Recognition and Generation of time-series Patterns Based on Division and Integration

Shogo Okada
Department of Computational Intelligence and Systems Science
Tokyo Institute of Technology
R2-52, 4259, Nagatsuta, Midori-ku, Yokohama, 226-8503, Japan
E-mail: okada99@isl.titech.ac.jp

Osamu Hasegawa
Imaging Science and Engineering Lab.
Tokyo Institute of Technology
Yokohama, Japan
PRESTO, JST
E-mail: hasegawa@isl.titech.ac.jp

Abstract– This study is intended to realize a flexible learning mechanism that can recognize and generate a time-series pattern (dynamic pattern) in the real world. The mechanism in this study comprises a mixture-of-experts (MoE) system. Experts of the MoE are small non-monotonous neural networks. The mechanism divides self-organized patterns into primitives. Each expert learns the pattern primitives; it can then recognize and generate the primitives. Learning patterns are expressed as a permutation of primitives that are output by the MoE, recognized, and then generated by applying the permutation. In addition, this mechanism can learn new patterns incrementally by increasing the number of experts.

1. INTRODUCTION

In recent years, many studies have aimed at realizing both recognition (union of time-series patterns and symbols) and generation of time-series patterns (gesture, sound, and others). In the future, such technology will provide an important basis for humanoid robots to perform imitative learning and possibly communicate with humans. In the real world, it is difficult to recognize time-series patterns included noise, deficit, expansion and contraction as a symbol and to generate it.

Regarding this problem, Simozaki [2,3] and Inaba [4] used the self-organized algorithm in the middle layer of non-monotone neural networks that Morita [1] formulated. Their mechanisms permitted the recognition of time-series patterns and their generation. By relating a time-series pattern to a symbol, the mechanism in these studies learns time-series patterns, recognizes them, and becomes capable of generating them (Fig. 1 right). These mechanisms require N-independent trajectory attractors to memorize N time-series patterns. In our method, many small non-monotonous neural networks learn a time-series short pattern that has the role of a primitive. Our method divides them and relates them to a small network (Fig. 1 left).

Here, time-series short patterns that serve as primitives correspond to a symbol (this symbol is called a primitive symbol in this paper). The permutations of primitive symbols that are output by networks express the time-series pattern that is input. Primitive symbols can recognize the common primitive of all time-series patterns and generate it. In addition, symbols can express other time-series patterns. Furthermore, our mechanism can add new primitives that are needed to express unknown patterns without breaking the network structure after learning. Few or no systems can enable incremental learning of a time-series pattern. The permutation of primitive symbols is output from a lower layer. Next, in the upper layer, our mechanism distinguishes the permutation by comparing permutations: it relates the permutation to a large symbol (termed a "pattern symbol" herein (Fig. 1)) and recognizes time-series patterns that are input. We summarize this section below. By dividing the long time-series pattern that is input and expressing the time-series pattern as the permutation of primitive symbols, the mechanism has the following effects. In the course of the generation process, our mechanism can generate a time-series pattern by outputting the primitive along with the permutation of primitive symbols. In addition, our mechanism can generate a pattern that contains average patterns that are recognized as identical class. By applying the mechanism above, we hope to achieve a system that can learn a pattern compactly and incrementally, recognize it (produce a union between a time-series pattern and a symbol), and generate it flexibly.

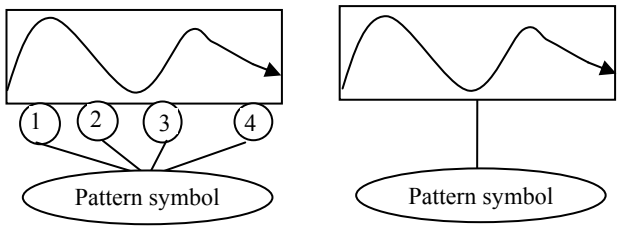

Fig. 1: Correspondence of the symbol to the pattern
(Small balls in the left are the primitive symbols.)

2. OUTLINE OF THE PROPOSED ALGORITHM

2-1 System outline

For structuring and dividing time-series patterns, we used Mixture-of-Experts (MoE) [5], which comprises many small non-monotone neural networks (Fig. 2). Because this system differs from a usual MoE system, our MoE system allows both

recognition of the time-series pattern and its generation; moreover, corresponding to unknown patterns, the system can add an expert. In the upper layer, by applying DP matching algorithm, the system classifies the permutation of primitive symbols that are output from MoE system and recognizes the time-series pattern.

2-2 Learning algorithm

2-2-1 Learning method

We break the time-series pattern down into time-series short patterns. Then we input the short patterns continuously. The system compels the expert (which takes a maximum output value (*) in response to each time-series short pattern) to learn, then repeat. Each expert gradually comes to express the separate time-series short pattern (primitive) included in the time-series pattern. Consequently, the system learns the time-series pattern and divides it.

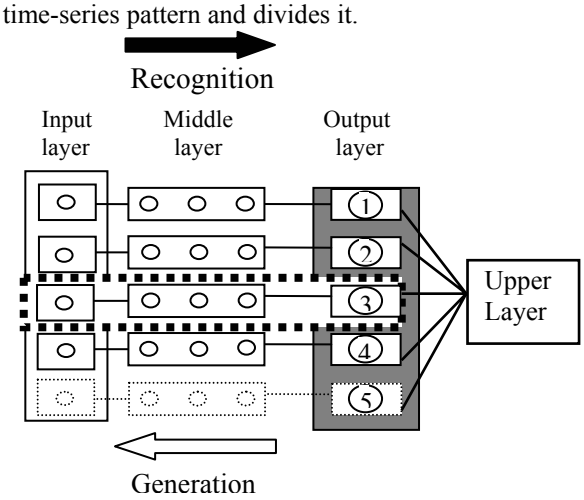

Fig. 2: The MoE system we produced
(In Fig. 2, white balls represent neurons. All neurons are connected. The gray part represents output. This part recognizes patterns. The thin dotted line enclosure is the newly added expert that. The thick dotted line enclosure corresponds to an expert. We input the same pattern from the input layer to each expert. The number of experts indicates the permutation of primitive symbols.)

2-2-2 Recognition and generation

Using the output from experts, the system obtains the permutation of primitive symbols. Subsequently, the system applies DP matching and performs recognition by comparing the permutation of the primitive symbol numbers of a test pattern with the permutation of primitive symbol numbers of the recognized pattern.

2-2-3 Correspondence to the unknown pattern

When we input an unknown pattern, the system attempts to express the unknown pattern by applying experts that have finished learning. When the unknown pattern contains a part (primitive) which can not be expressed, all the experts take out the lower output value. In addition, the MoE system compels experts (which have not been applied yet) to learn the part. If such experts do not exist, MoE system adds a new expert automatically. Here, we set the threshold for evaluation of output value experimentally.

2-3 Outline of the expert of MoE

Experts used by the proposed MoE system must recognize a time-series short pattern and generate it. In this study, a non-monotonous recurrent neural network (RNN)
[1, 2, 3] is used as a learning unit which can realize this object. The dynamics of RNN (which we apply for the system) are shown as the following equation.

$$\tau \frac{du_i}{dt} = -u_i(t) + \sum_{j=1}^{n} w_{ij} y_j + z_i \tag{1}$$

Here, the output equation is the following.

$$y_i = f(u) = \frac{1-e^{-c_1 u}}{1+e^{-c_1 u}} \cdot \frac{1+\kappa e^{c_2(|u|-h)}}{1+e^{c_2(|u|-h)}} \cdot \frac{1}{1+e^{-\upsilon(u-\mu)}} \tag{2}$$

The network trains according to the following equation.

$$\tau' \frac{dw_{ij}}{dt} = w_{ij}(t) + \alpha \operatorname{sgn}(u_i) f_w(u_i, z_i) r_i y_j \tag{3}$$

Into eq. (3), the following can be substituted.

$$f_w(u_i, z_i) = \frac{1-e^{-c_1 u}}{1+e^{-c_1 u}} \cdot \frac{1+\kappa e^{c_2(|u|-\beta|z|)}}{1+e^{c_2(|u|-\beta|z|)}} \tag{4}$$

The network comprises the input layer's 18 neurons, the middle layer's 58 neurons, and the output layer's 59 neurons. The output value of the neuron of an output layer is equivalent to the output value of (*). In addition, to advance specialization of learning of expert, the input intensity to the middle class and to an output layer was strengthened. It was changed gradually to a stronger signal from a weaker signal during learning.

2-4 DP matching algorithm

The permutation of primitive symbols is output from MoE. Because this symbol sequence includes expansion and contraction, it uses DP matching as the technique for comparing and clustering this permutation. When the matching passes ϖ (i(l),j(l), l=1,2,…) are given, the distance between patterns is as that shown in Eq. (5).

$$D(A,B;\varpi) = \sum_{l} d(a_i(l) - b_j(l)) \tag{5}$$

In that equation, $d(a_i(l) - b_j(l))$ is the distance $a_i(l)$ between $b_j(l)$, and l expresses the length of a matching pass. Next, when the matching pass is not given, $D(A,B)$ (the distance between patterns) is defined as the minimum $D(A,B;\varpi)$ in all matching passes ϖ .

$$D(A,B) = \min\{D(A,B;\varpi)\} \tag{6}$$

We define cumulative distance $g(i,j)$ as in (7).

$$D(A,B) = g(i,j) = \min\{g(i,j;\varpi)\} \quad (7)$$

By applying recurrence formula, we solve $g(i,j)$ and obtain the distance between patterns: i is the distance of pattern A and j is the distance of pattern B.

$$g(A,B;\varpi) = \sum_{l} (a_i(l), b_j(l)) \quad (8)$$

When the matching pass $\varpi((i(0),j(0),\cdots,(i(l),j(l))$ is given, $g(A,B;\varpi)$ is the cumulative distance from $(0,0)$ to (i,j).

We define recurrence formula $g(i,j)$ as in (9)

$$g(i,j) = \min\begin{cases} g(i-1,j) + 2w_{ij}d(i,j) \\ g(i-1,j-1) + w_{ij}d(i,j) \\ g(i,j-1) + 2w_{ij}d(i,j) \end{cases} \quad (9)$$

According to the recurrence formula (Eq. (9)), we obtain the degree of similarity between patterns.

In Eq. (9), w_{ij} which is decided as below by patterns, is the weight.

$$w_{ij} = number_i \times number_j \quad (10)$$

number is the continuous value of the permutation of primitive symbols. If the permutation of a primitive symbol is $(1,1,1,1,4,4,4,9\cdots\cdots)$, then the continuous value of 1 is 4 and the continuous value of 4 is 3, as shown below.

$$number_0 \sim number_3 = 4,$$

$$number_4 \sim number_6 = 3$$

$$number_i = \begin{cases} 1 & number < boundary \\ number_i & number \geq boundary \end{cases} \quad (11)$$

It is thought that the portion that the same symbol follows for a long time in the permutation expresses the important part. In cases where this part was missing, weight was given so that an error value became large. The degree of similarity between patterns is computed with the above algorithm.

3. EXPERIMENT

3-1 Outline of experiment

This experiment examined the division ability of a proposed system, its recognition and its generation ability. Five kinds of motion images (M1, M2, M3, M4, and M5) were applied as input time-series patterns. Appearances (Fig. 4) (M1: clockwise rotation, M2: a counter clockwise rotation, M3: movement toward the right and left, M4: movement toward the upper and lower sides, and M5: the body was opened and closed and aslant motion) that three experimenters moved indoors were used as the input image.

First of all, 15 patterns, which had three patterns per kind (M1–M5), were input in a random order. These patterns were learned and divided sequentially and incrementally. Next, 15 patterns, which were output as the permutation of primitive symbols from MoE, were classified using the DP matching algorithm. Here, these patterns were classified by applying only a boundary value, without inputting the teacher signal. Second, 10 patterns had two patterns per type. We applied these patterns to test them and examined the recognition accuracy. Third, we experimented to test generation by inputting the permutation of primitive symbols from the output layer of MoE.

In addition, we tested the generation of common parts (common permutations of primitive symbols) of patterns which were classified as being of the same class. Thereby, we verified whether the generated pattern expressed the motion of the class and whether it would be the average pattern of this class. As above, we experimented five times, changing the test patterns and the learning patterns. In addition, the test patterns and learning patterns that were applied in this experiment included many deficits, expansions and contractions because these patterns were composed of human motions.

3-2 Image processing

The image processing method that was applied in this experiment is represented below. After smoothing in each frame of input images, we calculated the difference between frames and extracted a self-correlation feature of the time-series direction. Here, a 3 × 3 mask was applied to calculate the correlation feature. The feature of direction was expressed by the position of a mask which had a large value. For extracting the motion direction of the human subject, three positions of a mask that had large value and the sum of the position number were expressed as a binary number: it became the input vector. However, if the position number or the sum of the position number was odd, they were expressed as 1's complement of binary number. The number of experts contained in the MoE was 50.

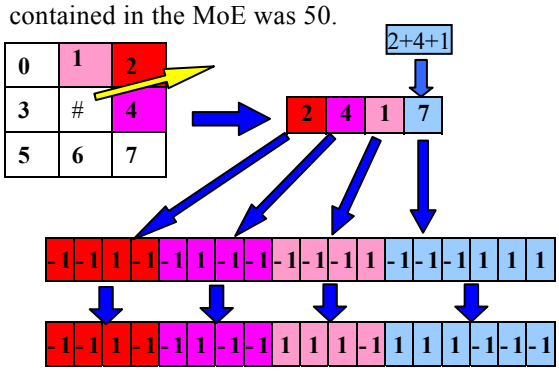

Fig. 3: Example of input composition vector

(In Fig. 3, if # is in the center, the number expresses the direction on the mask side. This example represents that motion which has direction "2" closed to "4"(yellow arrow). The permutation of "2 4 1 7" is expressed as the binary number and 1's complement of them)

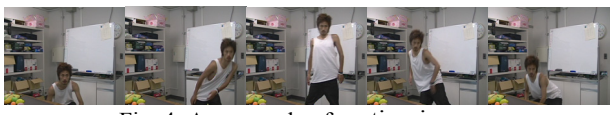

Fig. 4: An example of motion images

(Fig. 4 represents the appearance of motion M2)

4. EXPERIMENTAL RESULT

4-1 Recognition result

The time-series patterns are learned and divided by the MoE. The resultant output, as the permutation of primitive symbols, is shown in Fig. 5. Here, the horizontal axis expresses the time-series number; the vertical axis expresses the number of experts.

The first experiment compared the permutation of primitive symbols of three patterns of the class of M1. M1 and M2 are images in which a human draws a circle along an outline. Individual differences among people also tend to arise from these patterns compared with other kinds of patterns. Nevertheless, experiments with the time-series numbers (ca. 50–100) show that the degree of similarity among the three patterns is large. Moreover, the degree of similarity of the output permutation of primitive symbols is large between other kinds of patterns. This result showed that common parts (in Fig. 5, time-series number 50–100) between patterns of the same kind can be expressed using the same primitive (symbol) by making a primitive into a symbol. Moreover, even if there were expansion, contraction and deficit portions (in Fig. 5, the time-series number is 1–40, 100–120), the information on these parts was not lost, but these portions were expressible using the different primitive.

Results of the second experiment, which classified the permutation of primitive symbols by DP matching, are shown in Fig. 6. Figure 6 shows the matching result between the pattern that belongs to M1 represents the number of the learning pattern and the vertical axis represents the error value of DP matching. From Fig. 6, it can be inferred that the patterns of the same class as this pattern are the 9th and the 13th.

A boundary value set up used because the judgment standard was performed experimentally. As mentioned above, it was clustered by performing DP matching to all 15 data. In addition, we gave no information that there are three patterns of this class, and that there is a pattern of five classes. They clustered mechanically according to the boundary value. Regarding the test patterns, they shall be recognized as the class that pattern which had the smallest error value belonged to.

Table 1 lists recognition results for the five experiments. In Table 1, the length of the yellow portion represents the experiment times; the width shows respective class patterns (from M1 to M5). The pink portion expresses the number with which the learning pattern of each class has been recognized among three patterns: sum 1 expresses the number of the sum with which recognition of a learning pattern was successful. The blue portion expresses the number with which the test pattern of each class was recognized among the two patterns; and sum 2 expresses the number of the sum with which the recognition of a test pattern was successful. Excepting the third time and the fourth time, we succeeded in discriminating in five classes about learning patterns. Also regarding the test pattern, an approximate 90% success rate was obtained.

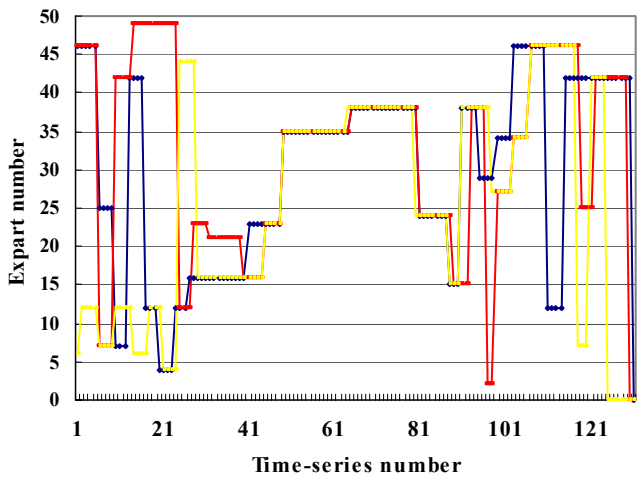

Fig. 5: Permutation of symbols was output by MoE
(A comparison of the division symbol sequence of three patterns of the class of M1 is shown in the first experiment. The horizontal axis expresses the time-series turn (time-series number) and the vertical axis expresses the number of experts.)

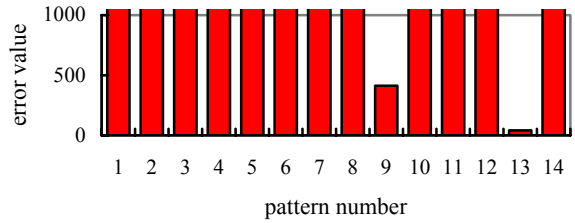

Fig. 6: A matching result of patterns
(The matching result between one pattern that belongs to M1 and the 14 remaining patterns. The error values of the 9th and the 13th patterns much smaller than the others)

Table 1: A recognition result of five experiments
(The pink (blue) portion expresses the number with which the learning (test) pattern of each class was recognized among three (two) patterns.)

	M1		M2		M3		M4		M5		Sum 1	Sum 2
1	3	2	3	2	3	2	3	2	3	2	15	10
2	3	2	3	1	3	2	3	2	3	2	15	9
3	3	2	2	1	3	2	3	2	3	2	14	9
4	3	1	2	2	3	2	3	2	3	2	14	9
5	3	1	3	1	3	1	3	2	3	2	15	7

4-2 Generation result

The result that has generated one which belongs to M2 of the first experiment is shown in Fig. 7. Figure 7 shows the locus (red line) that generated the pattern that was generated by each expert along with the locus (black line) that restored the pattern that is obtained from the actual input image. Change of a minute direction is approximated and is smooth. The turning point (at the time series number), the point at which direction changes, is equal to that of the locus which restored the pattern that was actually obtained in almost all places. Therefore, the locus of a generation pattern can generate the input pattern well.

Experimentation confirmed that the system is capable of similar generation of patterns of other classes. The turning point (at the time series number) is equal for two loci. For that reason, the locus of a generation pattern can generate the input pattern. This experiment confirmed that the pattern of other classes can also be generated. Figure 8 shows a result whereby we generated a permutation by extracting common parts of the permutation of primitive symbols of three patterns of M2. Compared with the locus (Fig. 7) which generated one pattern that belonged to M2, this generation pattern approximates the locus of an ellipse. Quantitative evaluation of this approximation is difficult because there is no candidate for evaluation for this generated locus.

However, if an intelligent system can recognize and generate incrementally, it is thought that a locus that has been generated by common parts serves as the locus that expresses the class best. If we replace this argument by imitative learning, we infer that it is more natural to transpose that imitation to one's action gradually, reflecting the learning pattern as it learns rather than a robot which uses only the action learned first as its own action. This generation method is important for this reason.

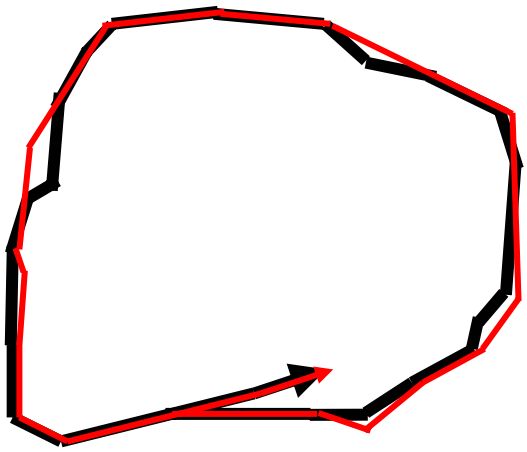

Fig. 7: The locus generated the pattern
(The locus (red line) which generated the pattern that was generated by each expert is compared with the locus (black line) which restored the pattern that was obtained from the actual input image.)

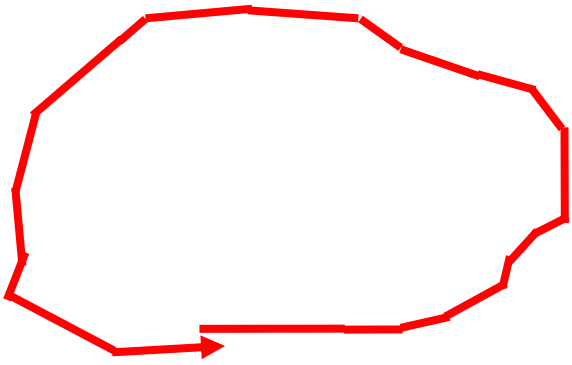

Fig. 8: Generation of common parts of the permutation of symbols

5. CONCLUSION

As mentioned above, we proposed a mechanism that learned time-series patterns and divided them in a lower layer by applying MoE; it integrated the pattern again in the upper layer. Moreover, it could recognize and generate. Furthermore, this mechanism could learn incrementally. Experimentation confirmed that flexible generation, which is not available in the conventional technique, was possible. Hereafter, to realize a system that recognizes the complex time-series pattern and can generate it. we intend to realize flexible and compact learning of a time-series pattern, recognition of it (union of time-series patterns and symbols), and generation of it. We shall also explore application of this mechanism to a humanoid robot.

REFERENCES

[1] M. Morita. “Memory and learning of sequential patterns by non monotone neural networks.” Neural Networks, 9:1477–1489, 1996

[2] M. Shimozaki and Y. Kuniyosi. “The neural network model that predicts situations based on memories” In Proc. Annual Conf. of Robotics Soc. of Japan, pp.3H15, 2002

[3] M. Simozaki, T. Inamura and Y. Nakamura. ”Recognition memorization and generation of human motion pattern by an associative memory.” In Proc. ROBOMEC ‘01, pp.2P1–B11, 2001

[4] T. Inamura, Y. Nakamura and M. Simozaki. “Integration of behavior recognition and generation processes based on associative memory.” In Proc. Annual Conf. of Robotics Soc. of Japan, pp.1237–1238, 2001

[5] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. “Adaptive mixture of local experts.” Neural Computation, 3: 79–87, 1991