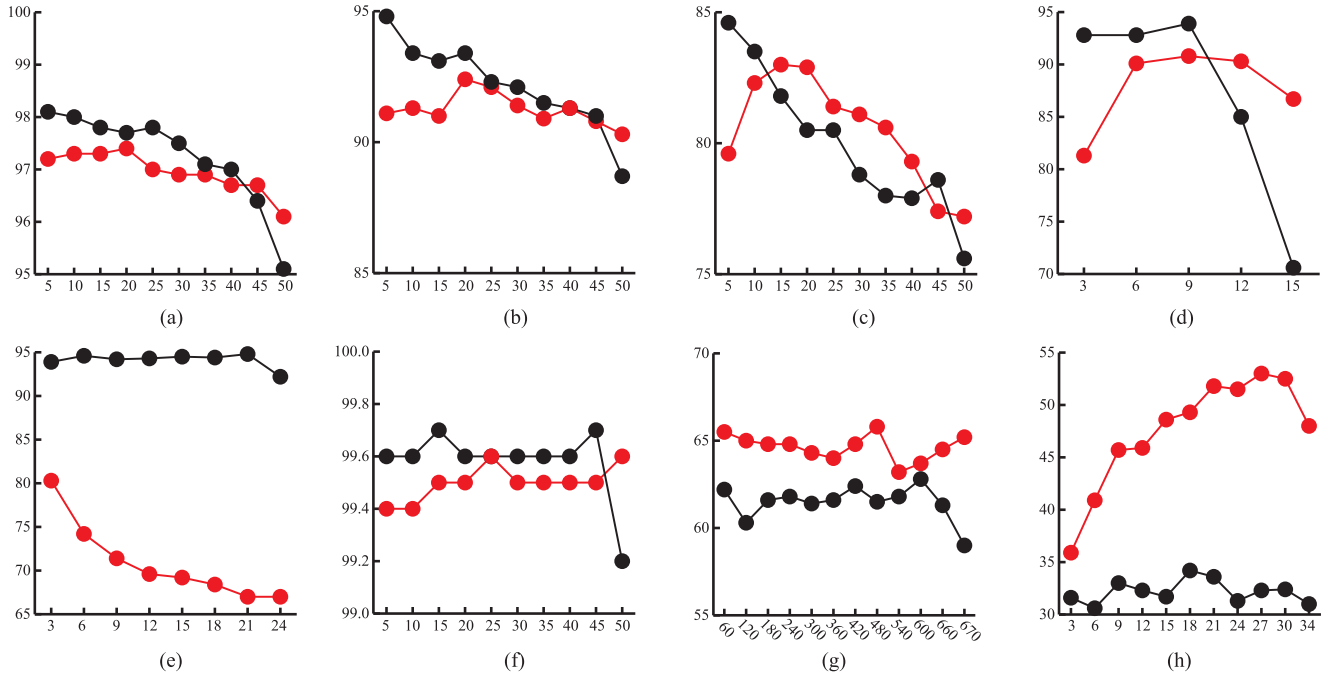

FIGURE 8. The effects of the cardinality of a subset n_c. The y- and x-axes represent average recognition rate (%) and n_c, respectively. The black and red lines represent different weak classifiers, decision tree and NN, respectively. (a) LOW. (b) MED. (c) HIGH. (d) GREC. (e) COIL. (f) AIDS. (g) Mutagenicity. (h) Protein.

TABLE 3. Average numbers of weak classifiers voting to correct and other classes. The maximum number is 100 since 100 rounds are proceeded.

	Correct	Other
LOW	78.4	1.5
MED	52.6	3.4
HIGH	36.8	4.5
GREC	63.9	2.1
COIL	47.1	0.5
AIDS	89.1	10.8
Mutagenicity	48.2	51.8
Protein	27.5	14.5

TABLE 4. Comparison of recognition rate.

Method	Approach	LOW	HIGH	GREC
Riesen [4]	One.	91.1	77.6	-
Jiang [19]	Model	96.4	74.5	80.5
He [29]	Model	93.6	49.1	57.0
Wong [20]	Model	93.9	80.1	85.8
Serratosa [22]	Model	93.9	80.1	85.8
SanFeLiu [23]	Model	94.0	80.9	91.2
Silva [51]	Emb.	96.5	-	97.2
Risen [52]	Emb.	92.7	73.3	92.9
Risen [53]	Emb.	99.3	92.5	96.8
Livi(v1) [54]	Emb.	98.6	**96.2**	96.2
Livi(v2) [54]	Emb.	99.0	**96.2**	**97.9**
Bunke [5]	Emb.	92.9	-	-
Proposed (PAGGM)	Model	96.1	76.4	84.9
Proposed (GMB)	Model	**99.5**	90.9	97.3

We carefully determined n_c since random sampling is an important process. We repeated this experiment 10 times for each n_c and calculated the maximum values of the average recognition rates. Fig. 8 shows the results. The results show the robustness of GMB. In general, the effects were within a few points of one another on the datasets. GMB achieved good results on the datasets using various n_c. Note that the results in LOW and GREC signified that performance can be maximized when the number of samples is small. The small number of samples facilitated the generation of different PAGGMs. Consequently, more comprehensive structural variations were incorporated into the GMB.

We compared GMB with existing graph recognition methods, categorized into the one-vs-one [4], the model-based [19], [20], [22], [23], [29], and the embedding approaches [5], [51]–[54]. The comparisons on LOW, HIGH, and GREC are summarized in Table 4, where the results of GMB are obtained with 1,000 boosting rounds. We referred to the results of [4], [20], [22], [23], [29] to [46] on the IAM database, because they were not evaluated in the original papers. The method of the one-vs-one approach recorded high scores, and the methods of the model-based approach obtained even higher scores. These results signify the importance of structural information. The PAGGM obtained high scores on all the datasets. Furthermore, GMB outperformed the other methods of the model-based approach, and the results of GMB are comparable with the methods of the

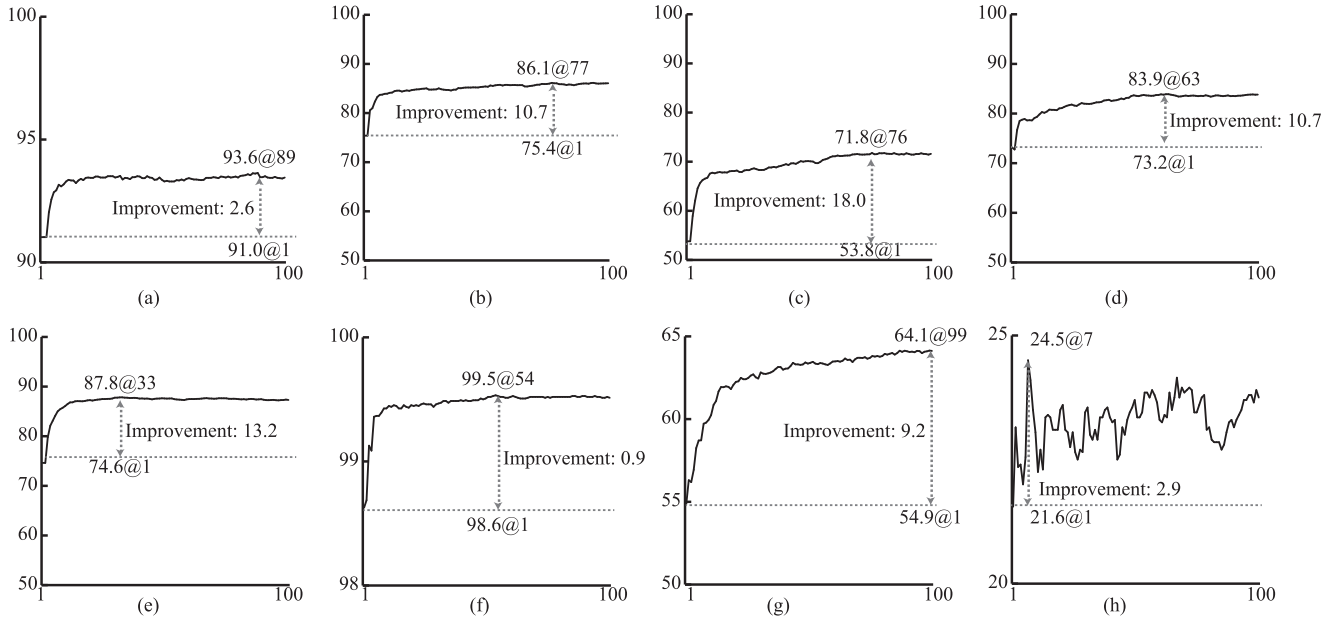

FIGURE 9. Evolutions of average recognition rates of GMB using *median graph* as model. (a) LOW. (b) MED. (c) HIGH. (d) GREC. (e) COIL. (f) AIDS. (g) Mutagenicity. (h) Protein.

embedding approach. It was shown that the proposed method can incorporate a large amount of structural information into a powerful classifier. These comparison results verified the effectiveness of GMB.

Finally, to demonstrate the capability of GMB, we carried out further experiments whether GMB works well with other graph models. To this end, we used the median graph as graph models constructed in GMB. The parameters n_c and T are same as the experiment using PAGGM. We also calculated average recognition rates, and Fig. 9 shows the results. The average recognition rates using median graphs increased on every dataset, likewise the GMB using PAGGM. The improvements were 2.6 in LOW, 10.7 in MED, 18.0 in HIGH, 10.7 in GREC, and 13.2 in COIL, 0.9 in AIDS, 9.2 in Mutagenicity, and 2.9 in Protein. The GMB successfully captured structural variation using the generated median graphs, and the recognition performance was significantly improved. These results signify the capability of GMB for graph models.

D. TIME COMPLEXITY ANALYSIS

PAGGM construction demands steps for an edit distance matrix and vertex label assignment: n^2 and m_v steps, respectively. Where n and m_v are the number of the training graphs and vertices in the training graphs, respectively. GMB repeats PAGGM construction over T rounds, resulting in a considerable amount of time. Therefore, to reduce the steps, we calculated the edit distance matrix only once, prior to GMB. We simply referred to the matrix at PAGGM construction in GMB. Consequently, the complexity of GMB is $O(m_v)$. We stress that the computational time of GMB is feasible. For instance, FOGG [21] requires $O(m_v^2)$ steps.

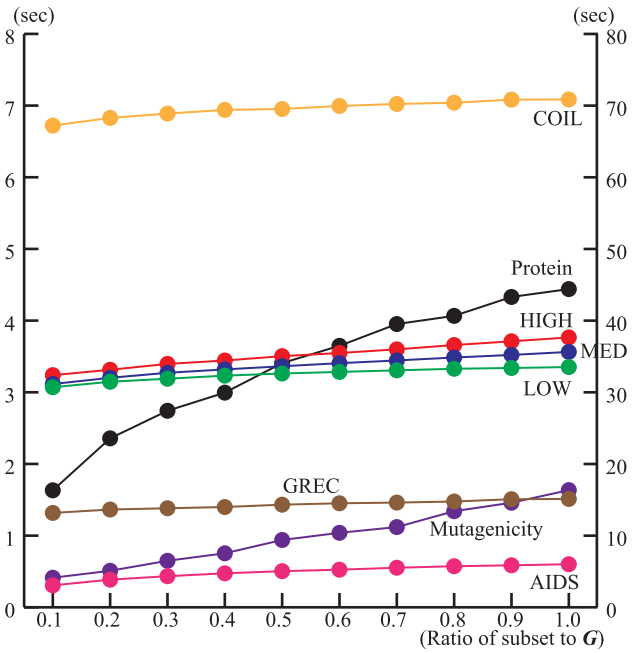

FIGURE 10. Computational time of one round in GMB. The x-axis represents ratio of the subset to the all training graphs. The right y-axis is for COIL and Mutagenicity, and the left one is for the others.

We show the computational time of GMB in Fig. 10. The times in COIL and Mutagenicity are long due to the large numbers of training graphs. Generally, the times were proportional to the size of the training data n. Note that the number of vertices m_v is correlated with n.

The details of processing times in GMB were shown in Table 5. The constructions for PAGGM and weak classifiers are composed of most of the processing times.

TABLE 5. The details of processing times (sec/round) in GMB.

	One round	Train PAGGM	Train weak classifier	Others
LOW	3.2	0.08	3.1	0.0009
MED	3.3	0.09	3.2	0.0009
HIGH	3.4	0.1	3.3	0.0008
GREC	1.4	0.13	1.3	0.0006
COIL	69.7	0.35	69.3	0.007
AIDS	0.52	0.26	0.25	0.0002
Mutagenicity	10.8	6.1	4.6	0.002
Protein	3.5	2.3	1.3	0.008

VI. CONCLUSIONS

In this paper, we proposed a novel algorithm for graph recognition called GMB. The proposed method constructs a graph model, PAGGM, to capture structural variations in composition and attribute of vertex and edge. We developed an efficient algorithm for searching vertex correspondences among multiple graphs. The algorithm enables us to construct PAGGM in a feasible amount of time. GMB constructs a large number of PAGGMs to comprehensively capture the structural variation of graphs. Then, we formed a strong classifier using the constructed PAGGMs in the boosting framework. Consequently, the classifier was equipped with the requisite information concerning structural variation and a powerful recognition ability.

The experimental results showed that PAGGM successfully captured structural variation and GMB significantly enhances recognition performance. Furthermore, GMB outperformed the existing methods of the model-based approach, and the results of GMB are comparable with the methods of the embedding approach. Therefore, we successfully strengthened the recognition capability and the ability to deal with structural variation in graphs.

Structural data are a powerful representation of objects even in images. The proposed method can be applied to computer vision tasks where relation of visual features needs to be considered. Developing object recognition applications using the proposed method is planned for future work.

REFERENCES

[1] D. White and R. C. Wilson, "Parts based generative models for graphs," in *Proc. Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

[2] L. Han, L. Rossi, A. Torsello, R. C. Wilson, and E. R. Hancock, "Information theoretic prototype selection for unattributed graphs," in *Structural, Syntactic, and Statistical Pattern Recognition* (Lecture Notes in Computer Science), vol. 7626. Berlin, Germany: Springer, 2012, pp. 33–41.

[3] B. Zhang *et al.*, "Multi-class graph boosting with subgraph sharing for object recognition," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 1541–1544.

[4] K. Riesen and H. Bunke, "Approximate graph edit distance computation by means of bipartite graph matching," *Image Vis. Comput.*, vol. 27, no. 7, pp. 950–959, Jun. 2009.

[5] H. Bunke and K. Riesen, "Towards the unification of structural and statistical pattern recognition," *Pattern Recognit. Lett.*, vol. 33, no. 7, pp. 811–825, May 2012.

[6] C. F. Moreno-García, F. Serratosa, and X. Jiang, "An edit distance between graph correspondences," in *Proc. Graph-Based Represent. Pattern Recognit.*, 2017, pp. 232–241.

[7] T. Miyazaki and S. Omachi, "Graph model boosting for structural data recognition," in *Proc. 23rd Int. Conf. Pattern Recognit.*, 2016, pp. 1708–1713.

[8] J. R. Ullmann, "An algorithm for subgraph isomorphism," *J. ACM*, vol. 23, no. 1, pp. 31–42, Jan. 1976.

[9] D. E. Ghahraman, A. K. C. Wong, and T. Au, "Graph optimal monomorphism algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. 10, no. 4, pp. 181–188, Apr. 1980.

[10] M. A. Eshera and K.-S. Fu, "A graph distance measure for image analysis," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-14, no. 3, pp. 398–408, May/Jun. 1984.

[11] A. Sanfeliu and K.-S. Fu, "A distance measure between attributed relational graphs for pattern recognition," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, no. 3, pp. 353–362, May/Jun. 1983.

[12] M. Neuhaus, K. Riesen, and H. Bunke, "Fast suboptimal algorithms for the computation of graph edit distance," in *Proc. Int. Conf. Struct., Syntactic, Statist. Pattern Recognit.*, 2006, pp. 163–172.

[13] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[14] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Soc. Ind. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.

[15] S. Bougleux and B. Gaüzère, and L. Brun, "A Hungarian algorithm for error-correcting graph matching," in *Proc. Graph-Based Represent. Pattern Recognit.*, 2017, pp. 118–127.

[16] K. Riesen, A. Fischer, and H. Bunke, "Improved graph edit distance approximation with simulated annealing," in *Proc. Graph-Based Represent. Pattern Recognit.*, 2017, pp. 222–231.

[17] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 18, no. 3, pp. 265–298, 2004.

[18] P. Bille, "A survey on tree edit distance and related problems," *Theor. Comput. Sci.*, vol. 337, nos. 1–3, pp. 217–239, Jun. 2005.

[19] X. Jiang, A. Münger, and H. Bunke, "On median graphs: Properties, algorithms, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 10, pp. 1144–1151, 2001.

[20] A. K. C. Wong and M. You, "Entropy and distance of random graphs with application to structural pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-7, no. 5, pp. 599–609, Sep. 1985.

[21] A. D. Bagdanov and M. Worring, "First order Gaussian graphs for efficient structure classification," *Pattern Recognit.*, vol. 36, no. 6, pp. 1311–1324, Jun. 2003.

[22] F. Serratosa, R. Alquézar, and A. Sanfeliu, "Function-described graphs for modelling objects represented by sets of attributed graphs," *Pattern Recognit.*, vol. 36, no. 3, pp. 781–798, Mar. 2003.

[23] A. Sanfeliu, F. Serratosa, and R. Alquézar, "Second-order random graphs for modeling sets of attributed graphs and their application to object learning and recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 18, no. 3, pp. 375–396, 2004.

[24] A. Torsello and E. R. Hancock, "Learning shape-classes using a mixture of tree-unions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 6, pp. 954–967, Jun. 2006.

[25] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[26] A. Torsello, "An importance sampling approach to learning structural representations of shape," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–7.

[27] J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods*. Hoboken, NJ, USA: Wiley, 1964.

[28] L. Han, R. C. Wilson, and E. R. Hancock, "A supergraph-based generative model," in *Proc. Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 1566–1569.

[29] H. He and A. K. Singh, "Closure-tree: An index structure for graph queries," in *Proc. Int. Conf. Data Eng.*, Apr. 2006, p. 38.

[30] B. J. Jain and F. Wysotzki, "Central clustering of attributed graphs," *Mach. Learn.*, vol. 56, nos. 1–3, pp. 169–207, 2004.

[31] V. N. Vapnik, *Statistical Learning Theory*. Hoboken, NJ, USA: Wiley, 1998.

[32] T. Kudo, E. Maeda, and Y. Matsumoto, "An application of boosting to graph classification," in *Proc. 17th Adv. Neural Inf. Process. Syst.*, 2005, pp. 729–736.

[33] S. Nowozin, K. Tsuda, T. Uno, T. Kudo, and G. Bakir, "Weighted substructure mining for image analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[34] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *J. Artif. Intell. Res.*, vol. 2, no. 1, pp. 263–286, Jan. 1995.
[35] M. Cho, J. Lee, and K. Lee, "Reweighted random walks for graph matching," in *Computer Vision—ECCV* (Lecture Notes in Computer Science), vol. 6315. Berlin, Germany: Springer, 2010, pp. 492–505.
[36] M. Cho, K. Alahari, and J. Ponce, "Learning graphs to match," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 25–32.
[37] T. Cour, P. Srinivasan, and J. Shi, "Balanced graph matching," in *Proc. 19th Adv. Neural Inf. Process. Syst.*, 2007, pp. 313–320.
[38] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 4, pp. 377–388, Apr. 1996.
[39] L. Han, R. C. Wilson, and E. R. Hancock, "Generative graph prototypes from information theory," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2013–2027, Oct. 2015.
[40] H. Bunke, P. Foggia, C. Guidobaldi, and M. Vento, "Graph clustering using the weighted minimum common supergraph," in *Graph Based Representations in Pattern Recognition* (Lecture Notes in Computer Science), vol. 2726. Berlin, Germany: Springer, 2003, pp. 235–246.
[41] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.
[42] J. Zhu, H. Zou, S. Rosset, and T. Hastie, "Multi-class AdaBoost," *Statist. Interface*, vol. 2, no. 3, pp. 349–360, 2009.
[43] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *J. Mach. Learn. Res.*, vol. 1, pp. 113–141, Sep. 2001.
[44] L. Breiman, *Classification and Regression Trees*. London, U.K.: Chapman & Hall, 1984.
[45] K. Riesen and H. Bunke, "IAM graph database repository for graph based pattern recognition and machine learning," in *Proc. Int. Workshop Struct., Syntactic, Statist. Pattern Recognit.*, 2008, pp. 287–297.
[46] A. Solé-Ribalta, X. Cortś, and F. Serratosa, "A comparison between structural and embedding methods for graph classification," in *Structural, Syntactic, and Statistical Pattern Recognition* (Lecture Notes in Computer Science), vol. 7626. Berlin, Germany: Springer, 2012, pp. 234–242.
[47] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library: Coil-100," Dept. Comput. Sci., Columbia Univ., New York, NY, USA, Tech. Rep. CUCS-006-96, 1996.
[48] D. Comaniciu and P. Meer, "Robust analysis of feature spaces: Color image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 1997, pp. 750–755.
[49] J. Kazius, R. McGuire, and R. Bursi, "Derivation and validation of toxicophores for mutagenicity prediction," *J. Med. Chem.*, vol. 48, no. 1, pp. 312–320, 2005.
[50] H. M. Berman *et al.*, "The protein data bank," *Nucl. Acids Res.*, vol. 28, no. 1, pp. 235–242, 2000.
[51] F. B. Silva, R. de O. Werneck, S. Goldenstein, S. Tabbone, and R. S. da Torres, "Graph-based bag-of-words for classification," *Pattern Recognit.*, vol. 74, pp. 266–285, Feb. 2018.
[52] K. Riesen and H. Bunke, "Reducing the dimensionality of dissimilarity space embedding graph kernels," *Eng. Appl. Artif. Intell.*, vol. 22, no. 1, pp. 48–56, 2009.
[53] K. Riesen and H. Bunke, "Graph classification by means of Lipschitz embedding," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1472–1483, Dec. 2009.
[54] L. Livi, A. Rizzi, and A. Sadeghian, "Optimized dissimilarity space embedding for labeled graphs," *Inf. Sci.*, vol. 266, pp. 47–64, May 2014.

TOMO MIYAZAKI (M'09) received the Ph.D. degree from the Graduate School of Engineering, Tohoku University, in 2011. He joined Hitachi, Ltd., in 2011. He was a Researcher with the Graduate School of Engineering, Tohoku University, from 2013 to 2014, where he has been an Assistant Professor since 2015. His research interests include pattern recognition and image processing. He is a member of the Institute of Electronics, Information and Communication Engineers.

SHINICHIRO OMACHI (M'96–SM'11) received the B.E., M.E., and Ph.D. degrees from Tohoku University, Japan, in 1988, 1990, and 1993, respectively. He was a Research Associate with the Education Center for Information Processing, Tohoku University, from 1993 to 1996. Since 1996, he has been with the Graduate School of Engineering, Tohoku University, where he is currently a Professor. From 2000 to 2001, he was a Visiting Associate Professor at Brown University. His research interests include pattern recognition, computer vision, image processing, image coding, and parallel processing. He received the IAPR/ICDAR Best Paper Award in 2007, the Best Paper Method Award of the 33rd Annual Conference of the GfKl in 2010, the ICFHR Best Paper Award in 2010, and the IEICE Best Paper Award in 2012. He is a member of the Institute of Electronics, Information and Communication Engineers, among others.