Paper

Representative Graph Generation for Graph-Based Character Recognition

Tomo Miyazaki, Shinichiro Omachi (Member)
Graduate School of Engineering, Tohoku University

⟨**Summary**⟩ In graph-based pattern recognition, representative graph influences the performances of recognition and clustering. In this paper, we propose a learning method for generating a representative graph of a set of graphs by constructing graph unions with merging corresponding vertices and edges. Those corresponding vertices and edges are obtained using common vertices of a set. The proposed method includes extracting common vertices and correspondences of vertices. To show the validly of the proposed method, we applied the proposed method to pattern recognition problems with character graph database and graphs obtained from decorative character images.

Key words: Graph-based pattern recognition, representative graph, graph clustering, character recognition

1. Introduction

The learning method for representative graph is important in the field of computer vision and pattern recognition. The determination for representative graph greatly influences the performance of clustering graphs. Graph clustering can be applied to various applications since the object representation with graph is more convenient and suitable than the object representation with vector in various applications.

The character recognition is one of those applications. A graph can represent the structure of a character simply and clearly. Fig.1 shows examples of graph representations of characters. Once the structures of characters are represented by graphs, a structure analysis method can be applied to graphs for recognition. The structure analysis method is one of the main methods for character recognition. The structure analysis method extracts structures from characters and calculates similarities by matching structures between test data and models. The advantage of structure matching is the robustness for bending and expansion of character strokes[1),2),6)]. Moreover, structure analysis method is effective and applicable for not only common fonts like *Times Roman* and *Arial*, but also not common fonts such as decorated characters.

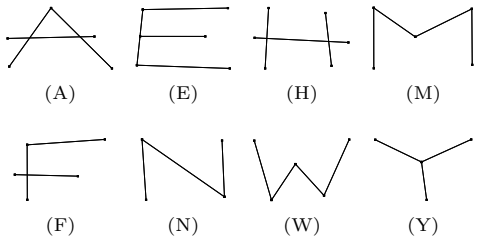


Fig. 1 Examples of character graphs.

As we mentioned above, the representative graph greatly influences the performance of graph clustering and recognition. Therefore, there has been some efforts aimed at learning representative graphs and summarizing graphs in classes. Torsello and Hancock combined available sample graphs with the minimum description length approach to obtain representative graphs[8)]. Bunke et al. extracted representative graphs by creating common supergraphs[7)]. However, both methods are inappropriate to apply character

graphs directly. The former method is limited for only tree graphs and the character graphs are not always trees. The latter method generates representative graphs of which the minimum number of vertices among supergraphs of a set. However, such criterion is inappropriate for character graphs.

In this paper, we propose a learning method for representative character graphs. The proposed method extracts common vertices from each training set, then search correspondences of vertices and edges are using common vertices. We designed representative graph to have all structures of a training set. Such representative graph can be obtained by generating graph union of a set. Moreover, in order to reduce the redundancy of the graph union, we merged vertices and edges of the graph union with obtained correspondences. To show the effectiveness of the proposed method, experiments are carried out using character graph datasets and decorative character images.

2. Proposed Method

We propose a learning method for representative character graphs. The proposed method extracts the common vertices and correspondences of vertices, then form the representative graph with merging vertices and edges.

2.1 Graph Definitions

First of all, we describe the graph definition used in this paper. Let g be a graph defined as

$$g = (\boldsymbol{V}, \boldsymbol{E}, \lambda, \varepsilon, \omega)\,, \tag{1}$$

where $\boldsymbol{V}$ and $\boldsymbol{E}$ are finite sets of vertices and edges, respectively. λ and ε are the functions assigned weights to $\boldsymbol{V}$ and $\boldsymbol{E}$, respectively. λ and ε are initially 1. ω is a function indicating points of vertices in plane of a Cartesian coordinate system. ω is normalized by adjusting the longer length of height or width to 1 and the center of vertices to the origin. We set three vertex types, *end vertex*, *curve vertex* and *junction vertex*, that have one, two and more than three edges, respectively.

2.2 Extraction of Common Vertices

There is a tendency that vertices are gathered at specific locations because of the characteristics of characters. For example, vertices shown in Fig. 2 are expected to gather into five locations. The vertices (a_1, b_1, c_1, d_1, e_1, f_1), vertices (a_4, b_4, c_4, d_4, e_4, f_4), vertices (a_2, b_2, c_2, d_2, e_2, f_2), vertices (a_5, b_5, c_5, d_5, e_5, f_5) and vertices (a_3, b_3, c_3, d_3, e_3, f_3, f_6) are gathered, respectively. Therefore, we define common vertices as centroids of gathered vertices. The common vertices of class k is defined as equation (2).

$$\boldsymbol{C}^k = \left\{c_1^k, c_2^k, \cdots, c_M^k\right\}. \tag{2}$$

We applied ISODATA[10] for ω of training graphs to obtain $\boldsymbol{C}^k$. The ISODATA is an unsupervised classification technique and similar to the k-means algorithm. The k-means algorithm commences to generate centroids randomly and assigns data to the nearest centroid. Then it updates centroids according to data of classes. The k-means algorithm repeats above procedures till centroids convergence. The ISODATA expands the k-means algorithm to integrate classes and divide a class if the distance between centroids is less than the threshold and if the variance of a class is bigger than the threshold, respectively. Note that the k-means algorithm requires the number of classes in advance, while the ISODATA adjusts the output number of classes. The common vertex c_i^k is obtained by equation (3) from training graphs $\boldsymbol{G}_t^k = \{g_{t_1}^k, \cdots\}$ of a class k.

$$c_i^k = \frac{\sum_{j=1}^{|{}_i\boldsymbol{V}_t^k|} \omega({}_i v_{t_j}^k)}{|{}_i\boldsymbol{V}_t^k|}, \tag{3}$$

where ${}_i\boldsymbol{V}_t^k = \{{}_i v_{t_1}^k, \cdots\} \in \boldsymbol{G}_t^k$ is a set of vertices assigned to the nearest centroid i.

Once the common vertices are extracted, two characteristics of common vertices can be obtained. The characteristics are the tendency of vertex type and the connectivity strength of common vertices. First characteristic indicates which vertex type tends to appear at a common vertex. For example, c_1 and c_5 are

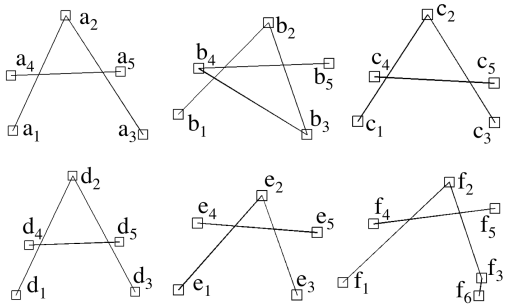


Fig. 2 Graphs of "A".

supposed to be obtained from vertices (a_1, b_1, c_1, d_1, e_1, f_1) and vertices (a_3, b_3, c_3, d_3, e_3, f_3, f_6). The *end vertex* tend to be gathered at c_1. However, there will be some structure distortions, and some vertex type can appear at a common vertex, such as c_5 has 5 *end vertices* and 1 *curve vertices*. We observed the tendency from $\boldsymbol{G}_t^k$ and define a function indicates how a vertex type frequently appear at c_i^k as

$$T_i^k(x) = \frac{1}{|\boldsymbol{G}_t^k|} \cdot \sum_{j=1}^{|\boldsymbol{G}_t^k|} \frac{|{}_i^x\boldsymbol{V}_{t_j}^k|}{|\boldsymbol{V}_{t_j}^k|} \tag{4}$$

where x denotes vertex type. ${}_i^x\boldsymbol{V}_{t_j}^k$ is a set of vertices which are type x and assigned to c_i^k from $\boldsymbol{V}_{t_j}^k \in g_{t_j}^k$. Second, a function which indicates the connectivity strength between c_i^k and c_j^k is defined by equation (5).

$$S^k(i,j) = \frac{1}{|\boldsymbol{G}_t^k|} \cdot \sum_{l=1}^{|\boldsymbol{G}_t^k|} \frac{|{}_{i,j}\boldsymbol{E}_{t_l}^k|}{|\boldsymbol{E}_{t_l}^k|}, \tag{5}$$

where ${}_{i,j}\boldsymbol{E}_{t_l}^k = {}_i\boldsymbol{V}_{t_l}^k \times_j \boldsymbol{V}_{t_l}^k$.

2.3 Correspondence of Vertices

Once the common vertices and function T and S are extracted, we search the correspondences between vertices of graphs to common vertices. The correspondence of vertices of one graph to common vertices should be one-to-one. Therefore, we select one vertex which corresponds to a common vertex from one graph. Basically, vertices of a graph correspond to the nearest common vertex. When one-to-one correspondence is not satisfied, that is, in the case of vertices of a graph correspond to a common vertex, we select one vertex. Let ${}_j\boldsymbol{V}_{t_i}^k \in g_{t_i}^k$ be a set of vertices assigned to c_j^k, a vertex is selected by equation (6).

$$\arg\max_{v \in {}_j\boldsymbol{V}_{t_i}^k} T_j^k(\theta(v)) + \sum_{a \in \boldsymbol{A}_v} S^k(\phi(v), \phi(a)), \tag{6}$$

where θ is a function assigning a vertex to a vertex type. ϕ is a function assigning a vertex to the nearest common vertex. $\boldsymbol{A}_v$ is a set of adjacent vertices of v. Vertices corresponding to a same common vertex are corresponded each other.

Moreover, we take correspondences between ${}_j\bar{\boldsymbol{V}}_{t_i}^k$ and ${}_j\bar{\boldsymbol{V}}_{t_j}^k$ if the Euclidean norm of two vertices on ω are less than threshold. Where ${}_j\bar{\boldsymbol{V}}_{t_i}^k$ is a set of vertices which are not selected by equation (6). We denote the obtained correspondence of vertices and edges $\mathcal{O}$.

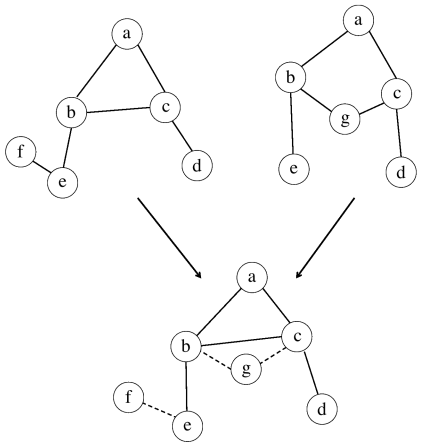


Fig. 3 Graph union of two graphs with merging corresponding vertices and edges. The alphabets denote correspondence of vertices.

2.4 Representative Graph

We generate the representative graph from a set of graphs. The representative graph is designed to contain all the structure information of a set. The graph union is one of the solutions to form a graph which contains all the structure information of a set. Therefore, we form a representative graph by making graph union.

The graph union[5]is defined as follows. Given two graphs g_1 and g_2, the graph union $\hat{g}$ of g_1 and g_2 is defined as $\hat{g} = g_1 \cup g_2$, where $\hat{\boldsymbol{V}}_g$ and $\hat{\boldsymbol{E}}_g$ are $\boldsymbol{V}_1 \cup \boldsymbol{V}_2$ and $\boldsymbol{E}_1 \cup \boldsymbol{E}_2$, respectively. As we follow the definition of graph union, formed representative graph is equal to the set of graphs. Therefore, we merge vertices and edges of graph union to compress the structure information. The vertices and edges are merged according to correspondence of vertices. Note that correspondences of edges is introduced by correspondences of vertices, that is, if an edge e_a corresponds to an edge e_b, vertices of e_a must correspond to vertices of e_b.

We merge vertices and edges according to $\mathcal{O}$ and update λ, ε and ω of graph union by equation (7), (8) and (9). The vertices and edges that do not correspond to any vertices and edges are added to the formed graph.

$$\lambda(\boldsymbol{v}) = \lambda(\mathcal{O}(\boldsymbol{v}_1)) + \lambda(\mathcal{O}(\boldsymbol{v}_2)), \tag{7}$$

$$\varepsilon(\boldsymbol{e}) = \varepsilon(\mathcal{O}(\boldsymbol{e}_1)) + \varepsilon(\mathcal{O}(\boldsymbol{e}_2)), \tag{8}$$

$$\omega(\boldsymbol{v}) = \frac{\omega(\mathcal{O}(\boldsymbol{v}_1)) + \omega(\mathcal{O}(\boldsymbol{v}_2))}{2}, \tag{9}$$

where $\mathcal{O}(\boldsymbol{v}_i)$ is a set of vertices of g_i corresponding

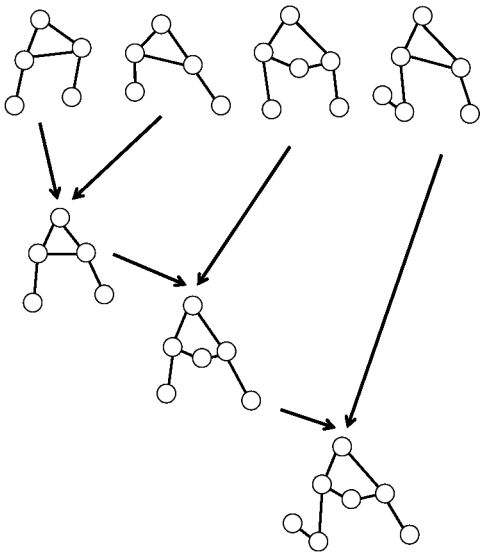


Fig. 4 The process of generating representative graph.

to $\boldsymbol{v}$. Fig. 3 shows a graph union of two graphs with merging corresponded vertices and edges. Finally, the generating algorithm is described follow.

1. Given $\boldsymbol{G}_t^k$, the algorithm commences with fixing common vertices from $\boldsymbol{G}_t^k$ and eq. (4) and (5).
2. Pick one graph from $\boldsymbol{G}_t^k$ and set it as representative graph $\mathcal{R}^k$.
3. Pick one graph g_{t_i} from $\boldsymbol{G}_t^k$.
4. Calculate correspondence of vertices between $\mathcal{R}^k$ and g_{t_i}.
5. Form a graph union of $\mathcal{R}^k$ and g_{t_i} with merging vertices and edges. Then set the formed graph union to new $\mathcal{R}^k$.
6. Repeat 3, 4 and 5 until $\boldsymbol{G}_t^k$ is empty.

At the end of this section, the process of generating a representative graph from four graphs is illustrated in Fig. 4.

3. Experimental Results

We carried out two experiments to evaluate the proposed method. First, we generate a representative graph from training data for each class, then evaluate the obtained representative graphs by recognition experiments. We used two database, IAM graph database repository[11] and the decorative character images.

The IAM database is a standard database to evaluate methods for graph-based pattern recognition and machine learning. The IAM database has three sets that have different distortion levels: low, medium and high. Each set has 750 graphs for training, validation and testing. The letters are 15 capital letters of the Roman alphabets that only consist of straight lines (A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z). Fig. 5 shows examples of graphs of IAM database.

The decorative character images are alphabets with various decorations. The database of decorative characters consists of 260 decorative character images. There are 26 classes containing 10 different decorated images. A hundred thirty images (each of 26 classes has 5 images) are used as trainings, others are used as testing. The graphs of decorative characters can be obtained by the following procedure. We convert decorative characters into global structures using multi-scale method[3]. In this method, topographical features such as ridges and ravines obtained from an intensity surface[4] are extracted from multi-scale images and used for structure extraction and interpolation. The topology of global structures becomes clear by thinning[12]. The end, curve and junction point of thinned global structure are defined as vertices. The lines are defined as edges. The examples of decorative characters, global structures and graphs are shown in Fig. 6. Retrieving graphs from decorative characters are quite sensitive so that the severe structure variation occurred even if in the same class. However, we think that it is important to observe how does the proposed method work in graphs of decorative characters.

3.1 Experiments for Learning

First of all, we carried out brief experiments to extract the representative graph by all the order of three graphs. Fig. 7 shows the results. From these results, we can see the representative graph shape changes according to the order of training data set.

We obtained the representative graphs from IAM database and decorative characters by the proposed method. Various representative graphs will be obtained depending on the order of training graphs since the proposed method is iterative. Therefore, we randomly generated 100 and 10 orders for each class from IAM database and decorative character database, respectively. We removed vertices and edges of which

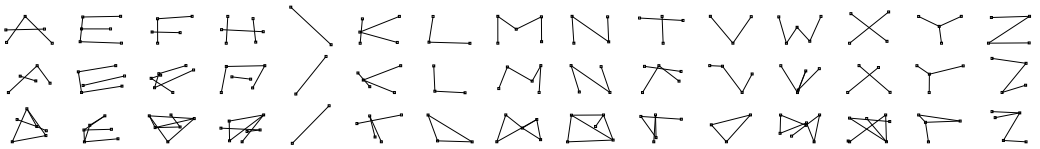


Fig. 5 Graphs of IAM database. The row of top, middle and bottom are low, medium and high distortion level, respectively.

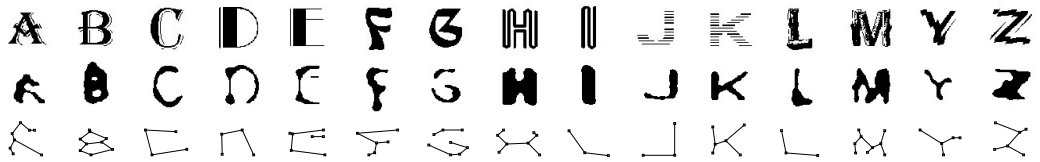


Fig. 6 Decorative characters, global structures and graphs are shown in top, middle and bottom row, respectively.

(a) 1 (b) 2 (c) 3 (d) (3, 2, 1) (e) (3, 1, 2) (f) (2, 3, 1) (g) (2, 1, 3) (h) (1, 2, 3) (i) (1, 3, 2)

Fig. 7 Samples and representative graphs. (a), (b) and (c) are samples. The numbers are idecies of samples. (d), (e), (f), (g), (h) and (i) are representative graphs and 3–tuples are orders.

weights are less than threshold since the vertices and edges which have small weight can be seen as noise, excepting for all weights of vertices or edges are 1. In this experiments, we set thresholds 5 and 2 for IAM database and decorative characters, respectively. Fig. 8 and 9 show some of results of obtained representative graphs from IAM database and decorative characters, respectively. The representative graph come to have various structures when structure distortion level increases.

3.2 Recognition Experiments

We carried out recognition experiments using obtained representative graphs. The test data is assigned to a class according to the dissimilarities between test data and representative graphs.

The distance between a graph g_i and a representative graph $\mathcal{R}^k$ is defined as

$$d(g_i, \mathcal{R}^k) = E(\eta) - E(\mathcal{R}^k) + C(\mathcal{R}^k, g_i), \qquad (10)$$

where η is a graph union of g_i and $\mathcal{R}^k$ obtained by the proposed method. The function E calculates how compact a graph union is. The E returns small value if graph union is compact, that is, the value will be small if the graph union is obtained from graphs which have similar structures. Therefore, $E(\eta) - E(\mathcal{R}^k)$ indicates the similarity between g_i and $\mathcal{R}^k$ on structure. Let $\hat{g}$ be a graph union obtained from $\boldsymbol{G}$. The value of E is defined as

$$E(g) = -\sum_{i=1}^{|\boldsymbol{V}|} \frac{\lambda(v_i)}{|\boldsymbol{G}|} \log \frac{\lambda(v_i)}{|\boldsymbol{G}|} - \sum_{j=1}^{|\boldsymbol{E}|} \frac{\varepsilon(e_j)}{|\boldsymbol{G}|} \log \frac{\varepsilon(e_j)}{|\boldsymbol{G}|}. \qquad (11)$$

The function C, third term of equation (10), is defined as the sum of matching cost of corresponding edges. The matching cost of two edges is

$$\kappa(e_i, e_j) = 2 l_i l_j \cos(\angle e_i e_j), \qquad (12)$$

where l_i denotes the length of e_i.

We calculated distances between test data and representative graphs by equation (10). In recognition experiments, the test data is regarded to be recognized correctly if the distance between the test data and its correct class is within N nearest.

The results of recognition experiments are shown in Fig. 10. The characteristic that the order of the training data affects the shape of the representative graph may be a weak point of the proposed method. Therefore we show the expected recognition performance of the representative graph. Those are the average, standard deviation, maximum and minimum recognition rate. In decorative characters, the averages of recognition performance are 25.3, 37.6, 46.2, 52.5 and 58 % at $N = 1, 2, 3, 4, 5$, respectively.

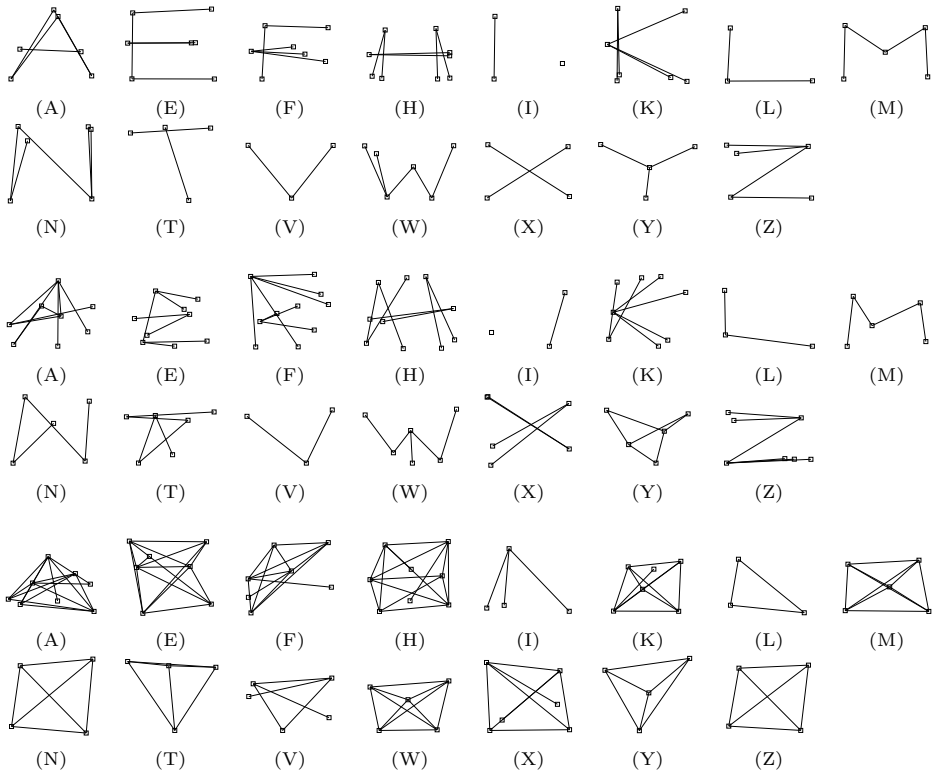


Fig. 8 Representative graphs from IAM database. 1st and 2nd, 3rd and 4th, 5th and 6th row are from low, medium and high, respectively.

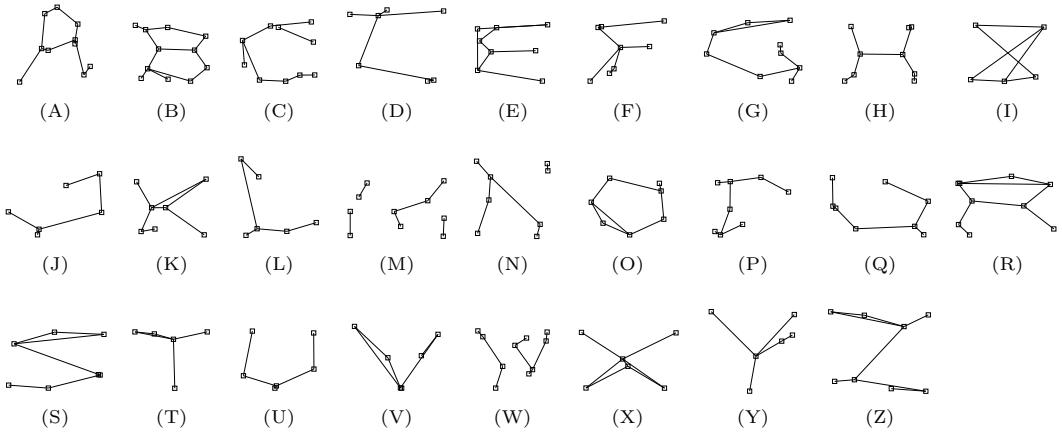
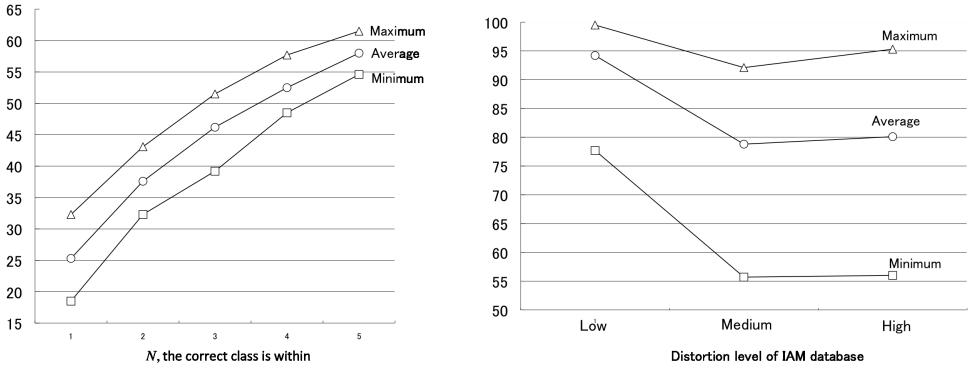


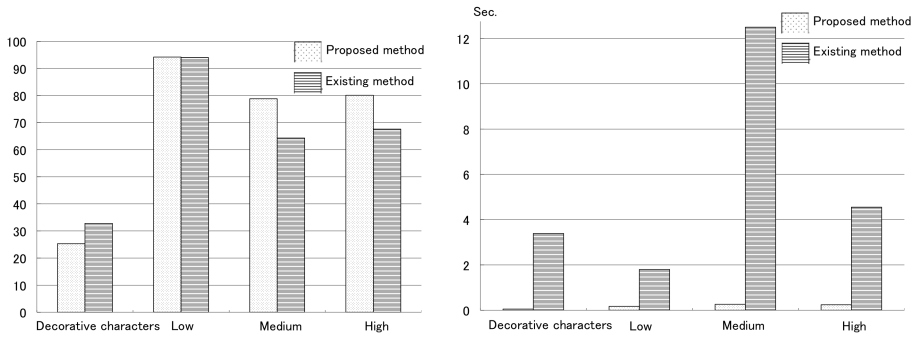
Fig. 9 Representative graphs from decorative characters.

The standard deviations of recognition performance are 4.3, 3.7, 3.8, 2.8 and 2.2 at $N = 1, 2, 3, 4, 5$, respectively. The proposed method could not find appropriate correspondences since ω are variously located. We must improve the extraction of graphs from decorative character images to apply the proposed method. In IAM database, the average recognition performances are 94.2, 78.8 and 80.1 %, in low, medium and high distortion datasets at $N = 1$, respectively. The standard deviations of recognition performance are 4.4, 4.4 and 3.3 in low, medium and high, respectively. These experimental results can deduce the consequences that the order of the training dataset affects the shape and the recognition perfor-



(a) Decorative characters (b) IAM database

Fig. 10 Results of recognition experiments.



(a) Recognition performance (b) Computational time

Fig. 11 Comparison of proposed and existing method.

mance of the representative graph. However, the influence on recognition performance is small since the standard deviations are small.

4. Discussions

We compared the proposed method and existing method[7] in performance of recognition and computational time for generating representative graphs.

We review the existing method briefly. Existing method proposed a weighted minimum common supergraph, or WMCS and treat WMCS as the representative graph. WMCS is defined as a common supergraph which has the minimum number of vertices among all common supergraphs. WMCS of two graphs is obtained by equation (13).

$$WMCS(g_1, g_2) = mcs(g_1, g_2) \cup (g_1 - mcs(g_1, g_2)) \cup (g_2 - mcs(g_1, g_2)), \quad (13)$$

where $mcs(g_1, g_2)$ is a common subgraph which has the maximum number of vertices among all subgraphs of g_1 and g_2. The $mcs(g_1, g_2)$ is calculated by the backtrack algorithm[9]. Note that the difference of graphs,$g_2 - g_1$ is obtained by removing g_1 from g_2, $g_1 \in g_2$. The operation, $\cup$, generates a graph union. The weights of vertices and edges of $WMCS(g_1, g_2)$ are calculated by $\lambda(v) = \lambda(v_1) + \lambda(v_2)$ if vertex v is generated from v_1 of g_1 and v_2 of g_2. The weights of edges are calculated by the same manner. Finally, WMCS of $\boldsymbol{G}^k$ is applied equation (13) iteratively. In the same as equation (10), the distance between test data and WMCS is defined as

$$d(g_i, \mathcal{W}^k) = E(WMC(g_i, \mathcal{W}^k)) - E(\mathcal{W}^k) + C(\mathcal{W}^k, g_i), \quad (14)$$

where $\mathcal{W}^k$ denotes WMCS of class k.

The comparison results are shown in Fig. 11. The recognition results of existing method are 32.8, 94, 64.3 and 67.6 %, and proposed method are 25.3, 94.2, 78.8 and 80.1 %, in decorative characters, low, medium and high, at $N = 1$, respectively. In the decorative characters, the both results are not high. While in IAM database, the proposed method is bet-

	μ	$\hat{\mu}$	n_v	n_e	n_e/n_v
IAM (Low)	0.87	0.92	4.7	3.1	0.66
IAM (Medium)	0.81	0.85	4.7	3.2	0.68
IAM (High)	0.82	0.86	4.7	4.4	0.94
Decorative characters	0.74	0.46	6.6	5.6	0.85

Table 1 Characteristics of graphs of IAM database and Decorative characters.

ter than the existing method at all distortion level. The existing method is superior than the proposed method in decorative characters, while the proposed method is superior than the existing method in IAM database. In order to analysis these results, it is necessary to make the difference between graphs of decorative characters and IAM database clear. Table 1 shows the characteristics of two databases. μ, $\hat{\mu}$, n_v and n_e are the averages of distance between vertices, distance between adjacent vertices, number of vertices and number of edges, respectively. The graphs of decorative characters are the graphs of which distance between adjacent vertices is small. We can see the value of n_e/n_v as ratio of noise edges. In the graphs of IAM database, the noise edge increases according to the distortion level. Especially, the number of noise edges of IAM (high) is greater than the decorative characters. Therefore, we can say the existing method is more robust than the proposed method for the graphs of which distances between adjacent vertices are small. On the other hand, the proposed method is more robust than the existing method for noise edges and the graphs of which distances between adjacent vertices are large.

Moreover, we can see the improvement of the proposed method at computational time. The computational time of existing method are 3.4, 1.8, 12.5 and 4.6 (s), and that of the proposed method are 0.1, 0.2, 0.3 and 0.2 (s), in decorative characters, low, medium and high, respectively. The proposed method is about 34 and 25 times faster at decorative characters and IAM database, respectively. The computational time of the proposed method is significantly better than the existing method.

5. Conclusions

We proposed a generating method for representative graphs. The proposed method extracts common vertices of training sets. Then the correspondences of vertices are calculated using characteristics of common vertices, that are locations, vertex type and connection between vertices. We evaluated the proposed method using IAM graph database and decorative characters. While there are problems to apply the proposed method for decorative character images, the proposed method obtained about 80% recognition performance in IAM graph database. Moreover, the significant improvement in computational time is obtained.

References

1) K. Iwata, K. Kato, K. Yamamoto : "Recognition of Handprinted Characters by Relaxation Matching with Improvement in Association for Segments", Proc. IEICE, General Conference pp.306, 1997.
2) J. Rocha, T. Pavlidis : "A Shape Analysis Model with Application to A Character Recognition System", IEEE Trans. Patt. Anal. Mach. Intell., vol.16, no.4, pp.393-404, 1994.
3) S. Omachi, M. Inoue, and H. Aso : "Structure Extraction from Decorated Characters using Multiscale Images", IEEE Trans. Patt. Anal. Mach. Intell., vol.23, no.3, pp.315-322, 2001.
4) R. M. Haralic, L.T. Watson, and T. J. Laffey : "The topographic primal sketch", International Journal of Robotics Research, vol.2, no.1, pp.50-72, 1983.
5) F. Harary : "Graph Theory", Addison-Wesley, Reeding, MA., 1994.
6) S. Megawa, S. Omachi, H. Aso : "Decorated Character Recognition Using Non-Deterministic Shape Analysis Model", Proc. Meeting on Image Recognition and Understanding, vol.1, pp.241-246, 2000.
7) H. Bunke, P. Foggia, C. Guidobaldi, and M. Vento : "Graph Clustering Using the Weighted Minimum Common Supergraph", Graph Based Representations in Pattern recognition, pp.235-246, 2003.
8) A. Torsello and E.R. Hancock : "Learning Shape-Classes Using a Mixture of Tree-Unions", IEEE Trans. Patt. Anal. Mach. Intell., vol.28, no.6, pp.954-967, 2006.
9) J.J. McGregor : "Backtrack Search Algorithms and the Maximal Common Subgraph Problem", Software Practice and Experience, vol.12, pp.23-34, 1982.
10) J.T. Tou and R.C. Gonzalez : "Pattern Recognition Principles", Addison-Wesley, 1974.
11) K. Riesen, and H. Bunke : "IAM graph database repository for graph based pattern recognition and machine learning", In N. da Vitoria Lobo et al., editor, Structural Syntactic, and Statical Pattern Recognition, LNCS 5342, pp.287-297, 2008.
12) L. Lam, S.W. Lee, and Y.S. Ching : "Thinning Methodologies - A Comprehensive Survey", IEEE Trans. Patt. Anal. Mach. Intell., Vol.14, No.9, pp.869-885, 1992.

(Received Nov. 10, 2011)
(Revised Mar. 10, 2011)

Tomo Miyazaki

He recieved B.E. degree from Department of Informatics, Faculty of Engineering, Yamagata University in 2006. He recieved M.E. and Ph.D. degrees from Graduate School of Engineering, Tohoku University in 2008 and 2011, respectively. He joined Hitachi, Ltd. in 2011.

Shinichiro Omachi (Member)

He received his B.E., M.E. and Doctor of Engineering degrees in Information Engineering from Tohoku University, Japan, in 1988, 1990 and 1993, respectively. He has worked as a research associate at the Education Center for Information Processing at Tohoku University from 1993 to 1996. Since 1996, he has been with the Graduate School of Engineering at Tohoku University, where he is currently a professor. From 2000 to 2001, he has been a visiting associate professor at Brown University. His research interests include pattern recognition, computer vision, information retrieval, data mining and parallel processing. He has received the MIRU Nagao Award and IAPR/ICDAR Best Paper Award in 2007, and ICFHR Best Paper Award in 2010. Dr. Omachi is a member of the IEEE, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, and the Japanese Society of Artificial Intelligence.