Development of Efficient and Robust Linkage Pattern Mining for Multiple Sequential Data

Kyosuke Maeda, Issei Yokota, Yoshifumi Okada and Saerom Lee

***Abstract*—Linkage pattern mining is a method used to extract frequently occurring patterns from multiple sequential data without considering the similarity or correlation between frequent patterns. Therefore, it is expected to be a promising approach for disease prediction and voice data analysis. In the previous method, closed itemset mining was introduced in the linkage pattern-mining algorithm to ensure robustness against noise. Although this method can extract linkage patterns from noisy artificial datasets, a reduction in computation time and stringent parameter settings are essential for its practical application to real data. In this study, we employed Episode Mining using Memory Anchor algorithm for frequent pattern mining to overcome the limitations of the previous method. The objective of this study is to develop a new robust linkage pattern-mining method that is more applicable to real data. A performance comparison between the previous and proposed methods using artificial datasets showed that the proposed method achieved a reduction in computation time while maintaining an extraction accuracy that is comparable to that of the previous method, particularly on noisy artificial datasets.**

***Index Terms*—closed itemset, EMMA, interval graph, linkage pattern, sequential pattern mining**

I. Introduction

Pattern mining is a general technique used in data mining to extract patterns with valuable information from large amounts of data. With the proliferation of big data in recent years, pattern mining has attracted attention for its application in data analysis. Sequential pattern mining identifies repeated patterns in sequential data by focusing on similarities and correlations. Since the introduction of sequential pattern mining by Agrawal et al. [1], various methods have been proposed [2]-[4]. These methods have been applied in a wide range of fields, including e-learning [5], medicine [6], and malware detection [7].

Among the sequential pattern-mining methods, linkage pattern mining extracts frequently occurring patterns from multiple sequences of data without using similarities or correlations [8]. This method targets multiple sequences and extracts groups of frequent patterns that appear repeatedly across sequential data as linkage patterns.

Lee et al. [9] proposed a linkage pattern-mining algorithm in which closed-itemset mining was introduced to make it robust against noise. Their method (hereinafter referred to as "previous method") enabled a certain degree of pattern extraction from artificial datasets containing noise. However, the problem with the previous method that uses the mining algorithm proposed by Mannila [10] is that the extraction of frequent patterns utilizes most of the total computation time, which is affected by the parameter settings.

This study aims to reduce the computation time required for frequent pattern mining, which accounts for the majority of the computation time of previous methods, for its application to real data with noise. To achieve this objective, we developed a new linkage pattern-mining method by replacing the algorithm proposed by Mannila [10], which was used to extract repeatedly occurring frequent patterns from sequential data in the previous method, with Episode Mining using Memory Anchor (EMMA) algorithm [11]. Because the EMMA algorithm searches sequential data by focusing only on frequent patterns, it is expected to be faster than the algorithm proposed by Mannila, which comprehensively searches for sequential data for frequent pattern mining. To demonstrate the efficiency of the proposed method, we compared the extraction accuracy and computation time of both methods using noisy and non-noisy artificial datasets.

The remainder of this paper is organized as follows. Section II defines the linkage pattern extracts used in the study. Section III explains the implementation of the proposed method. Section IV describes evaluation experiments conducted using artificial datasets. Section V presents the results and discusses the experiments, and Section VI summarizes the study.

II. Definition of Linkage Pattern

The linkage pattern to be extracted is defined according to Lee et al. [9] as follows.

First, we consider a frequent pattern, where S is a single sequence. In this case, $freq(S, \alpha)$ denotes the number of

Manuscript received March 7, 2024; revised December 25, 2024.

This paper was supported by Office for the Promotion of Diversity at Kagawa University.

K. Maeda is a postgraduate student of the Division of Science for Creative Emergence, Kagawa University Graduate School, 2-1, Saiwai-cho, Takamatsu, Kagawa 760-8523, Japan (e-mail: s23g362@kagawa-u.ac.jp).

I. Yokota is a postgraduate student of the Division of Science for Creative Emergence, Kagawa University Graduate School, 2-1, Saiwai-cho, Takamatsu, Kagawa 760-8523, Japan (e-mail: s24g212@kagawa-u.ac.jp).

Y. Okada is a professor of the College of Information and Systems, Muroran Institute of Technology, 27-1, Mizumoto-cho, Muroran, Hokkaido 050-8585, Japan (e-mail: okada@muroran-it.ac.jp).

S. Lee is an assistant professor of the Faculty of Engineering and Design, Kagawa University, 1-1, Saiwai-cho, Takamatsu, Kagawa 760-8523, Japan (corresponding author to provide email: lee.saerom@kagawa-u.ac.jp).

occurrences of the subsequence α in S. For a predefined constant value θ, α is a frequent pattern in S if $freq(S, \alpha) \geq \theta$. Let us assume that multiple sequential data sets are provided (Patterns A, B, and C) have already been extracted from these sequences. A group of frequent patterns is called a linkage pattern if the frequent patterns that occur in the sequential data during a certain timeframe satisfy the following two conditions.

1) One or more frequent patterns exist that partially or entirely overlap in the occurrence time of all frequent patterns.

2) A set of frequent patterns satisfying condition 1 occurs θ or more times along the sequential data.

For example, when $\theta = 2$, in Fig. 1, a group of patterns (i.e., Patterns A to C) exists with overlapping timings of occurrences (Condition 1), and these patterns occur three times (Condition 2); consequently, the group of frequent patterns A, B, and C is extracted as a linkage pattern.

III. Method

Fig. 2 illustrates the steps involved in the proposed method. The proposed method follows a flow similar to that of the previous method; however, the algorithm for extracting frequent patterns from each sequence (Fig. 2a) has been improved. A detailed description of the proposed method is presented in this section.

A. Preprocessing

First, normalization and discretization were performed on all sequential input data before frequent pattern extraction. For normalization, the sequential data were converted to values ranging from 0 to 1. In discretization, the normalized data are divided into D stages and assigned discrete values from 0 to D-1. In the proposed method, D was set to 50 stages, which is the same value used in the previous method.

B. Frequent Episode Extraction and Labeling

Next, the EMMA algorithm [11] was used to extract frequent episodes from the sequential data (Fig. 2a). EMMA is a frequent episode mining algorithm for single sequences. In this study, we used the EMMA algorithm proposed by Huang et al., and EMMA uses the minimum number of occurrences, θ (θ is a natural number ≥ 2), as a parameter.

The process flow of the EMMA algorithm is shown in Fig. 3. Events, episodes, and patterns are defined as follows: An event is recorded data at one time point. An episode is a combination of events with more than θ occurrences, and a pattern is the same label assigned to the same episode by labeling process. In Step 1, the input data are searched once, and the number of event occurrences is counted. In Step 2, the events whose number of occurrences in Step 1 is less than θ are deleted (in Fig. 3, $\theta = 2$ for explanation). In Step 3, a list recording the time points of event occurrence, called the Location List, is created. In Step 4, the Location List is used to identify larger episodes by recursively searching for episodes whose number of occurrences is greater than or equal to θ, where θ is the same value as that used in Step 2. The Bound List created in Step 4 is a list of episode occurrence locations in the form of [episode start location, episode end location]. In Fig. 3, an episode is identified by extending event {a} as an example. <{a}, {b}> is searched recursively, because the number of occurrences is greater than two when event {a} is extended with event {b}. Finally, all frequent episodes whose number of occurrences is more than θ are extracted as an output. The steps of the EMMA algorithm described above were modified for linkage pattern mining from those of the original EMMA algorithm.

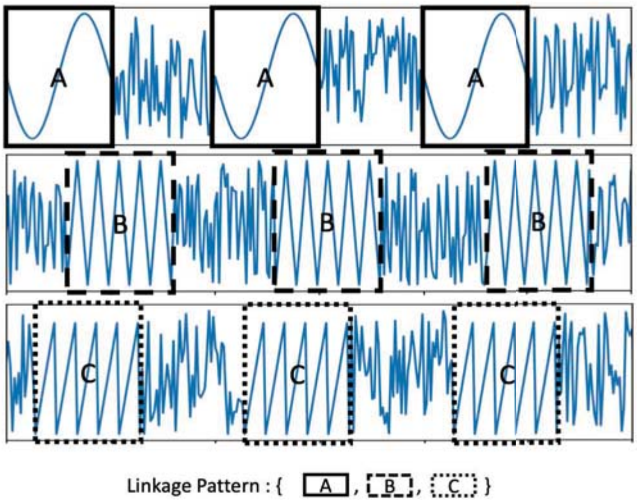


Fig. 1. Example of linkage pattern (redrawing based on [9])

The frequent episodes extracted using the EMMA algorithm were then labeled. In the labeling process, unique labels were assigned to each frequent episode after excluding frequent episodes with lengths less than two. If multiple patterns appeared during the same period, a label corresponding to a longer episode was assigned. In Fig. 3, episode <{a}, {b}> is assigned label 1, and output as pattern.

C. Interval Graph Generation

In this step, interval graphs were generated from the interval representations of the frequent patterns extracted in Subsection III.B. (Fig. 2b). An interval graph is a graph in which each labeled frequent pattern is associated with a node, and the overlap of any two labeled frequent patterns on a certain time axis between sequential data points is represented by an edge [12]–[14]. Thus, the set of frequent patterns that appear to be linked simultaneously among different sequential data is the interval graph.

D. Extraction of Linkage Patterns Based on Closed Itemset

In the previous method, closed-itemset mining was applied to exclude noise patterns that mistakenly appeared in the interval graph generation using robust linkage pattern mining. Fig. 2c illustrates the process by which noise patterns are removed from an interval graph. Each interval graph is indicated as a transaction, and a labeled frequent pattern, where each node in an interval graph is indicated as an item. The maximum closed itemset satisfying minsup is extracted from an itemset consisting of an interval graph by applying closed itemset mining. Finally, the most frequently occurring closed itemset is output as the linkage pattern. The randomly constructed noise patterns can be removed using closed-itemset mining.

Fig. 2c shows an example in which the noise patterns nA, nB, and nC are removed by closed-itemset mining, and only the precise linkage patterns {A, B, C} are extracted. In the

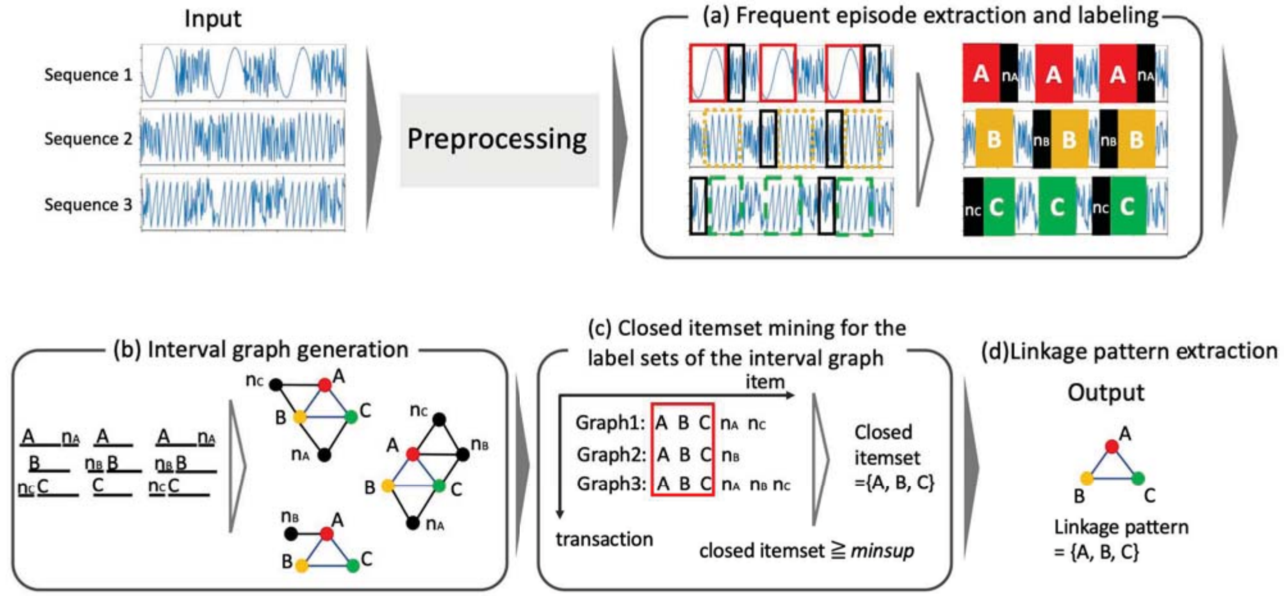
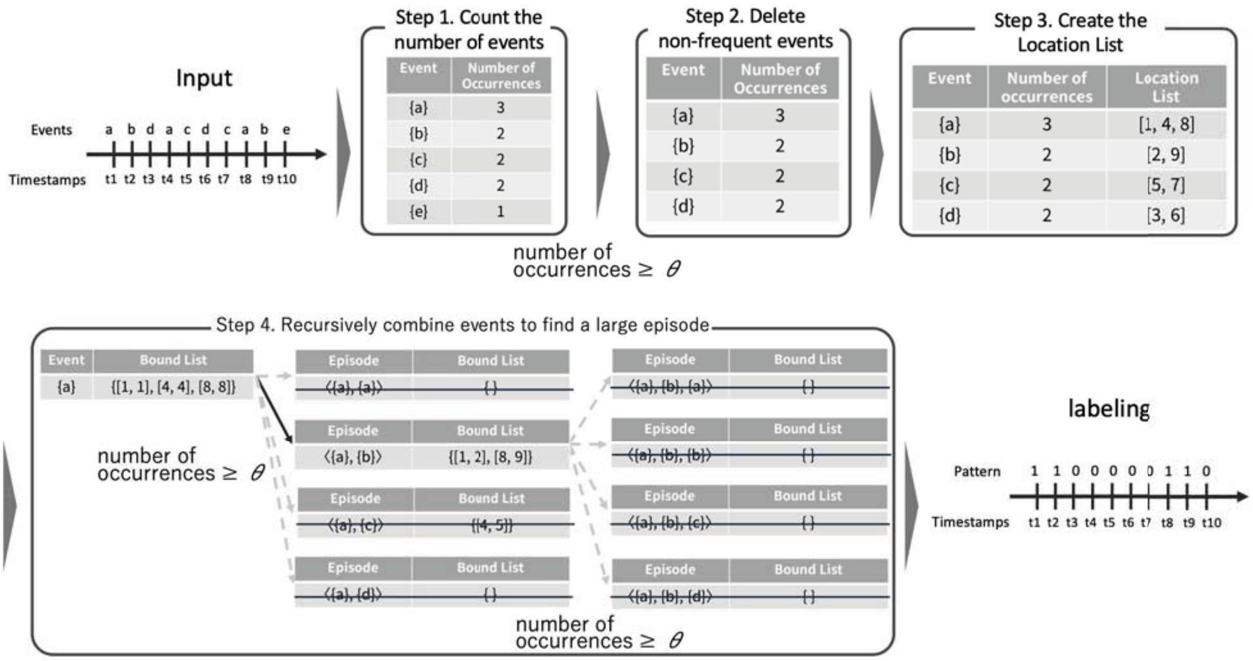


Fig. 2. Procedure of the proposed method (redrawing based on [9])


Input
Events a b d a c d c a b e
Timestamps t1 t2 t3 t4 t5 t6 t7 t8 t9 t10
Step 1. Count the number of events
Step 2. Delete non-frequent events
Step 3. Create the Location List
number of occurrences ≥ θ
Step 4. Recursively combine events to find a large episode
number of occurrences ≥ θ
number of occurrences ≥ θ
labeling
Pattern 1 1 0 0 0 0 0 1 1 0
Timestamps t1 t2 t3 t4 t5 t6 t7 t8 t9 t10

Step 1. Count the number of events

Event	Number of Occurrences
{a}	3
{b}	2
{c}	2
{d}	2
{e}	1

Step 2. Delete non-frequent events

Event	Number of Occurrences
{a}	3
{b}	2
{c}	2
{d}	2

Step 3. Create the Location List

Event	Number of occurrences	Location List
{a}	3	[1, 4, 8]
{b}	2	[2, 9]
{c}	2	[5, 7]
{d}	2	[3, 6]

Event	Bound List
{a}	{[1, 1], [4, 4], [8, 8]}

Episode	Bound List
~~⟨{a}, {a}⟩~~	~~{}~~
⟨{a}, {b}⟩	{[1, 2], [8, 9]}
~~⟨{a}, {c}⟩~~	~~{[4, 5]}~~
~~⟨{a}, {d}⟩~~	~~{}~~

Episode	Bound List
~~⟨{a}, {b}, {a}⟩~~	~~{}~~
~~⟨{a}, {b}, {b}⟩~~	~~{}~~
~~⟨{a}, {b}, {c}⟩~~	~~{}~~
~~⟨{a}, {b}, {d}⟩~~	~~{}~~

Fig. 3. Procedure of Episode Mining using Memory Anchor algorithm and labeling

proposed method, the linear time closed-itemset miner algorithm [15], which is a fast and exhaustive closed-itemset mining algorithm, was used in the same manner as in the previous method.

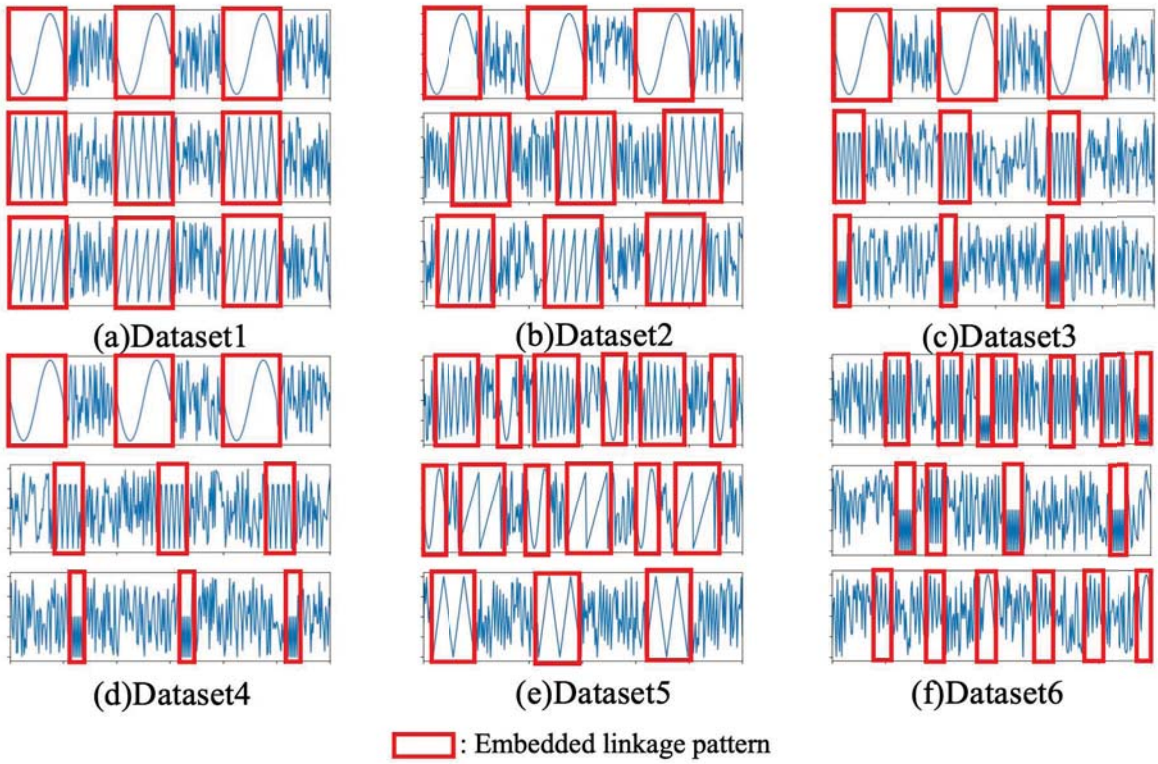
IV. Experiments

In this study, the proposed method was evaluated and compared with previous methods in terms of extraction accuracy and computation time using artificially created sequential datasets. Note that the artificial datasets used in these experiments are new additions to the datasets used by Lee et al.

A. *Artificial Datasets*

The artificial datasets used in these experiments are described by Lee et al. [9] as follows:

Each artificial dataset has a data length of 1000 points per sequence and comprises three sequential sets of data. The datasets were generated by inserting 10 linkage patterns (embedded linkage patterns) into random sequential data created using uniform random numbers. For the experiments, five non-noisy artificial datasets (Dataset 1 to Dataset 5) and five noisy datasets (Dataset1_noise to Dataset5_noise), which were fluctuations based on normal random numbers with a standard deviation (SD) of 0.01, were added to each non-noisy dataset created. Fig. 4 shows a cross-section of each non-noisy artificial dataset. In Dataset 1, frequent patterns of the same length are embedded with identical start times across the three sequential sets of data (Fig. 4a). In



(a)Dataset1 (b)Dataset2 (c)Dataset3

(d)Dataset4 (e)Dataset5 (f)Dataset6

Fig. 4. Artificial datasets (redrawing based on [9])

Dataset 2, frequent patterns of the same length are embedded with different start times across three sequential sets of data (Fig. 4b). In Dataset 3, frequent patterns of different lengths are embedded with identical start times for three sequential sets of data (Fig. 4c). In Dataset 4, frequent patterns of different lengths are embedded with different start times in each of the three sequential sets of data (Fig. 4d). In Dataset 5, a few types of frequent patterns are embedded with different lengths and different start times in each of the three sequential sets of data (Fig. 4e).

In these experiments, new datasets—Dataset 6 and dataset 6_noise—were also added for application to real data. Dataset 6 has a few types of short-length frequent patterns embedded with different lengths and different start times in each of the three sequential sets of data. In Dataset 6, five different types of embedded linkage patterns are contained to make it more complex and the extraction of patterns more difficult than other datasets. Dataset6_noise results from fluctuations being added to Dataset6 with an SD of 0.01.

B. Parameter Settings

Experiments were conducted to compare the previous and proposed methods by setting the best-performing parameters for each method. The best-performing parameters were selected using a grid search. The previous method set the minimum number of occurrences θ and window width w, which are parameters typically used for frequent pattern extraction, and the minimum number of occurrences *minsup*, which is used in closed-itemset mining. Conversely, the proposed method does not require w; thus, only θ and *minsup* were set. In the experiments, the computation time was represented as the average computation time for 10 runs. The parameter with the highest F-measure was also selected as the optimal parameter for determining extraction accuracy. Table I lists the parameters used in the evaluation experiments. Tables Ia and Ib list the parameters used to evaluate the extraction accuracies for the non-noisy and noisy datasets, respectively. Similarly, Tables Ic and Id show the parameters used to evaluate the computation time for the noisy and non-noisy datasets, respectively.

C. Extraction Accuracy of Linkage Patterns

The previous and proposed methods were compared in terms of their accuracy in extracting embedded linkage patterns using the artificial dataset described in Section IV.A. Precision, recall, and F-measure were used as the evaluation indices. These values were calculated as follows:

$$Precision = \frac{CDP}{DDP}$$

$$Recall = \frac{CDP}{EDP}$$

$$F-measure = \frac{2 * Precision * Recall}{(Precision + Recall)}$$

where CDP represents the number of data points in the embedded linkage patterns correctly detected by each method, DDP represents the number of data points in the embedded linkage patterns extracted by each method, and EDP represents the total number of data points in the embedded linkage patterns in each dataset.

D. Overall Experiments Flow

The evaluation experiments were conducted as follows. Extraction accuracies and computation time were compared for each case.

Case 1 comprises experiments that were conducted in the same manner as the previous method's evaluation experiments using non-noisy and noisy datasets (Dataset1–6 and Dataset1_noise–Dataset6_noise), respectively.

Case 2 comprises experiments that were conducted on Dataset1_noise–Dataset6_noise, increasing the data length from 1000 to 10000 in increments of 1000 while the SD was fixed at 0.01.

TABLE I
PARAMETERS USED IN THE EXPERIMENTS

(a) Parameter used to evaluate extraction accuracies (without noise)

	Previous Method			Proposed Method	
	θ	*w*	*minsup*	θ	*minsup*
Dataset1	5	5	5	4	3
Dataset2	5	5	5	5	2
Dataset3	5	3	5	5	4
Dataset4	5	5	5	5	3
Dataset5	3	5	5	4	2
Dataset6	9	5	5	5	3

(b) Parameter used to evaluate extraction accuracies (with noise)

	Previous Method			Proposed Method	
	θ	*w*	*minsup*	θ	*minsup*
Dataset1_noise	6	3	2	3	2
Dataset2_noise	7	3	2	4	2
Dataset3_noise	7	3	2	4	2
Dataset4_noise	7	3	2	4	2
Dataset5_noise	5	3	2	3	2
Dataset6_noise	7	3	2	4	2

(c) Parameter used to evaluate computation time (without noise)

	Previous Method			Proposed Method	
	θ	*w*	*minsup*	θ	*minsup*
Dataset1	10	3	10	10	10
Dataset2	10	3	9	10	9
Dataset3	10	3	10	10	4
Dataset4	10	3	8	10	6
Dataset5	10	3	10	10	8
Dataset6	10	3	10	10	2

(d) Parameter used to evaluate computation time (with noise)

	Previous Method			Proposed Method	
	θ	*w*	*minsup*	θ	*minsup*
Dataset1_noise	10	3	10	10	9
Dataset2_noise	10	3	10	10	10
Dataset3_noise	10	3	9	10	9
Dataset4_noise	10	3	10	10	5
Dataset5_noise	10	3	10	10	6
Dataset6_noise	10	3	10	9	3

TABLE II
COMPARISON OF THE EXTRACTION ACCURACIES OF THE PREVIOUS AND PROPOSED METHODS ON NON-NOISY DATASETS

		Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	Dataset6
Precision	Previous Method	1	1	1	1	1	1
	Proposed Method	1	1	1	1	1	1
Recall	Previous Method	1	1	1	1	1	1
	Proposed Method	1	1	1	1	1	1
F-measure	Previous Method	1	1	1	1	1	1
	Proposed Method	1	1	1	1	1	1

TABLE III
COMPARISON OF THE EXTRACTION ACCURACIES OF THE PREVIOUS AND PROPOSED METHODS ON NOISY DATASETS

		Dataset1_n	Dataset2_n	Dataset3_n	Dataset4_n	Dataset5_n
Precision	Previous Method	0.880	0.879	0.845	0.820	0.812
	Proposed Method	0.932	0.947	0.904	0.952	0.914
Recall	Previous Method	0.929	0.903	0.860	0.838	0.913
	Proposed Method	0.774	0.778	0.752	0.739	0.775
F-measure	Previous Method	0.904	0.891	0.852	0.829	0.860
	Proposed Method	0.845	0.854	0.821	0.832	0.838

Case 3 comprises experiments that were conducted on Dataset1_noise–Dataset6_noise, increasing the noise by SD=0.01 to SD=0.1 in increments of 0.01 while the data length was fixed at 1000.

V. RESULTS AND DISCUSSION

The results for Case 1–Case 3 are explained in order. Note that the results in each Table is rounded to the third decimal place. Note that noisy datasets are marked with "_noise" or "_n" in the table and figure.

A. Results and Discussion on Case 1

1) Extraction accuracies

Tables II and III show the extraction accuracies of the previous and proposed methods on non-noisy and noisy artificial datasets, respectively. Table II shows that the linkage patterns were extracted perfectly from the non-noisy datasets using both the previous and proposed methods. Conversely, for noisy datasets (Table III), although the precision marginally improved for all datasets, the recall and F-measure were lower than those of the previous method.

One reason for the higher recall of the previous method is that a search with a small *w* increases comprehensiveness and reduces the number of missed frequent patterns. The previous method performed an exhaustive search; however, its precision was reduced by the extraction of pseudo-patterns accidentally created by noise. In the proposed method, frequent patterns were searched by combining episodes. Therefore, pseudo-patterns were not extracted, and a higher precision was achieved than that of the previous method. However, the proposed method is less exhaustive than the previous method, which is thought to have reduced recall and affected the F-measure as well.

2) Computation time

Tables IV and V show a comparison of the computation time of the previous and proposed methods on non-noisy and noisy artificial datasets, respectively. From Table IV, we can observe that the computation time of the proposed method is

TABLE IV
COMPARISON OF THE COMPUTATION TIME OF THE PREVIOUS AND PROPOSED METHODS ON NON-NOISY DATASETS

	computation time (unit: seconds)					
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	Dataset6
Previous Method	3.067	2.901	2.524	2.178	3.486	2.057
Proposed Method	6.211	6.509	3.079	3.109	4.509	0.682

TABLE V
COMPARISON OF THE COMPUTATION TIME OF THE PREVIOUS AND PROPOSED METHODS ON NOISY DATASETS

	computation time (unit: seconds)					
	Dataset1_n	Dataset2_n	Dataset3_n	Dataset4_n	Dataset5_n	Dataset6_n
Previous Method	2.005	2.562	1.679	1.431	2.057	1.053
Proposed Method	0.480	0.338	0.248	0.238	0.194	0.226

TABLE VI
COMPARISON OF THE PRECISION IN DIFFERENT DATA LENGTHS

		data length									
		1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Dataset1_n	Previous Method	0.880	0.873	0.873	0.873	0.873	0.873	0.873	0.873	0.873	0.873
	Proposed Method	0.932	0.924	0.924	0.924	0.924	0.924	0.924	0.924	0.924	0.924
Dataset2_n	Previous Method	0.879	0.832	0.832	0.832	0.832	0.832	0.832	0.832	0.832	0.832
	Proposed Method	0.947	0.903	0.903	0.903	0.903	0.903	0.903	0.903	0.903	0.903
Dataset3_n	Previous Method	0.845	0.837	0.837	0.837	0.837	0.837	0.837	0.837	0.837	0.837
	Proposed Method	0.904	0.906	0.906	0.906	0.906	0.906	0.906	0.906	0.906	0.906
Dataset4_n	Previous Method	0.820	0.817	0.817	0.817	0.817	0.817	0.817	0.817	0.817	0.817
	Proposed Method	0.952	0.934	0.934	0.934	0.934	0.934	0.934	0.934	0.934	0.934
Dataset5_n	Previous Method	0.812	0.806	0.806	0.806	0.806	0.806	0.806	0.806	0.806	0.806
	Proposed Method	0.914	0.905	0.905	0.905	0.905	0.905	0.905	0.905	0.905	0.905
Dataset6_n	Previous Method	0.831	0.823	0.823	0.823	0.823	0.823	0.823	0.823	0.823	0.823
	Proposed Method	0.920	0.919	0.919	0.919	0.919	0.919	0.919	0.919	0.919	0.919

3 s longer computation time than that of the previous method for Dataset 1 and Dataset 2, which is approximately double the computation time of the previous method. However, the computation time of the proposed method is only approximately 1 s longer than that of the previous method for Dataset 3 to Dataset 5. Conversely, the proposed method has a shorter computation time than that of the previous method on Dataset 6. Table V shows a comparison of the computation time for the noisy datasets. Using the proposed method, linkage patterns could be extracted in less than half the computation time of the previous method for all noisy datasets.

The difference in computation time between the previous and proposed methods is attributed to the differences in the episode mining methods, the Mannila`s algorithm and EMMA. The algorithm proposed by Mannila searches for a sequence multiple times while extending the w width. By contrast, EMMA searches for a sequence once and then combines and extends the episodes using a recursive processing based on the searching results. Therefore, when using the previous method, the computation time depended on the length of the sequential data, whereas in the proposed method, it depended on the length of the embedded linkage pattern. The difference between the two algorithms causes the computation time of the previous method to be shorter than the proposed method on non-noisy datasets because the search space of the algorithm proposed by Mannila is smaller than that of EMMA by being absent of noise. Conversely, the search space of EMMA is smaller than that of the algorithm proposed by Mannila on noisy datasets because short episodes are extracted owing to noise. Thus, the computation time of the proposed method was significantly reduced on noisy datasets.

Furthermore, the proposed method exhibits the potential to further reduce the computation time. For example, the algorithm can be improved to terminate a search when it retrieves a subset that is included in the largest set of episodes that are currently held. By improving the program in this manner, the proposed method is expected to extract linkage patterns faster than the previous method even for non-noisy datasets.

B. *Results and Discussion on Case 2*

Before conducting the evaluation experiments of Case 2, a grid search was conducted to find the best-performing parameter to fit the long length of datasets again. In the results of the grid search, constant high extraction accuracies were obtained for each dataset, independent of the data length by increasing θ by a specific width. Therefore, the two experiments described below were conducted using the parameters obtained from this grid search, not those shown in Table I.

1) *Extraction accuracies*

Tables VI, VII, and VIII present the precision, recall, and F-measure of the previous and proposed methods, respectively, on noisy artificial datasets with different data lengths. As shown in Table VI, for all datasets, the precision was higher in the proposed method than in the previous method, even when the data length changed. Additionally, from Tables VII and VIII, the recall of the proposed method tends to be higher after a data length of 2000, and as a result, the F-measure of the proposed method outperforms that of the previous method after a data length of 2000. For this reason, the proposed method outperforms the previous method for datasets with long data lengths, such as real data in terms of extraction accuracies.

TABLE VII
COMPARISON OF THE RECALL IN DIFFERENT DATA LENGTHS

		data length									
		1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Dataset1_n	Previous Method	0.929	0.946	0.946	0.946	0.946	0.946	0.946	0.946	0.946	0.946
	Proposed Method	0.774	0.952	0.952	0.952	0.952	0.952	0.952	0.952	0.952	0.952
Dataset2_n	Previous Method	0.903	0.933	0.933	0.933	0.933	0.933	0.933	0.933	0.933	0.933
	Proposed Method	0.778	0.947	0.947	0.947	0.947	0.947	0.947	0.947	0.947	0.947
Dataset3_n	Previous Method	0.860	0.869	0.869	0.869	0.869	0.869	0.869	0.869	0.869	0.869
	Proposed Method	0.752	0.836	0.836	0.836	0.836	0.836	0.836	0.836	0.836	0.836
Dataset4_n	Previous Method	0.838	0.858	0.858	0.858	0.858	0.858	0.858	0.858	0.858	0.858
	Proposed Method	0.739	0.818	0.818	0.818	0.818	0.818	0.818	0.818	0.818	0.818
Dataset5_n	Previous Method	0.913	0.949	0.949	0.950	0.950	0.950	0.950	0.950	0.950	0.950
	Proposed Method	0.775	0.901	0.901	0.901	0.901	0.901	0.901	0.901	0.901	0.901
Dataset6_n	Previous Method	0.884	0.892	0.892	0.892	0.892	0.892	0.892	0.892	0.892	0.892
	Proposed Method	0.758	0.855	0.855	0.855	0.855	0.855	0.855	0.855	0.855	0.855

TABLE VIII
COMPARISON OF THE F-MEASURE IN DIFFERENT DATA LENGTHS

		data length									
		1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
Dataset1_n	Previous Method	0.904	0.908	0.908	0.908	0.908	0.908	0.908	0.908	0.908	0.908
	Proposed Method	0.845	0.938	0.938	0.938	0.938	0.938	0.938	0.938	0.938	0.938
Dataset2_n	Previous Method	0.891	0.880	0.880	0.880	0.880	0.880	0.880	0.880	0.880	0.880
	Proposed Method	0.854	0.924	0.924	0.924	0.924	0.924	0.924	0.924	0.924	0.924
Dataset3_n	Previous Method	0.852	0.853	0.853	0.853	0.853	0.853	0.853	0.853	0.853	0.853
	Proposed Method	0.821	0.870	0.870	0.870	0.870	0.870	0.870	0.870	0.870	0.870
Dataset4_n	Previous Method	0.829	0.837	0.837	0.837	0.837	0.837	0.837	0.837	0.837	0.837
	Proposed Method	0.832	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872
Dataset5_n	Previous Method	0.860	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872
	Proposed Method	0.838	0.903	0.903	0.903	0.903	0.903	0.903	0.903	0.903	0.903
Dataset6_n	Previous Method	0.857	0.857	0.857	0.857	0.857	0.857	0.857	0.857	0.857	0.857
	Proposed Method	0.831	0.886	0.886	0.886	0.886	0.886	0.886	0.886	0.886	0.886

2) Computation time

Fig. 5 presents a comparison of the computation time of the previous and proposed methods on data length-changed noisy artificial datasets. From Fig. 5, for all datasets, the computation time of the proposed method is shorter than that of the previous method. The computation time of the proposed method is shorter than that of the previous method because of the episode mining advantage of the proposed method described in Section V.A.2, even when the data length is changed. Note that in terms of time complexity, the proposed method is expected to eventually become larger than the previous method due to the property of the algorithm. Although the recursive process is becoming a bottleneck because the embedding pattern increases linearly with increasing data length on artificial datasets, real data does not necessarily show a linear increase in the length of the linkage pattern with increasing data length, the proposed method is expected to be more significant in terms of computation time than the previous method. The results of Case 2 experiments indicate that the proposed method outperforms the previous method in both extraction accuracies and computation time on large-scale datasets, such as real data.

C. Results and Discussion on Case 3

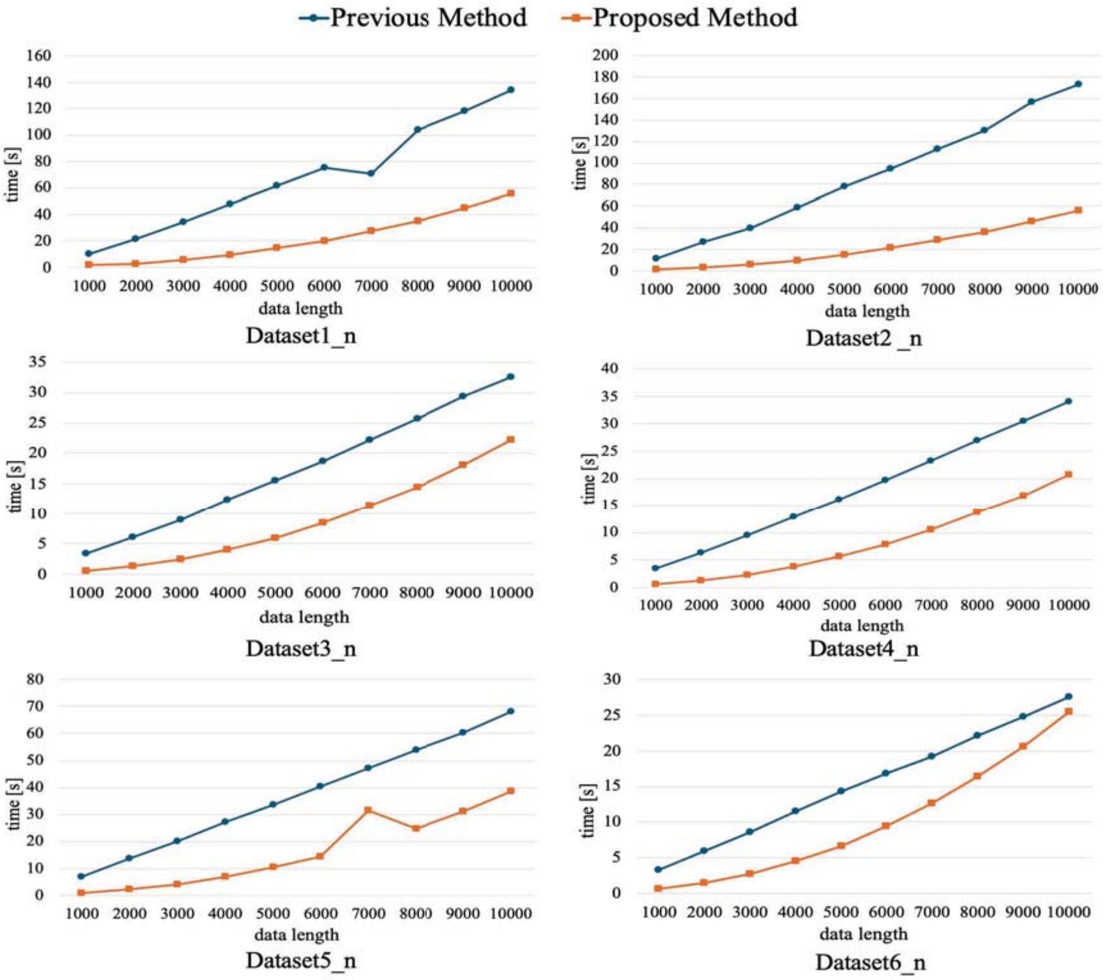
1) Extraction accuracies

Tables IX, X, and XI show the precision, recall, and F-measure of the previous and proposed methods on noise-changed artificial datasets, respectively. From Table IX, for all datasets, although the precision of the proposed method is higher than that of the previous method. As shown in Tables X and XI, the recall and F-measure are lower than those of the previous method. These results are consistent with the result of Section V.A.1 and are due to the fact that as in Section V.A.1, Mannila's algorithm exhaustively searches for patterns, whereas EMMA searches for exact patterns. However, it should be noted that even if real data are the target, large noise is not realistic because linkage pattern mining is excuted after noise is removed by multiple preprocessing step, such as smoothing or detrending. Furthermore, another reason why large noise is not realistic is that the linkage pattern with large noise added may not satisfy the definition of the linkage pattern because each embedded frequent pattern is identified as different patterns by affecting noise.

2) Computation time

Fig. 6 compares the computation time of the previous and proposed methods on noise-changed artificial datasets. Note that the parameters that are used in these experiments were not parameters for computation time (Table Id), but parameters for extraction accuracies (Table Ib) were used because when the noise is large, linkage patterns were not extracted under the computation time parameters. From Fig. 6, for all datasets, the computation time of the proposed method is shorter than that of the previous method. Because the patterns extracted decrease as the noise increases, the computation time tends to decrease for both methods.

In summary, the proposed method reduced the computation time while maintaining an extraction accuracy comparable to that of the previous method. The results of Section V.B show that the proposed method outperforms the previous method in both extraction accuracies and computa-



Dataset1_n

Dataset2 _n

Dataset3_n

Dataset4_n

Dataset5_n

Dataset6_n

Fig. 5. Comparison of the computation time in different data length

TABLE IX
COMPARISON OF THE PRECISION IN DIFFERENT NOISE

		noise level									
		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
Dataset1_n	Previous Method	0.880	0.780	0.752	0.731	0.695	0.673	0.655	0.630	0.595	0.590
	Proposed Method	0.936	0.874	0.822	0.773	0.735	0.726	0.678	0.638	0.617	0.617
Dataset2_n	Previous Method	0.879	0.810	0.817	0.775	0.745	0.721	0.679	0.675	0.676	0.670
	Proposed Method	0.947	0.913	0.891	0.841	0.805	0.768	0.735	0.725	0.691	0.709
Dataset3_n	Previous Method	0.845	0.842	0.753	0.686	0.603	0.687	0.496	0.479	0.470	0.415
	Proposed Method	0.904	0.907	0.827	0.761	0.702	0.691	0.621	0.537	0.516	0.510
Dataset4_n	Previous Method	0.820	0.750	0.704	0.672	0.652	0.549	0.514	0.506	0.527	0.423
	Proposed Method	0.952	0.808	0.860	0.768	0.745	0.587	0.679	0.573	0.560	0.458
Dataset5_n	Previous Method	0.812	0.728	0.703	0.663	0.643	0.612	0.624	0.590	0.593	0.566
	Proposed Method	0.914	0.812	0.765	0.712	0.712	0.676	0.667	0.652	0.663	0.617
Dataset6_n	Previous Method	0.812	0.842	0.753	0.686	0.603	0.687	0.496	0.479	0.470	0.415
	Proposed Method	0.914	0.878	0.833	0.804	0.678	0.531	0.381	0.428	0.430	0.381

TABLE X
COMPARISON OF THE RECALL IN DIFFERENT NOISE

		noise level									
		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
Dataset1_n	Previous Method	0.929	0.781	0.724	0.570	0.534	0.484	0.485	0.433	0.480	0.444
	Proposed Method	0.774	0.759	0.723	0.590	0.525	0.466	0.445	0.399	0.444	0.353
Dataset2_n	Previous Method	0.903	0.743	0.546	0.464	0.346	0.365	0.271	0.392	0.363	0.307
	Proposed Method	0.778	0.665	0.469	0.315	0.273	0.231	0.152	0.172	0.170	0.158
Dataset3_n	Previous Method	0.860	0.649	0.541	0.472	0.377	0.390	0.315	0.272	0.295	0.223
	Proposed Method	0.752	0.552	0.408	0.347	0.243	0.222	0.218	0.133	0.181	0.115
Dataset4_n	Previous Method	0.838	0.652	0.582	0.475	0.449	0.342	0.373	0.297	0.285	0.267
	Proposed Method	0.739	0.568	0.461	0.361	0.280	0.161	0.202	0.142	0.170	0.113
Dataset5_n	Previous Method	0.913	0.732	0.621	0.540	0.575	0.527	0.561	0.496	0.607	0.516
	Proposed Method	0.775	0.625	0.501	0.409	0.414	0.380	0.358	0.319	0.391	0.302
Dataset6_n	Previous Method	0.913	0.649	0.541	0.472	0.377	0.390	0.315	0.272	0.295	0.223
	Proposed Method	0.775	0.565	0.422	0.286	0.260	0.121	0.072	0.095	0.095	0.085

TABLE XI
COMPARISON OF THE F-MEASURE IN DIFFERENT NOISE

		noise level									
		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
Dataset1_n	Previous Method	0.904	0.781	0.737	0.640	0.604	0.563	0.557	0.513	0.531	0.507
	Proposed Method	0.845	0.813	0.769	0.669	0.613	0.568	0.537	0.491	0.517	0.449
Dataset2_n	Previous Method	0.891	0.775	0.654	0.581	0.473	0.484	0.388	0.495	0.473	0.421
	Proposed Method	0.854	0.770	0.614	0.458	0.407	0.356	0.252	0.278	0.273	0.258
Dataset3_n	Previous Method	0.852	0.733	0.630	0.559	0.464	0.497	0.385	0.347	0.363	0.290
	Proposed Method	0.821	0.686	0.546	0.476	0.361	0.336	0.323	0.213	0.268	0.187
Dataset4_n	Previous Method	0.829	0.698	0.637	0.557	0.532	0.422	0.432	0.374	0.370	0.327
	Proposed Method	0.832	0.667	0.601	0.491	0.407	0.253	0.312	0.228	0.261	0.181
Dataset5_n	Previous Method	0.860	0.730	0.659	0.595	0.607	0.566	0.591	0.539	0.600	0.540
	Proposed Method	0.838	0.707	0.606	0.520	0.523	0.487	0.466	0.429	0.492	0.405
Dataset6_n	Previous Method	0.860	0.733	0.630	0.559	0.464	0.497	0.385	0.347	0.363	0.290
	Proposed Method	0.838	0.688	0.560	0.422	0.376	0.197	0.121	0.155	0.155	0.139

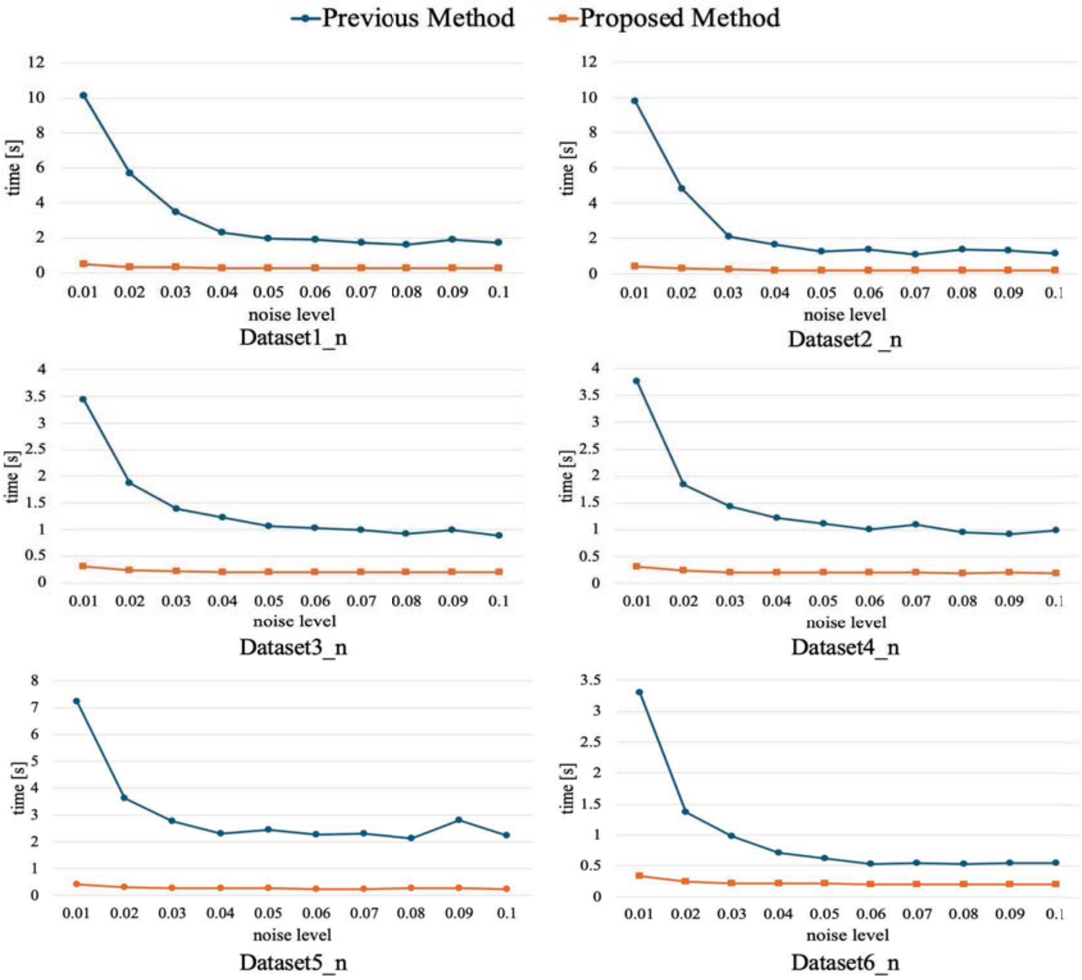


Fig. 6. Comparison of the computation time in different noise

tion time, especially when targeting datasets with long length data such as real data. Therefore, the problem that the previous method had in applying linkage pattern mining to real data is solved by the proposed method.

VI. CONCLUSION

In this study, we have proposed an efficient and robust linkage pattern-mining method to reduce the computation time and number of parameters, which are the limitations of the previous method. We employed EMMA as a search method for frequent episodes to reduce the search space and dimensionality of configuration parameters, as compared to those of the previous method that uses Mannila for episode mining. In the experiments, the best parameters were selected based on a grid search, and the extraction accuracy and computation time of the previous and proposed methods were compared and evaluated using artificial datasets. The experimental results showed that the proposed method exhibited the same level of extraction accuracy as the previous method, although the number of parameters was reduced by one. Moreover, a comparison of computation time showed a significant reduction in computation time for the noisy datasets. Particularly, when the data length of the dataset was increased, the proposed method outperformed the previous method in both extraction accuracy and computation time. This implies that the significant increase in the computation time depending on the data length, which is a limitation of the previous method, was resolved, and the results are expected to be applicable to large real data with noise.

In the future, we will apply the proposed method to real sequence data with noise, such as vital, sensor, and log data, and evaluate the practicality of the proposed method in terms of extraction accuracy and computation time.

REFERENCES

[1] R. Agrawal, and R. Srikant, "Mining sequential patterns," Proceedings of the Eleventh International Conference on Data Engineering. IEEE, 1995. pp. 3-14.

[2] J. Han, et al, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth," Proceedings of the 17th International Conference on Data Engineering. IEEE, 2001. pp. 215-224.

[3] M. J. Zaki, "SPADE: An efficient algorithm for mining frequent sequences," Machine Learning vol. 42, Springer. 2001. pp. 31-60.

[4] P. Fournier-Viger, J. C. W. Lin, R. U. Kiran, Y. S. Koh, and R. Thomas, R, "A survey of sequential pattern mining," Data Science and Pattern Recognition vol. 1, no. 1, 2017. pp. 54-77.

[5] J. K. Tarus, Z. Niu, and A. Yousif, "A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining," Future Generation Computer Systems vol. 72, Elsevier. 2017. pp. 37-48.

[6] A. P. Wright, A. T. Wright, A. B. McCoy, and D. F. Sittig, "The use of sequential pattern mining to predict next prescribed medications," Journal of Biomedical Informatics vol. 53, Elsevier. 2015. pp. 73-80.

[7] Y. Fan, Y. Ye, and L. Chen, "Malicious sequential pattern mining for automatic malware detection," Expert Systems with Applications vol. 52, Elsevier. 2016. pp. 16-25.

[8] T. Miura, and Y. Okada, "Detection of linkage patterns repeating across multiple sequential data," International Journal of Computer Applications vol. 63, no. 3, 2013. pp. 14-17.

[9] S. Lee, T. Miura, Y. Okubo, and Y. Okada, "Linkage pattern mining method for multiple sequential data with noise," IAENG International Journal of Computer Science vol. 42, no. 4, 2015. pp. 361-367.

[10] H. Mannila, H. Toivonen, and A. I. Verkamo, "Discovery of frequent episodes in event sequences," Data Mining and Knowledge Discovery vol. 1, Springer. 1997. pp. 259-289.

[11] K. Y. Huang, and C. H. Chang, "Efficient mining of frequent episodes from complex sequences," Information Systems vol. 33, no. 1, Elsevier. 2008. pp. 96-114.

[12] N. Miyoshi, T. Shigezumi, R. Uehara, and O. Watanabe, "Scale free interval graphs," Theoretical Computer Science vol. 410, no. 45, Elsevier. 2009. pp. 4588-4600.

[13] N. Korte, and R. H. Möhring, "An incremental linear-time algorithm for recognizing interval graphs," SIAM Journal on Computing vol. 18, no. 1, 1989. pp. 68-81.

[14] G. S. Lueker, and K. S. Booth, "A linear time algorithm for deciding interval graph isomorphism," Journal of the ACM (JACM) vol. 26, no. 2, 1979. pp. 183-195.

[15] T. Uno, M. Kiyomi, and H. Arimura, "Lcm ver. 3: Collaboration of array, bitmap and prefix tree for frequent itemset mining," Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations. 2005. pp. 77-86.