A Study on Stream Prediction Based on Timing Correlation among Multiple Data Stream

Kenji Arai†, Yoh Shiraishi‡, and Osamu Takahashi‡

†Graduate School of Systems Information Science, Future University Hakodate, Japan
‡School of Systems Information Science, Future University Hakodate, Japan
{g2110002†, siraisi‡, osamu‡}@fun.ac.jp

Abstract - Research on data streams has attracted a great deal of attention in many fields in recent years, such as sensor-network technologies and stock quote data. Therefore, stream prediction technologies have attracted the attention of stream mining technologies. When we want to obtain the predicted value of a certain single data stream, most methods use past data on the data stream. However, we think that correlations, such as synchronization, can be used for a method for predicting streams, and their accuracy might be better than methods that only use past data on single data streams. In addition, we need to take into consideration that correlations are not based on synchronization, which we call "similarity correlation". We suggest a method for detecting "timing correlation" from multiple data streams in this paper, and a method for predicting streams on the basis of these correlations. We also demonstrate and discuss the efficacy of these methods.

Keywords: Data Stream, Clustering, Classification, Correlation, Prediction

1 INTRODUCTION

Research on data streams has attracted a great deal of attention in numerous fields in recent years, such as sensor-network technologies and stock quote data. Data streams are expressed as time-series data of unlimited length and increase in real time. Stream mining technologies have been studied intensively [1] to find significant patterns from data streams. For example, some researchers have studied the diverse trends in data streams [2]. These trends express many features, such as periodicity. They are used to predict future data of data streams in accordance with past and present trends.

When predicting certain data stream, most methods only use past data on the predicted object. However, if two or more data streams are measured from sensors installed indoors, their correlation, such as synchronization of values, might appear from these data streams. We think their correlations can be used by methods for predicting streams and they might be more accurate than approaches that only use past data on single data stream. We assumed a situation where two data streams were measured simultaneously in Fig. 1 as an example of this theory. The wave forms of temperature data change in accordance with humidity data near the current time. The accuracy of prediction may decrease when strange values like these are measured if we only use a method for prediction using past data on single data streams. However, if the correlation in which temperature data increase after a rapid increment in humidity is found, accuracy of prediction will be improved by taking into consideration this correlation.

We assumed we needed to consider two correlations between data streams.

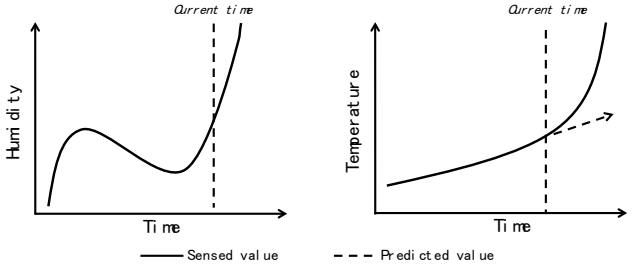

Figure 1: Sample of Stream Prediction Based on Correlation

I. Similarity Correlation

Streams A and B in Figure 2 are measured simultaneously. The two data streams are similar because of their measured values and their trends are very close. There is a correlation based on similarity between these two data streams which we call the "similarity correlation" in this paper.

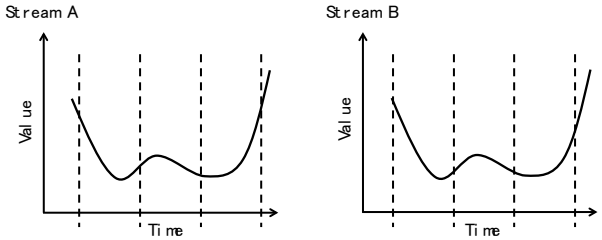

Figure 2: Example of Similarity Correlation

II. Timing Correlation

Streams C and D in Fig. 3 are measured simultaneously.

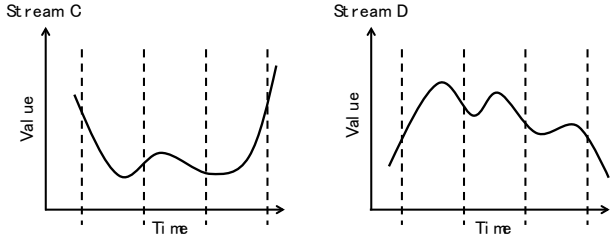

Figure 3: Example of Timing Correlation

The measured value and trend of stream C differ from those of stream D. However, we assumed there was a rule

based on the timing of changing trends. For example, when stream C increases, stream D decreases. We found this rule to be one of correlation and call it "timing correlation" in this paper.

Our task was to predict data streams on the basis of similarity and timing correlation, and we archived this by using our new approach.

2 RELATED WORK

2.1 Related Work in Stream Mining

Many researchers in stream mining technologies have studied trend detection technologies, such as detection trends from partial sequences made from data streams [3][4]. Trends represent features, such as the periodicity of partial sequences and data streams. Kawashima et al. found a method for reducing the number of calculations by cutting off the dimensions of partial sequences with adaptive piecewise constant approximation [5]. They succeeded in fast matching of partial sequences to sample sequences in the database with this method. Toyota et al. introduced a method for detecting trends based on the dynamic time warping (DTW) distance [6]. Papadimitriou et al. [7] predicted future data of data streams by using models like the auto regression model using the wavelet coefficients of data streams. The coefficients of wavelet transforms and model updates are executed each time a stream data is detected at minimum cost because previous results are used. However, this method does not consider the correlation between multiple data streams.

In addition, Sakurai et al. introduced a correlation detection technology called "BRAID" [8]. Zhu created StatStream [9] to detect correlations by comparing the DFT coefficients of partial sequences. However, these methods consider similarity correlation, timing correlation. Moreover, they did not refer to stream prediction.

2.2 Relationship with Data Mining

Data mining technologies are similar to those for stream mining. They find a significant rule from time-series data stored at databases. Stream mining technologies are regarded as one field in data mining technologies. However, stream mining technologies must consider the amount of memory and processing time, because data streams are time-series data that increase at certain intervals in real time. Therefore, some methods in stream mining technologies are different from these in data mining technologies. In stream mining technologies, the processing time and amount of memory are more important than complete results. In addition, approximated solutions are generically used instead of exact solutions in real time processes. Therefore, incremental algorithms are valued because they decrease the number of calculations. These algorithms use previous results to calculate new results.

3 PROPOSED METHOD

3.1 Overview of Proposed Method

We explain the requirements for our approach to solve our tasks in this section.

Data streams are predicted from estimation of future trends by taking into account past trends in data streams. Consequently, a method is required for detecting trends and correlations from past measured data. In addition, the method for detecting correlations needs to be based on numerous features of data streams to detect timing correlations, and it cannot use degrees of similarity such as the DFT coefficient and DTW distance. Therefore, these methods need to be incremental and only use a certain amount of memory.

First, we will explain the environment for data streams in our approach. Two or more data streams are measured simultaneously. All data in the data streams are measured accurately at certain intervals with no missing or delayed data.

Our method for detecting correlations and predicting data streams is clarified in what follows. Our method detects trends from data stream in real time. Next, it manages information on the appearance of trends. Correlation in data streams is detected by matching detected trends and the number of relationship on the basis of past trends. After that, our method predicts future data streams in accordance with correlations between data streams. It does the following:

1. Detects new trends.
2. Manages information on the appearance of trends.
3. Detects correlations between data streams, and
4. Predicts future data stream based on correlations.

3.2 Detection of Trends

A) System for Classifying Trends

Our method divides data streams into partial sequences in real time, and detects trends in partial sequences as current trends in the data streams by using a classification system. Therefore, a classification system must be prepared from past data. The method executes the following five steps to construct a classification system.

i. Divides all data streams into partial sequences by a certain interval.
ii. Extracts feature quantity patterns FQ-P and FQ-N from all partial sequences.
iii. Cluster partial sequences in accordance with FQ-P.
iv. Cluster partial sequences in associated with each cluster in accordance with FQ-N.
v. Construct a classifier from the clusters.

a) Divide Streams into partial sequences

Our method divides each data stream into multiple partial sequences with a certain interval. Each interval is called a window. We have assumed partial sequences divided by the same window have the same number of data. All data streams are divided into the same number of partial sequences.

b) Extract Feature Quantity Patterns

The method extracts feature quantities that express trends in all partial sequences and bundles a number of feature quantities as a pattern called a feature quantity pattern in this paper.

We used the power spectrum obtained from the DFT coefficients of partial sequences and natural values of partial sequences as resources for feature quantities. We called a feature quantity pattern obtained from the former an "FQ-P". We also called the latter an "FQ-N". The method extracts the following.

- Maximum Value / Data index of maximum value
- Minimum Value / Data index of minimum value
- Variance
- Value of integral

FQ-P expresses the periodicity of partial sequences while FQ-N expresses other waviness features.

c) Clustering Partial Sequences

Our method classifies partial sequences into various groups (clusters) in accordance with the extracted feature quantities (feature quantity pattern). A wide variety of feature quantity patterns can be generated from partial sequences. Therefore, detecting the number of clusters in advance is difficult. We must use clustering algorithm that can automatically detect the number of clusters. In addition, the clustering algorithm must consider similarity of classified partial sequences in a cluster, such as those in Fig. 4.

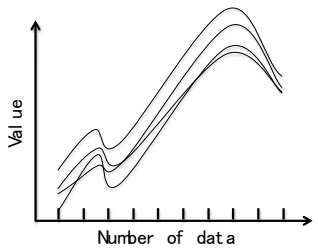

Figure 4: Example of Partial Sequences in Cluster

Therefore, we devised a clustering algorithm that uses a cluster division standard based on Eqs. (1) and (2). In these equations, variable k is the number of cluster, variable n is the number of partial sequences in cluster k, and variable m is the number of feature quantities in a feature quantity pattern. Variable x is the variable j th value in the i th partial sequences in cluster k. Variable V_{kj} is the variance of the j th values obtained from all partial sequences in cluster k. Variance E is the standard deviation of V_{kj}, and this is the cluster division standard. The variable E of a cluster expresses the variability of partial sequences.

$$V_{kj} = \sum_{i=1}^{n} x_{kij}^2 / n - \left(\sum_{i=1}^{n} x_{kij} / n \right)^2 \quad (1)$$

$$E_k = \sqrt{\sum_{j=1}^{m} V_{kj}^2 / m - \left(\sum_{j=1}^{m} V_{kj} / m \right)^2} \quad (2)$$

Furthermore, the method uses two feature quantity patterns to classify strictly partial sequences into clusters in accordance with similarity of multi-aspects. First, it uses FQ-P for classification. Second, it classifies partial sequences in each cluster into new clusters in accordance with FQ-N. The clustering algorithm using cluster division standard E involves six steps.

i. Classify partial sequences into two clusters by two-means clustering in accordance with FQ-P, and add these clusters to the cluster list.
ii. Classify each cluster into two clusters by two-means clustering. The divided clusters are the "parent cluster", and the new clusters are the "child clusters".
iii. Calculate each cluster's division standard.
iv. If the average of a child cluster's cluster division standard is lower than that of the parent cluster, add child clusters to the cluster list and remove the parent cluster.
v. Repeat steps 2 to 4 while the clusters are divided.
vi. Repeat steps 2 to 5 in accordance with FQ-N

d) Construction of classification system

The method constructs a classification system, to classify partial sequences in real time. A classification system is constructed for all data streams. We used C4.5, which is an algorithm for constructing decision trees implemented in Weka [10] as J48. All feature quantity patterns in each cluster are used for the construction. A decision tree classifies a partial sequence into a cluster in accordance with the thresholds of feature quantities, as shown in Fig. 5, which is created from the feature quantity patterns of stream A.

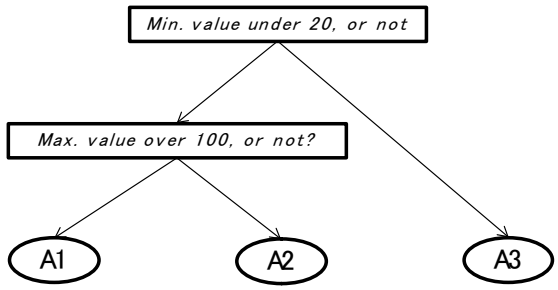

Figure 5: Example of Decision Tree for Feature Quantities

B) Detect Trends using Classification system

Our method detects trends from new partial sequences measured in real time. Each partial sequence obtains a cluster number. These numbers enable partial sequences to be expressed as cluster number streams, as shown in Fig. 6, where the partial sequences of data stream A are classified into {A1, A2, A3}.

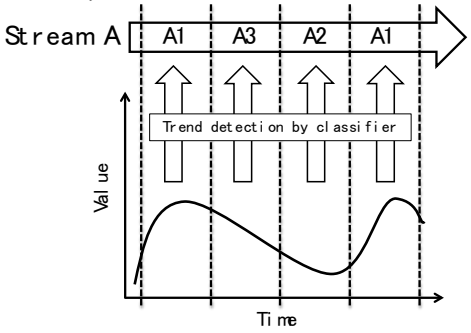

Figure 6: Example of Trend Detection in Stream A

Because the first and fourth partial sequences are similar, the system for stream A classifies both partial sequences into the same cluster, such as A1 in Fig. 6. Other partial sequences are classified into each cluster.

3.3 Managing Correlations

Our method detects the correlation between information on the appearance of trends. Our method manages two kinds of correlation rule between trends. The first is for the correlations between trends from different kinds of data streams. The second is for the correlations between trends in continuing windows from each data stream. We introduced a cluster correlation table to manage the former correlations and a cluster transition table to manage the latter correlations.

A) Cluster Correlation Table

The method detects trends at certain intervals, and a cluster simultaneously appears in all data streams. The cluster correlation table, in Fig. 7, manages the relationships between clusters from different data streams that appear. Stream A and B are measured simultaneously in this figure. Partial sequences from streams A, and B are classified into {A1, A2, A3} and {B1, B2, B3}. For example, when a partial sequence in stream A and partial sequences in stream B are simultaneously classified into A1 for the first and B3 for the second, the corresponding cells (A1, B3) are updated. Each cell means the frequency with which clusters from different data streams appear simultaneously.

		Stream A			Stream B		
		A1	A2	A3	B1	B2	B3
Stream A	A1				5	2	1
	A2				20	20	15
	A3				9	8	38
Stream B	B1	11	20	9			
	B2	29	20	8			
	B3	15	15	38			

Figure 7: Sample of Cluster Correlation Table

Transition rule		Count
From	To	
A1	A1	12
A1	A2	1
A1	A3	33
A2	A1	19
A2	A2	11
A2	A3	4
A3	A1	32
A3	A2	4
A3	A3	5

Figure 8: Example of Cluster Transition Table

B) Cluster Transition Table

We use a cluster transition table to estimate a cluster for a partial sequence in the next window of a stream. A cluster transition table manages the cluster transition rules for each stream. Figure 8 shows an example of a cluster transition table for stream A. When the cluster of the targeted window is A1 and the cluster of the next window is A2, the transition rule is A1→A2. The number of transitions in the corresponding record is updated every time a transition occurs. By referring to this cluster transition table, we can examine a cluster that is easy to change from the targeted cluster. For example, since there are many transitions of A1 →A3 and A3 →A1 in Fig. 8, these transitions are likely to occur.

3.4 Prediction Based on Correlation

A) Prediction Cluster based on Correlation

The method predicts a trend of the next partial sequence on the basis of correlations between clusters. It involves the four steps in Fig. 9.

i. First, it determines the latest window in accordance with the current time. The current time exceeds the end time for the latest window. The current window, including the current time, does not yet have enough data to detect trends with a classifier made for the target stream.
ii. Next, it searches a cluster correlation table for a correlated cluster that is likely to occur simultaneously with the latest cluster from the targeted stream.
iii. It searches the cluster transition table of the stream including the correlated cluster for a cluster (estimated cluster) that will occur in the next window.
iv. Finally, it again searches the correlation table for the cluster of the targeted stream that is likely to occur simultaneously with the estimated cluster. The cluster that is searched is the result of this prediction process.

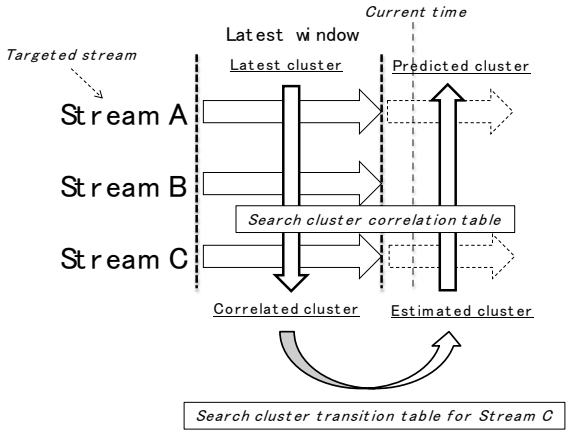

Figure 9: Stream Prediction by Correlation of Clusters

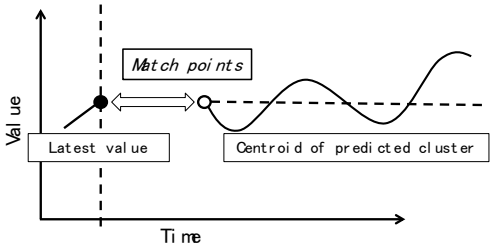

Figure 10: Restoration of sequences by Cluster Centroid

Table 1: Number of Clusters and Evaluation

Sensor	Scale	20 h		10 h		4 h		1 h	
		Num. of Clusters	Evaluation Value	Num. of Clusters	Evaluation Value	Num. of Clusters	Evaluation Value	Num. of Clusters	Evaluation Value
Temperature	0~40C	16	5.212	29	3.494	73	2.680	74	1.570
Humidity	0~100%	20	7.315	28	6.067	85	4.282	264	2.120
Illuminance	0~65000lx	20	1.819	43	1.945	**19**	**11.605**	**5**	**13.173**

Table 2: Average Number of Sequences in Each Cluster

Sensor	20 h	10 h	4 h	1 h
Temperature	3.062	3.370	3.342	12.568
Humidity	2.450	3.500	2.870	3.528
Illuminance	2.450	2.270	**12.842**	**196.000**

B) Partial Restoration of Sequences from Cluster

The method restores the next partial sequences from the predicted cluster in accordance with the latest data and the cluster centroid of the predicted cluster. A cluster centroid is a specific partial sequence created from the average of all partial sequences in a cluster. This means the effective features of partial sequences in the cluster. Our method attaches the latest data to the cluster centroid, and regards the virtual partial sequence starting at the latest data as the next partial sequence, as shown in Fig. 10. This enables the user and application to obtain the required data from the next partial sequence.

3.5 Approach to Multiple Interval Prediction

Feature quantity patterns express general features of partial sequences, but cannot determine the details. Thus, a cluster centroid cannot express the details of partial sequences associated with a cluster.

In addition, the predicted partial sequence length depends on the length of the cluster centroid used by prediction. Therefore, this depends on the length assumed by the classification system used in detection trends, because it is necessary to extract feature quantity patterns from partial sequences divided with the same interval as the interval of partial sequences to construct an accurate classification system.

According to this, the details on predicted partial sequences may differ from actual measured data. More specifically, this difference increases when the method predicts short partial sequences by using cluster correlation tables and cluster transition tables based on the classification system for long partial sequences. For example, the details on a centroid, which short partial sequences created, are lacking in long sequences, as shown in Fig. 11.

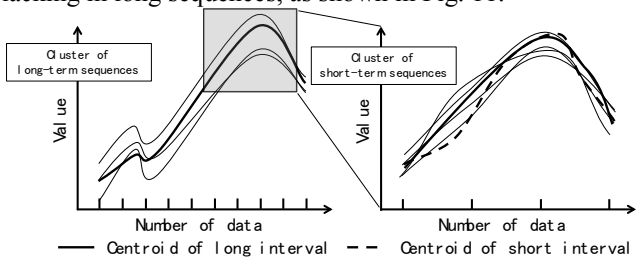

Figure 11: Data Lacking by Long-term Partial Sequences

Therefore, our method creates classification systems and tables with various partial sequence lengths. It selects the classifier and table in accordance with the time when a user and application want to obtain data. For example, if data for the near future are required, the method uses the tables based on short intervals. This enables the user and application to obtain accurate predicted data.

3.6 Using Past Measurements

Our method detects correlations in data streams from cluster correlation table and cluster transition table. Because these tables require a certain amount of past data to detect of correlation accurately, our method never detects correlations until these tables store enough past data. Therefore, we use the clustering results that are used for building a classifier to initialize two tables. We think this attempt improves the accuracy of prediction when real time trend detection is launched.

If unique measurements cannot be detected accurately by using classifier appearance, the accuracy of prediction decreases. This case may often happen in data streams treating seasonal data, such as measurements of temperature sensors in natural environments. Therefore, we suppose the classifier and two tables will be updated with a certain time.

4 EXPERIMENTS AND DISCUSSION

4.1 Evaluation of Clustering Algorithm

We carried out basic experiments to verify the efficacy of our method. We prepared sample data for the experiments and built classification system with various lengths.

This experiment used the stored data measured by a sensor unit used for farming [11] located in Fukushima Prefecture. This sensor unit measured the temperature, humidity, and illuminance from November 25, 2010 to February15, 2011. We only used 49 days worth of data from this span, when the sensor unit measured for 20 h between 0:00 to 20:00. The lengths of partial sequences were {20, 10, 4, 1}.

We defined the cluster evaluation value for prediction *CE* as the expectation value for prediction error obtained with Eqs. (3) and (4). Variable k is the number of clusters in these equations, and variable m is the number of partial sequences. In addition, variable c_{kj} means the *j* th data of the cluster centroid in cluster *k*, and x means one of the data.

Table 3: Accuracy of Each Classifier

Sensor	20 h			10 h			4 h			1 h		
	J48	MP	BN	J48	MP	BN	J48	MP	BN	J48	MP	BN
Temperature	95.9	92.3	97.8	95.9	97.6	90.2	89.8	84.8	71.9	80.6	60.2	57.1
Humidity	100	100	90	97.9	97.6	92.0	92.9	90.2	85.5	90.0	49.7	65.6
Illuminance	97.5	93.9	90.2	100	92.3	77.0	87.2	84.2	77.4	55.1	76.1	76.0

Variable x_{kij} means the j th data of the i th partial sequence at cluster k.

$$Dki = \left\{ \sum_{j=1}^{m} \frac{100}{(max - min)} \times \sqrt{(c_{kj} - x_{kij})^2} \right\} \Big/ m \quad (3)$$

$$CE_k = \left\{ \sum_{i=1}^{n} D_{ki} \right\} /n \quad (4)$$

Variable D_{ki} means the average of all deltas with all data of the cluster centroid. This variable is the delta of the i th partial sequence in cluster k. CE_k is the average of all deltas of partial sequences in cluster k. Variable *max* and *min* are the maximum and minimum values according to sensor performance. A delta is expressed as a percentage.

Tables 1 and 2 summarize the results obtained from the experiment. Table 1 lists the numbers of clusters and cluster evaluation values for prediction. Table 2 lists the expected number of partial sequences in each cluster. *CE* reached below 10 percent in most lengths of partial sequences. This means the error between actual data and predicted data will reach below at least 10 percent. For example, the error will reach bellow 4C in the temperature sensor. We assumed this error would be permissible for prediction in most cases. The expected numbers of partial sequences were near three in most lengths in Table 2. This is a particularly accurate result for our clustering algorithm that classifies the source into two clusters as recursively as possible. The numbers of clusters increase as intervals shorten. This means unique types of partial sequences increase in inverse relation to decreasing interval lengths. Therefore, we attributed these results to our clustering algorithm being able to detect unique types.

In both tables, the illuminance sensor's results in cases where the interval length is 4 or 1 h differ from the other results. These accented results mean trend-detection failed in these cases. We assumed our method failed because the feature quantities used in it were not compatible with the illuminance data in these interval lengths. Therefore, the method must correctly select feature quantities in accordance with the types of sensors and interval lengths.

4.2 Evaluation of Classifier Algorithm

We carried out a basic experiment on a classifier to determine an adequate algorithm for constructing it. Three classifiers were constructed in the experiment by using C4.5 and Multilayer Perceptron and Bayesian Networks, and we conducted 10-fold cross-validation to obtain the accuracy of the classifier. All algorithms were implemented in Weka. Multilayer Perceptron is an algorithm for building neural networks. Accuracy was expressed by the percentage of correct classifications. The experiment used the clustering results at each interval of partial sequences.

Table 3 summarizes the accuracy of each classifier where J48 is much more accurate than the others. Therefore, J48 was considered to be the best algorithm. In addition, the best accuracy was over 90% at 20 and 10 h. However, at 4 and 1 h, accuracy worsened. Most accuracy at 1 h especially fell below 80%. We assume the algorithms failed to construct classifiers in these cases. The algorithms failed because of our clustering algorithm, which could not consider the threshold used by the classifier, because our clustering algorithm executed classification in accordance with the Euclidean distance of the feature quantity pattern made from each partial sequence. We assumed the clustering algorithm would be ruled unsuitable as an algorithm for constructing classifiers by using a threshold. The clustering algorithm needs to be used by considering threshold of feature quantity to succeed in constructing classifiers.

5 CONCLUSION

This paper proposed a method for detecting the "timing correlation" between multiple data streams by using information on the appearance of trends. The basic experiments demonstrated the efficacy of the method for prediction using clusters created from the detection of trends. However, we found problems with the clustering method and feature quantities in clustering. It is necessary to do further research on adequate feature quantities for each sensor and intervals to find a new clustering method. In addition, we will verify the efficacy of our method by additional experiments. We will estimate clusters of past measured data by using the proposed method. The accuracy will be given by matching estimated clusters to actual clusters detected by classifier. If this experiment shows good accuracy, we will implement the system predicts data stream in real time, which using our method, in order to verify the efficacy against a single data stream prediction, where multiple data streams exist.

REFERENCES

[1] Y. Sakurai, ``Stream mining technologies for time-series data,'' IPSJ Magazine, Vol.47, No.7, pp.755-761 (2006). (*in Japanese*)

[2] H. Arimura and T. Kida, ``Mining technologies for data stream,'' IPSJ Magazine, Vol.46, No.1, pp.4-11 (2005). (*in Japanese*)

[3] Y. Fujiwara, Y. Sakurai and M. Yamamuro, ``A search method for multiple data streams,'' Database Society of Japan Letters, Vol.4, No. 4, pp.13-16 (2006).

[4] K. Sugawara, Y. Shiraishi and O. Takahashi, ``Estimation of the user action and static object using

3-Axis Acceleration sensor and vibration motor based on PWM control for cell-phone,'' Collection of papers of DPS Workshop 2009, pp.25-30 (2009). (*in Japanese*)

[5] H, Kawashima, M. Toyama and M. Imai, Y. Anzai, ``Retrieval of similar sequences with waveform features,'' technical report of IECIE SIG-DE, Vol.102, No.209, pp.125-128 (2002). (*in Japanese*)

[6] M. Toyoda, Y. Sakurai and T. Ichikawa, ``Stream matching based on Dynamic Programming,'' IPSJ SIG Technical Report DBS, Vol.2008, No.88, pp.277-282 (2008). (*in Japanese*)

[7] S. Papadimitriou, A. Brockwell and C. Faloutsos, ``Adaptive, hands-off stream mining,'' Proceedings of the 29th International Conference on VLDB, pp.560-571 (2003).

[8] Y. Sakurai, S. Papadimitriou and C. Faloutsos, ``BRAID: stream mining through group lag correlations,'' Proceedings of the 2005 ACM SIGMOD, pp.599-610 (2005).

[9] Y. Zhu and D. Shasha, ``StatStream: statistical monitoring of thousands of data streams in real time,'' Proceedings of the 28th International Conference on VLDB, pp.358-369 (2002).

[10] S. Onishi, Y. Sakai, N. Yamaguchi and T. Shiraishi, ``e-kakashi project,'' HTML, Available at "http://project.e-kakashi.com/" (2010). (*in Japanese*)

[11] Machine Learning Group a University of Waikato, ``Weka 3 - Data Mining with Open Source machine Learning Software in Java,'' HTML, Available at http://www.cs.waikato.ac.nz/ml/weka/

(Received February 27, 2012)
(Revised November 23, 2012)

Kenji Arai received B.E. and M.S. degrees from Future University Hakodate in 2010 and 2012. His research interests include sensor database system and data mining. He currently works at KDDI corporation in Japan.

Yoh Shiraishi received doctor's degree from Keio University in 2004. He is currently an associate professor at the Department of Media Architecture, School of Systems Information Science, Future University Hakodate Japan. His research interests include database, mobile sensing and ubiquitous computing. He is a member of IPSJ, IEICE, GISA and ACM.

Osamu Takahashi received master's degree from Hokkaido University in 1975. He is currently a professor at the Department of System Information Science at Future University Hakodate, Japan. His research interest includes ad-hoc net-work, network security, and mobile computing. He is a member of IEEE, IEICE, and IPSJ.