Automaton-Based Anticipatory System

Tadashi Ae, Hiroyuki Araki, Keiichi Sakai
Electrical Engineering, Faculty of Engineering, Hiroshima University
1-4-1 Kagamiyama, Higashi-Hiroshima, 739-8527 Japan
FAX: +81-824-22-7195
Email: ae@aial.hiroshima-u.ac.jp.

Abstract

A lot of research for anticipatory systems have been reported, where the chaotic equation including the hyper incursion equation plays an important role. The neural network model is also included in such a category, and will continue to be discussed. From the viewpoint of computer systems, however, we have proposed a hybrid system architecture mixed with neural network and artificial intelligence, where the two-level structure is introduced; the first layer: a neural network, and the second layer: an automaton system. On the two-layered system, the automaton part is dominant for anticipation, because the state transition is made by an automaton behavior although the selection among transitions is made by a neural network. In this paper, we discuss an automaton-based anticipation, since it is appropriate to discuss anticipation together with learnability.

Keywords

Inductive Learning, Automata Learning, Genetic Programming, Anticipatory System

1 Introduction

A lot of research for anticipatory systems have been reported, where the chaotic equation including the hyper incursion plays an important role [Dubois1997]. The neural network model which we are focusing on is also included in such a category, and will continue to be discussed, since the original model of neural networks [McCulloch 1957]is not enough anticipatory. From the viewpoint of computer systems, however, we have proposed a hybrid system architecture mixed with the neural network and the artificial intelligence [Kawada *et al.*1995], where the two-level structure is introduced; the first layer: a neural network, and the second layer: an automaton system. We have also proposed a new learning algorithm AST (Abstract State Transition algorithm) for the two-layered system, where the AST is a hybrid learning mixed with a neural network learning and an automaton learning [Ae *et al.*1998a, Ae *et al.*1998b].

On the two-layered system, however, the automaton part is dominant for anticipation, because the state transition is made by an automaton behavior, although the selection among transitions is made by a neural network. In other words, the global behavior is represented by an automaton, and the local behavior is done by a neural network.

In this paper, we introduce first an automaton learning, and discuss the learning mechanism which affects the anticipatory ability. Next, we refer to a system realization for such a direction.

2 Learnability

In general, system S is defined as

$$O = S(I)\ ,$$

where I and O are the input and the output, respectively. Both I and O are assumed to be a language each. More precisely, a language is defined as a set of words, where a word w is an element in the set of concatenated alphabets W^* (where W is the set of alphabets). (Note that a set W is extended to a larger set than a set of conventional alphabets, e.g., a set of bit maps of 512×512.)

i) State-less System; $O = S(I)$.

The feedforward neural network corresponds to this case. We have known several excellent models for learnability (e.g., AIC, PAC, ...). For the anticipation on the time-axis, however, we focus on the next case.

ii) State Transition System; $O = S(I, Q)$, where Q is the set of states.

When using the conventional model of state transition system, we have

$$Q(t+1) = S_1(I(t), Q(t)), \text{ and}$$
$$O(t) \quad = S_2(I(t), Q(t)),$$

where the discrete time is introduced and S is divided into S_1 and S_2.

We focus on S_1, since the essential behavior is represented by S_1. An instance of input sequence $w = i_1 i_2 \cdots i_k$ is regarded as a word in $L(S)$, where $L(S)$ is the language that system S (i.e., automaton S) accepts. Angluin introduced MAT (Minimal Adequate Teacher) Learning [Angluin1987], one of which is represented as follows;

Question Type1 : w in $L(T)$?

Type2 : $L(S)$ equivalent to $L(T)$? ,

where S and T are an automaton that the learner is constructing and an automaton that the teacher provides, respectively. In this case (really, a typical case) the learner is allowed to ask two types of questions to the teacher, and must construct automaton S that is equivalent to T.

In MAT learning,

Strong Condition : *Teacher knows T*

is introduced. Therefore, only the classes where the equivalence problem is decidable are applied to MAT learning (DFA, Counter Automata, Linear Language Automata, ... etc.,

where DFA means Deterministic Finite Automata). As is well-known, the equivalence problem of pushdown automata is undecidable (although the deterministic case is still open), and therefore, we focus on a proper subclass of pushdown automata on MAT Learning. Learnability on MAT learning is steady, but is not creative. For anticipatory system we need to exclude Strong Condition: Teacher knows T , and can introduce a class of automata where the equivalence problem not necessarily decidable.

3 Anticipation

An anticipatory system is a system which contains a model of itself and/or its environment in view of computing its present state as a function of the prediction of the model [Dubois1997]. For the anticipatory system in this paper, however, an equational model is not provided, because the computing model is only procedurally defined. The learning mechanism provides a function of the prediction of the model, and we continue to discuss the learnability.

Let us exclude simply Strong Condition: *Teacher knows* T. Then, one cannot evaluate S that the learner constructs, if no condition for system equivalence is assumed. Therefore, we introduce the following;

Weak Condition: *Teacher knows partially* T.

Actually we introduce the following learning mechanism.

1) Elementary Learning

T_i $(i=1,2,\cdots,p,p+1,\cdots,p+q)$ is a class of automata. The teacher knows each T_i for $i=1,2,\cdots,p$, where MAT learning is applied. The teacher , however, does not know T_i for $i=p+1,\cdots,p+q$ (The latter may be empty.). The class of language, $L(T_i)$ represents the language which each T_i accepts.

2) Anticipatory System

Essentially the new words will be born by combination of a lot of languages, to some of which MAT learning is already applied and to some of which it is not. How to combine them is not described here, but shuffling (including modulated shuffling) plays an important role of operation for combination. These operation is represented by the composition of automata, and several types are given by the operation on the language.

The total system T is constructed by a subset of $\{T_i\}$, where $i = 1,2,\cdots,p,p+1,\cdots,p+q$. The class of language, $L(T)$ represents the language which T accepts. The composition of automata is more complex than that of languages [Ae1977], and therefore, we explain here the case of language composition.

Let us assume that $L(T)$ be represented by a logical (Boolean-like) formula of $L(T_i)$.

Example1.

$L(T) = L(T_1) \cup L(T_2) \cup \neg L(T_3)$, where $\cup$ is the union of sets and $\neg$ is the negation. The teacher knows $L(T_1)$ and $L(T_2)$, but does not know $\neg L(T_3)$.

Example2.

$L(T) = L(T_1) \cap L(T_2) \cap L(T_3)$, where $\cap$ is the intersection of sets. Note that $L(T)$ may become an outer class even if $L(T_i)(i = 1, 2, 3)$ is in the same class, all of which the teacher knows. (This means the case where the language is not closed for such a set operation.)

For practice we focus on the type of Example2, i.e., the case where the total language, $L(T)$, may create new words, although all elementary languages are trained by MAT learning. Namely, the composition of automata plays an important role in anticipation.

4 System Realization

The composite system is fundamentally represented as in two types, i.e., the parallel composition (in short, PC) as in Figure 1, and the cascade composition (in short, CC) as in Figure 2 [Ae1977].

In the language expression, PC corresponds to the intersection of languages, but CC does not. For both cases the concept of language is used for anticipation.

The anticipation in a language means a creation of "New Words", where the set of words corresponds to the set of automata (Figure 3).

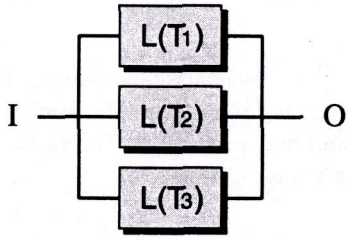

Figure 1. Example of PC (Parallel Composition)

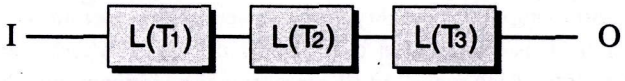

Figure 2. Example of CC (Cascade Composition)

Creation Procedure of "New Words" :

0) Setting Initial Pool; {Set of $L(T_i)$ is after MAT learning is used}

repeat 1) Pool Re-Setting; {Pool revised}

2) Generation Procedure; {GP-like procedure using Weak composite system}

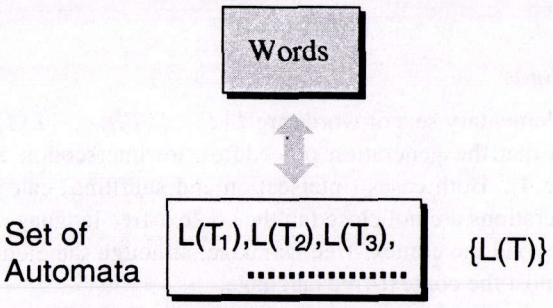

Figure 3. Words vs. Set of Automata

3) Filtering using Structural Condition; {Composite system condition}
4) Evaluation using Neural Network; {Special evaluation equation}
until "New Words" are born .

2) is used for generating Hypothesis, and 3) and 4) are for Verification or Evaluation, where 4) may be omitted sometimes. The global flow is represented as in Figure 4, which is similar to the GP (Genetic Programming) procedure [Koza1992, Aler *et al.*1998] and also to the induction cycle in ART (Adaptive Resonance Theory) [Carpenter and Grossberg 1988]. The anticipation (birth of "New Words") is made in 2), but it will be completed after a lot of cycles have been repeated, and it is determined by the evaluation.

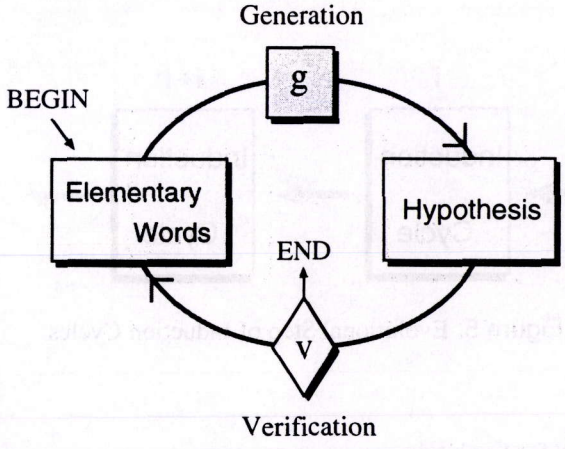

Figure 4. Induction Cycle

5 Evolution

5.1 Birth of New Words

Suppose that the elementary sets of words are $L(T_1), L(T_2), \cdots, L(T_p)$, where all sets are trained each and that the generation procedures are intersection and shuffling (as operation g in Figure 4). Both cases (intersection and shuffling) can produce the new words, since both operations are not closed in the context-free language. The new words are expected to be beyond the context-free language, although the elementary language is supposed to be at most the context-free language.

5.2 Evolution

Suppose that we obtain several new words after the induction cycle in Figure 4 was repeated. If the i-th stage is terminated, then the stage goes into the $i+1$-th stage, where $i = 1, 2, \cdots$. Figure 5 shows the evolution step of induction cycles. How to determine the end of each stage? This is important and the most difficult problem. The structural assumption will be required in practice, and depends on the application. First, a stage of induction cycle is regarded as an automaton, which belongs to the class of pushdown automata, since the language is supposed to be at most context-free. Then, the interconnection between two stages is regarded as that of two pushdown automata. More precisely, the interconnection between two automata (i.e., languages), $L(T_i)$ and $L(T_j)$, is represented as in Figure 6, but the control mechanism is unknown. If the mechanism is created after a stage of induction cycle, then the stage can go to the next step (Note that the intersection or the simple shuffling is a known control mechanism and, therefore, it is prepared for the initial condition.). One of unknown control mechanisms should be born (Figure 7).

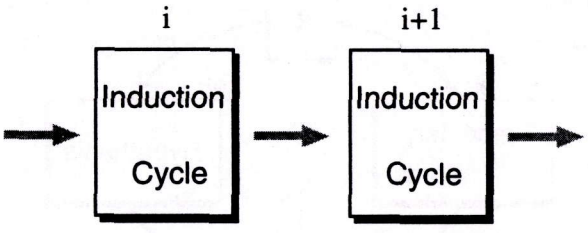

Figure 5. Evolutional Step of Induction Cycles

5.3 Automata View for Evolution

On the evolution step, the $i+1$-th stage must be higher than the i-th stage, and the evolution step in Figure 5 is from i-th to $i+1$-th, that is, one-directional.Then, the direction of interconnection in Figure 6 is also one-directional. The present cannot look

Control

Figure 6. Control from $L(T_i)$ to $L(T_{i+1})$

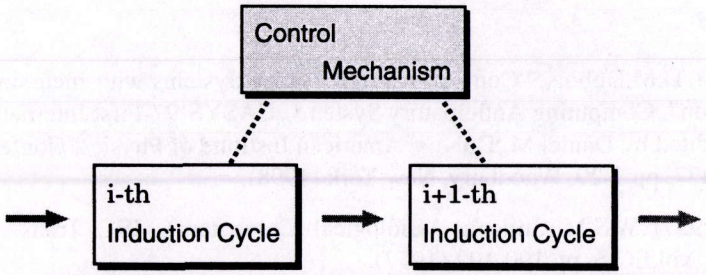

Figure 7. Control Mechanism for Evolution on Induction Cycles

at the future in the world of determinism. The automata theory, however, includes non-determinism, which is also important in the class of pushdown automata. If the non-determinism is allowed in the i-th stage ($L(T_i)$), the interconnection becomes equivalent to be two-directional, because the automaton of $L(T_i)$ can guess the behavior of $L(T_j)$ and the automaton of $L(T_j)$ can verify its guess [Ae1977]. On the evolution step, the i-th stage guesses the behavior of higher ($i+1$-th) stage, and its report is sent to the $i+1$-th stage. The $i+1$-th stage will judge whether the report is really "evolutional" or not. The $i+1$-th stage is a teacher for the i-th stage, and the i-th stage cannot ask the equivalence question (i.e., Question Type2: $L(S)$ equivalent to $L(T)$?). More precisely, the i-th stage cannot receive the answer, even if it sends the question. This is not essential for Question Type1, since the elementary system are already trained.

For the Question Type2, however, this is the essential defect, and therefore, the non-determinism plays an important role of direction. The problem for the interconnection of pushdown automata remains still open for the case of the determinism, i.e., the problem whether or not the one-direction is properly less strong than two-direction.

6 Conclusion

This paper is summarized as follows; From the viewpoint of computer systems, we are constructing a hybrid system architecture mixed with neural network and artificial intelligence [Ae *et al.*1998b], where the two-level structure is introduced; the first layer: a neural network, and the second layer: an automaton system. On the two-layered system,

however, the automaton part is dominant for anticipation, because the state transition is made by an automaton behavior although the selection among transitions is made by a neural network. In this paper, we discuss an automaton-based anticipation, since it is appropriate to discuss anticipation together with learnability. The evolutional step is controlled by a parameter as in the conventional constructive inductive learning, but it remains to be discussed.

References

[Dubois1997] D.M.Dubois, "Computing Anticipatory Systems with Incursion and Hyperincursion", Computing Anticipatory Systems: CASYS'97-First International Conference, edited by Daniel M. Dubois, American Institute of Physics, Conference Proceedings 437, pp.3-29, Woodbury, New York (1998).

[McCulloch1957] W.S.MuCulloch, "Biological Computers", IRE Trans. Electronic Computer, Vol.EC-6, pp.190-192 (1957).

[Kawada *et al.*1995] M.Kawada, X.Wu, T.Ae, "A Construction of Neural-Net Based AI Systems", Proceedings 1st IEEE ICECCS, edited by Alex Stoyenko, pp.424-427, Ft. Lauderdale (Nov. 1995).

[Ae *et al.*1998a] T.Ae, H.Araki, K.Sakai, "Learning Algorithm for Structured Brain Computer", Proceedings CISST'98, edited by Hamid R. Arabnia, pp.17-24, Las Vegas (June 1998).

[Ae *et al.*1998b] T.Ae, H.Araki, K.Sakai, "Structured Brain Computer and its Learning", Computing Anticipatory Systems: CASYS'98-Second International Conference, edited by Daniel M. Dubois, American Institute of Physics, Conference Proceedings 465, pp.111-120, Woodbury, New York (1998).

[Angluin1987] D.Angluin, "Learning Regular Sets from Queries and Counterexamples", Information and Computation, Vol.75, pp.87-106 (1987).

[Ae1977] T.Ae, "Direct or Cascade Product of Pushdown Automata", J. Computer and System Sciences, Vol.14, pp.257-263 (1977).

[Koza1992] J.R.Koza, "Genetic Programming: On the Programming of Computers by Natural Selection", MIT Press (1992).

[Aler *et al.*1998] R.Aler, D.Borrajo, P.Isasi, "Genetic Programming and Deductive-Inductive Learning: a Multi-strategy Approach", Proceedings ICML'98, edited by Jude Shavlik, pp.10-18, Madison (1998).

[Carpenter and Grossberg1988] G.A.Carpenter, S.Grossberg, "The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network", IEEE Computer, Vol.21, No.3, pp.77-88 (1988).