Classification System for Time Series Data Based on Feature Pattern Extraction

Hiroshi Sugimura
Graduate School of Engineering
Kanagawa Institute of Technology
Kanagawa, Japan
hiroshi.sugimura@gmail.com

Kazunori Matsumoto
Graduate School of Engineering
Kanagawa Institute of Technology
Kanagawa, Japan
matumoto@ic.kanagawa-it.ac.jp

***Abstract*—This paper proposes a system which acquires feature patterns and makes classifiers for time series data without using background knowledge given by a user. Time series data are widely appeared in finance, medical research, industrial sensors, etc. The system acquires the feature patterns that characterize similar data in database. We focus on two aspects of the feature pattern: global and local frequency. Our purpose is to acquire features of each data by extracting these patterns. The system cut out subsequences from time series data. Several representative sequences are extracted from these subsequences by using clustering. Feature patterns are acquired from these representative sequences. For this purpose, we develop a method that applies TF*IDF weight technique, which is often used in text mining, to time series data. The time series data are classified by using the acquired feature patterns. In accordance with a criterion that is based on the entropy theory, feature patterns are improved by the automatic process, generation by generation, using the genetic algorithm. By using the final and optimized feature patterns, we build a decision tree that determines future behaviors. We explain how these two tools are combinatory applied in the entire knowledge discovery process.**

***Keywords*—datamining, time series, classification, TF*IDF, dynamic time warping, decision tree, genetic algorithm**

I. INTRODUCTION

In this paper, we deal with time series data that are stored into databases as sequences of numerical data. This type of data occurs widely in business applications and in science. Well-known examples include audio data of voice, daily stock prices on the Tokyo Stock Market, product sales histories, engine testing histories, and daily sea-surface temperature readings. Finding a technique that effectively extracts knowledge from such data is important.

In [1], they propose a method that efficiently discovers frequent patterns from time series data. These methods regard the frequent patterns as feature patterns. However, the frequent patterns do not always correspond to the interests of analysts, because the patterns are common and not necessarily a source of new knowledge for the analysts. The patterns coinciding with the features may be buried in a large number of discovered patterns.

In [2], a method that uses a user-specified subsequence of time series data as background knowledge is proposed. The method finds similar patterns from time series data by using visual query language. The visual query language deals with user-specified two patterns and its combination. Using this method, analysts can discover feature patterns that accord with their interests. However, the method requires background knowledge depending on analysis tasks. If the background knowledge of analysts is insufficient, the method cannot discover interest patterns.

A classification method for time series data by using the support vector machine (SVM) is proposed in [3]. SVM strategy is a powerful method that has outperformed most other systems in a wide variety of applications. However, the classifier made by SVM has a problem that it is difficult to understand.

A method to predict future behavior by using clustering is proposed [4]. This method makes a classifier based on training data which are made by all cluster obtained by clustering. It is difficult that a user confirms all extracted clusters. Therefore a user is not able to discover important patterns.

This paper proposes a method that acquires feature patterns for time series data without using background knowledge given by a user. Our method starts with feature acquisition process. In accordance with statistical measurements to represent importance, feature patterns are acquired from the database. The acquired features become attributes in the machine learning. For the understandability, we use the decision tree learning. The decision tree learning can generate some rules to describe the decision model.

In addition, we describe a method that improves feature patterns by using the genetic algorithm (GA). GA is an adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetics [5]. This heuristic is generally used to generate useful solutions for optimization and search problems.

In [6], they propose a method to encode and decode a decision tree to and from a chromosome where genetic operators such as mutation and crossover can be applied are presented. Our approach indirectly increments the quality of a decision tree by improving feature patterns. This paper describes the method and applies it to real data.

II. Acquisition of Feature Pattern

We aim to acquire feature patterns from time series data. A feature pattern has the power to characterize similar time series data in database. A frequent pattern in a time series data is important. On the other hand, a pattern that exists widely in the entire database is not important for any specific datum. Fig. 1 shows an example of feature patterns for time series data.

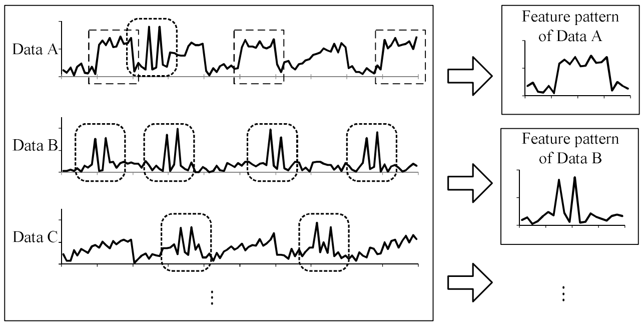


Figure 1. Example of feature patterns for time series data

A. TF*IDF

In text mining, the TF*IDF weight is recognized as a statistical measurement of importance of terms [8]. A document includes a set of terms, and a database consists of documents. The importance increases proportionally to the number of times a term appears in the document, but is offset by the frequency of the term in the database. Formally, we denote the number of occurrences of term w_i in document t_k by $\mathrm{tf}(w_i,t_k)$ and call it the term frequency factor. Term w_i that occurs in some specific documents is more important than term w_j that occurs in almost all documents. This is called the inverse document frequency, and is denoted by $\mathrm{idf}(w_i) = \log \frac{N}{n}$, where N is the number of documents, and n is the number of documents where word w_i occurs at least once. TF*IDF weight is defined as:

$$\mathrm{TF*IDF}(w_i,t_k) = \mathrm{tf}(w_i,t_k) \times \mathrm{idf}(w_i) \tag{1}$$

We uniquely extend this TF*IDF technique to time series data. We regard a sequence of time series data as a document and subsequences of it becomes terms. The total number of subsequences becomes very huge, therefore we find representatives of subsequences. The clustering techniques for this purpose that is explained more in detail in the next section.

B. Clustering

In order to obtain representative subsequences, we apply the *k*-means clustering to extracted subsequences, which is one of the most popular clustering algorithms [9]. Suppose we are given a data set. The *M*-clustering problem aims at partitioning this data set into *M* disjoint subsets (clusters), such that a clustering criterion is optimized. The clustering criterion is the distance between each data x_i and the cluster center c_j (centroid) of the subset C_j which contains $x_i^{(j)}$. This criterion is called clustering error and depends on the centroid $c_1, \ldots, c_k$, where $D(p,q)$ is a distance function that measures distance between data p and data q.

$$Err(X) = \sum_{j=1}^{k} \sum_{i=1}^{N} D\left(x_i^{(j)}, c_j\right) \tag{2}$$

The algorithm steps are:

1. Choose the number of clusters k,
2. Randomly generate k clusters and determine the centroids,
3. Assign each point to the nearest centroid, where nearest is defined with respect to one of the distance measurements $D(p,q)$,
4. Recompute the new centroids, and
5. Repeat the two previous steps until some convergence criterion is met (usually that the assignment has not changed).

The most widely used distance function $D(p,q)$ is the Euclidean distance or its variations. However, Euclidean distance can be an extremely brittle distance measurement. Euclidean distance may fail to produce an intuitively correct measurement of distance between two sequences because it is very sensitive to small distortions in the time axis.

C. Dynamic time warping

Dynamic Time Warping [10] is an algorithm for measuring distance between two time series data. A time series is a list of samples taken from a signal and ordered by the time that the respective samples are obtained. A naive approach to calculating a matching distance between two time series could be to resample one of them and then compare the series sample-by-sample. The drawback of this method is that it does not produce intuitive results, as it compares samples that might not correspond well (see Fig. 2 (a)).

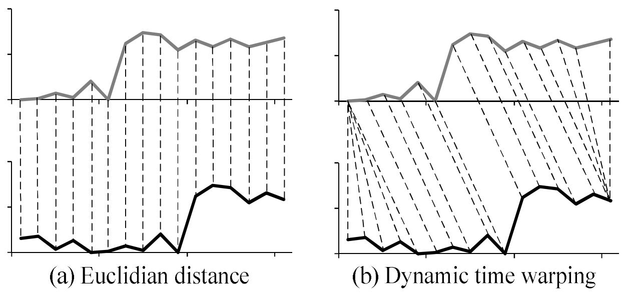


Figure 2. Euclidean distance and DTW distance

DTW solves this discrepancy between intuition and calculated matching distance by recovering optimal alignments between sample points in the two time series. The alignment is optimal in the sense that it minimizes a cumulative distance measure consisting of local distances between aligned samples. Fig. 2 (b) shows such an alignment. The procedure is called "Time Warping" because it warps the time axes of the two time series in such a way that corresponding samples appear at the same location on a common time axis.

For two sequences that have different lengths *i*, and *j*, the dissimilarity degree $D(x_i,y_j)$ is defined in the following equation, where x_{i-1} and y_{j-1} is the sequences with the last values removed.

$$D(x_i, y_j) = D(x_{i-1}, y_{j-1}) + \min \begin{cases} D(x_i, y_{j-1}) + q \\ D(x_{i-1}, y_j) + r \\ D(x_{i-1}, y_{i-1}) + s \end{cases} \qquad (3)$$

In equation (3), *q* and *r* represent the cost of shortening and expanding the sequences along the time axis, and *s* is the distance cost of values. A more similar pair has a smaller value, which becomes zero when they are exactly the same. For a given threshold value, we regard them as equal.

III. CLASSIFICATION

The classification is carried out as following steps:

1. Making of a classifier based on feature patterns,
2. Evaluation of the classifier, and
3. Improvement of feature patterns

First, the knowledge is discovered by the machine learning on the basis of acquired feature patterns. Second, discovered knowledge is evaluated. In accordance with this evaluation, the system chooses “stop” or “improvement”. If “stop” is chosen, the system outputs the result that is a set of discovered knowledge and used feature patterns. On the other hand, when “improvement” is chosen, the system carries out the third step. Third, the feature patterns are automatically improved on the basis of the evaluation, and return to the first step. This process is repeated until a termination condition has been reached in the second step.

A. Decision Tree Learning

A time series data *S* is a finite sequence of real values $S = (s_1, s_2, \ldots, s_n)$. A sequence of a feature pattern $P = (I_1, \ldots, I_m)$, where *n, m* is the length of its sequence. Each time series data *S* in classified into predefined categories, which is it is associated with a class label.

Fig. 3 shows a simple example of the method that makes instances in the training data from time series data. Time series data are labeled by using their future behavior. The representative sequences of future behaviors are obtained by clustering. Feature patterns are given to a data mining processor and used as “focus points” in the mining algorithm. The distance between a feature pattern and subsequences are computed by using the DTW mentioned above. The system applies this operation to all acquired feature patterns.

Fig. 4 illustrates an example of training data. Feature patterns from P_1 to P_3 are the given set of future patterns in this example. We compute how well all training data and each feature match. Each instance in the training data is associated with a class label C_1 to C_3, which denotes future behavior. A training data is made by tuples of these distances and classes

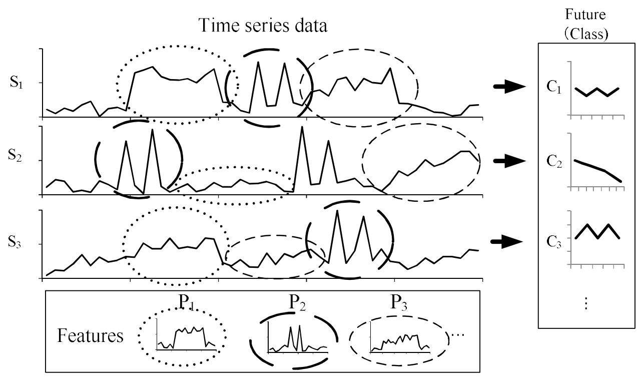


Figure 3. Matching and prediction

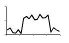
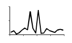
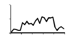
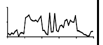
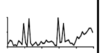
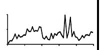
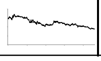
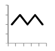
Data	P_1 	P_2 	P_3 	class
S_1 	58	43	85	C_1
S_2 	181	57	103	C_2
S_3 	133	88	81	C_2
S_4 	65	95	99	C_3 

Figure 4. Example of training data

The classifier made by using decision tree learning based on the training data in Fig. 4. We obtain knowledge as shown in Fig. 5.

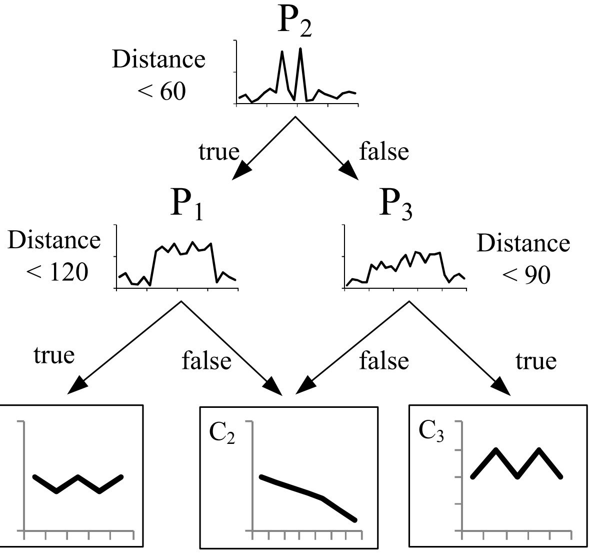


Figure 5. Example of rules generated by decision tree learning

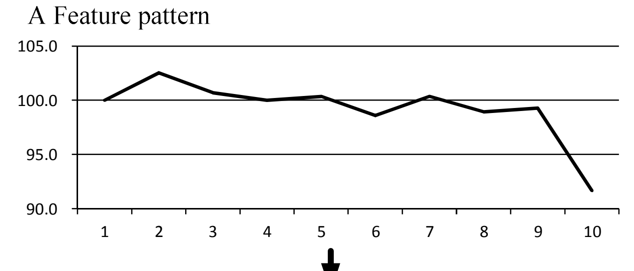
IV. IMPROVEMENT OF FEATURE PATTERNS

We also describe the mechanism to increment the quality of knowledge. For this purpose, the local search such as hill climbing and the reinforcement learning such as Q-learning can be applied. However, it may obtain a local optimal solution. Our approach is to use the GA.

As we described above, the essential idea of our approach is acquisition of feature patterns. And the quality of classification rules depends on the properties of the feature patterns. We naturally expect that better feature patterns discover better knowledge. Therefore, the quality of knowledge depends on the quality of feature patterns. The system improves the decision tree by improving feature patterns by using GA.

A. *Gene representation*

The representation of a feature pattern is an array of numerical values. Each pattern becomes each gene by normalization. A set of current genes becomes current generation of a family. Fig. 6 shows representation of a gene that corresponds to a pattern.



A Gene

index	1	2	3	4	5	6	7	8	9	10
value	100.0	102.6	100.7	100.0	100.4	98.6	100.4	98.9	99.3	91.7

Figure 6. A feature pattern and a gene

B. *Fitness function*

The system utilizes the information gain ratio criterion to evaluate genes. The information gain ratio criterion is used to determine the most appropriate gene for classifying all instances of training data. As we shown in Fig. 4, all instances are classified by future behavior, and distances are computed between their sequences and all genes. The system computes information gain for each gene by using training data.

In order to determine the information gain ratio, we have to look at the information conveyed by classified cases. We consider a set T of k training cases. If we select a single case $t \in T$ and decide that it belongs to class C_j, then the probability of this message is $\frac{\text{freq}(C_j,T)}{|\text{T}|}$ and it conveys $-\log_2\left(\frac{\text{freq}(C_j,T)}{|T|}\right)$ bits of information. Then the average amount of information needed to identify the class of a case in set T can be computed as a weighted sum of per-case information amounts:

$$\text{info}(T) = -\sum_{j=1}^{k} \frac{\text{freq}(C_j,T)}{|T|} \times \log_2 \frac{\text{freq}(C_j,T)}{|T|} \quad (4)$$

If the set T is partitioned into n subsets on the basis of outcomes of test X, we can compute a similar information requirement:

$$\text{info}_x(X) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times \text{info}(T_i) \quad (5)$$

Then, the information gained by partitioning T in accordance with the test X can be computed as:

$$\text{gain}(X) = \text{info}(T) - \text{info}_x(T) \quad (6)$$

The gain criterion is biased toward the high frequency data. To ameliorate this problem, we normalize the information gain by the amount of the potential information generated by dividing T into n subsets:

$$\text{split info}(X) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \times \log_2 \frac{|T_i|}{|T|} \quad (7)$$

Thus the gain ratio is defined as:

$$\text{gain ratio}(X) = \frac{\text{gain}(X)}{\text{split info}(X)} \quad (8)$$

C. *Selection*

Selection step rates the fitness of each gene and preferentially selects the best gene. We use roulette wheel selection [5] to do this. Typically, a proportion of the wheel is allocated to each of the possible selections on the basis of their fitness value. With fitness proportionate selection, some weaker genes may survive the selection process; this is an advantage, as although a gene may be weak, it may contain some components that could prove useful following the reproduction process. The roulette wheel selection algorithm consists of the following steps.

1. Sum the fitness of all population members; named total fitness n,
2. Generate a random number between zero and n, and
3. Return the first population member whose fitness added to the fitness of the preceding population members is greater than or equal to n.

D. *Reproduction*

The reproduction step is to create a next generation population of genes from those selected through genetic operators that are crossovers and mutated. To create each new gene, a pair of parent genes is selected for breeding from the pool selected previously. By creating a child gene using these methods, a new gene is created that has many of its parents' characteristics. The process continues until a new population of genes of suitable size has been generated. The mutation step adds to the current value of a gene with a randomly generated valid value that is in the range *-50* to *+50*.

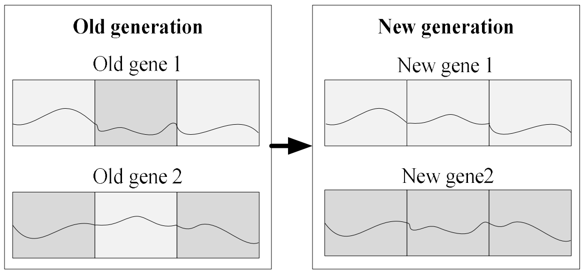


Figure 7. Two-points crossover

V. EXPERIMENT

In order to confirm the effectiveness of our proposed system, the experiments use real data. The real data are financial stock price data. Time series data are extracted by using the slide window method. The slide window length is 20 data. We use the C4.5 algorithm [11] within the WEKA [12], which is developed at the University of Waikato in New Zealand. In the DTW, the cost of horizontal (time axis) is *50.0*, and the cost of vertical is difference of values $|p_i - q_i|$.

In the experiment for the improvement of feature patterns, genetic algorithm increases the population of feature patterns one hundred times. The system shows the feature patterns and the decision tree that has the highest accuracy. When accuracy is the same, we prefer smaller trees. To evaluate the prediction accuracy, we use the 10-fold cross validation.

We compare the accuracy and the stability of three kinds of methods: direct approaches, a trading rule mining method proposed by [4], and our approach. This experiment uses stock price data from the Tokyo Stock Exchange in Japan. The time series stock data considered are row data from the Trade Science Corporation obtained from the Kaburobo website [13]. The data consist of the tuple of the opening price, the closing price, the highest price, and the lowest price of the trading day. The direct approaches and our method generally take the closing price as the feature set for the machine learning, and the prediction is created using the closing price only. The direct approaches carry out the decision tree learning, the support vector machine, and the neural network. The trading rule mining method is explained in section one. In accordance with the work, using Kaburobo SDK, we obtained four price values, trading volume, and 13 trend index values (that are Moving Average, Bollinger Band, Envelope, HLband MACD, DMI, volume ratio, RSI, Momentum, Ichimoku1, Ichimokuk2, Ichimoku3, and Ichimoku4). We obtain time series data consisting of the above mentioned attributes from about twenty five companies: Hoya, Japan Tobacco inc, Mizuho Financial Group, AEON, Orix, Canon, Kirin Brewery, Credit Saison, Komatsu, Sega Sammy Holdings, Secom, Softbank Mobile, Toyota Motor, Bridgestone, Honda Motor, Millea Holdings, Yamada Denki, Sumitomo Mitsui Financial Group, Mitsui Sumitomo Insurance, Asahi Glass, Kansai Electric Power, JFE Holdings, KDDI, Nippon Telegraph and Telephone (NTT), and NTT DoCoMo. The period from which we collect time series stock data, is from January 5th 2006 to December 29th 2006. About 2300 values are selected and used for experiment. Observation period of the future data is the next 5 data. In accordance with behavior of the period, stock price data is classified. Five classes are obtained by clustering as future behavior.

Our approach uses the feature patterns by the feature acquisition as an initial gene. The experiments are carried out using three sets of initial genes, which contain five, ten, and twenty initial genes. These genes are selected in descending order of TF*IDF weight.

Fig. 8 illustrates an outline of flow of our system. A stock price data of one company is expressed as a sequence of time series data. For each sequence, we first collect the set of all subsequences by using sliding window. We carry out a clustering on the whole subsequences of the entire series. Next, some representations are selected according to the TF*IDF measure, and then survived ones become feature patterns. Each stock price data is re-expressed with a use of the feature patterns. The feature patterns become attributes in the decision tree learning, and are improved by using the GA. Finally, we obtain classification rules.

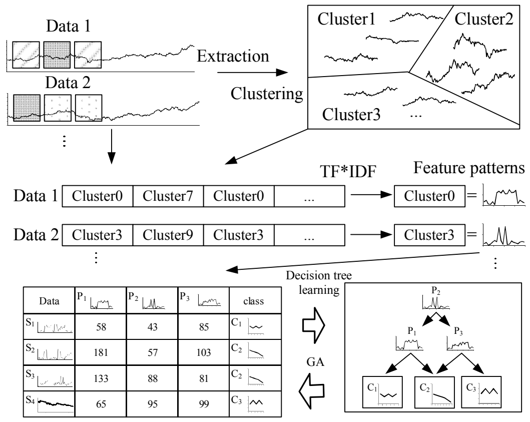


Figure 8. An outline of the system

Fig. 9 shows number of cluster *k* and total number of acquired feature patterns. The feature patterns are the clusters which have the maximum TF*IDF weight of each data. Same cluster is extracted from similar time series data.

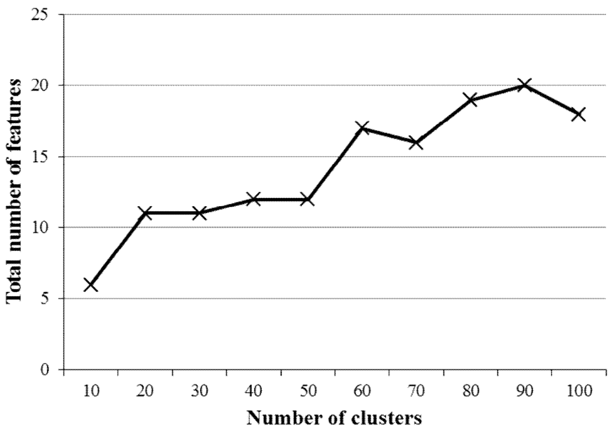


Figure 9. The total number of acquired features

Table I shows experimental results by using direct approaches, trading rule mining method, and our approach with stock price data. The accuracy is the average of twenty five companies. Fig. 10 shows the generations and the accuracy of the experimental results of the genetic algorithm.

TABLE I. ACCURACY OF EACH METHOD

Method		Accuracy
Direct approach	C4.5	30.89 %
	SVM	27.91 %
	NN	30.04 %
Traiding rule mining		53.11 %
Our approach	5 genes	34.60 %
	GA: 5 genes	65.23 %
	10 genes	46.13 %
	GA: 10 genes	85.28 %
	20 genes	69.96 %
	GA: 20 genes	89.87 %

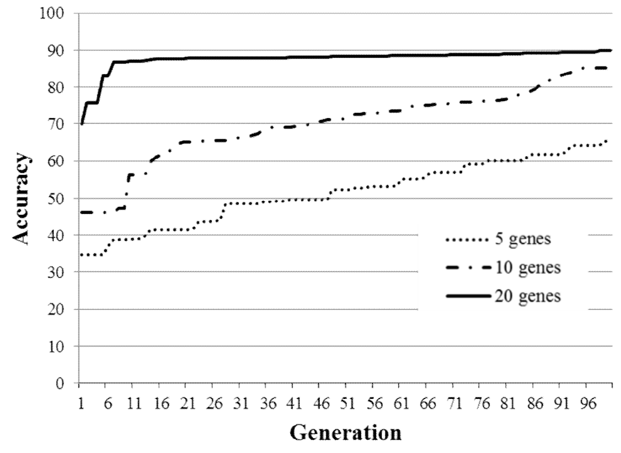


Figure 10. Improving accuracy by genetic algorithm

VI. DISCUSSION

In Fig. 9, in accordance with cluster *k*, the total number of acquired feature patterns is increased to 20. The classifier that is made by them has high accuracy. In Table I, the accuracy of 20 genes is greater than the trading rule mining method. In addition, the accuracy of classifier is improved by using GA. In Fig. 10 shows the accuracy that is improved generation by generation. In accordance with total number of genes, accuracy is improved quickly.

In order to be fair, our approach takes longer for processing than other approaches. The reason is the decision trees are made in each generations of GA. In this experiment, the termination condition of GA is when the 100th generation is created. However, it is thought that the processing time can be reduced by improving the termination condition. For example, we set the termination condition in which an accuracy threshold is a high value, such as 80%. Also, it seems that we should apply our method to predict longer periods than real-time trading.

VII. CONCLUSION

In this paper, we propose a system that acquires feature patterns and makes a classifier for time series data without using background knowledge. The system starts with acquisition of feature patterns from time series data. For this purpose, we develop the method that applies TF*IDF weight technique, which is often used in text mining, to time series data. The feature patterns become attributes in the machine learning, and are improved by using the genetic algorithm. The improved feature patterns are useful for improving the classification accuracy. The experimental results demonstrate the effectiveness of the system. In future work, we will try to apply this method to other data such as music, medical data, spam mail, and etc.

REFERENCES

[1] R. Agrawal and R. Srikant, "Mining sequential patterns," *in Data Engineering, 1995. Proceedings of the Eleventh International Conference on,* pp. 3-14, 2002.

[2] K.Z. Haigh, W. Foslien and V. Guralnik, "Visual Query Language: Finding patterns in and relationships among time series data", *Proceedings of the seventh Workshop on Mining Scientific and Engineering Datasets,* 2004.

[3] A. Kampouraki. G. Manis and C. Nikou, "Heartbeat time series classification with support vector machines", *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 4, pp. 512-518, 2009.

[4] H. Abe, S. Hirabayashi, M. Ohsaki and T. Yamaguchi, "Evaluating a trading rule mining method based on temporal pattern extraction," *The Third International Workshop on Mining Complex Data (MCD2007) In Conjunction with ECML/PKDD 2007*, pp. 49-58, 2007.

[5] J. H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[6] S. Hyuk Cha and C. Tappert, "A genetic algorithm for constructing compact binary decision trees," *Journal of Pattern Recognition Research (JPRR)*, vol. 4, no. 1, pp. 1-13, 2009.

[7] M. A. Bramer, *Principles of Data Mining. Springer*, New York, USA. 2007.

[8] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 60, No.5, pp. 493-502, 2004.

[9] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman and A. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 881-892, 2002.

[10] D. J. Berndt and J. Clifford, "Using Dynamic Time Warping to Find Patterns in Time Series," *Proceedings of KDD-94: AAAI Workshop on Knowledge Discovery in Databases*, Seattle, Washington, pp. 359-370, 1994.

[11] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

[12] The University of Waikato, in English, [Online]. Available: http://www.cs.waikato.ac.nz/.

[13] KabuRobo, in Japanese. [Online]. Available: http://www.kaburobo.jp.