Biological Cybernetics, Vol.72, pp.365-370, 1995.

Embedding a Grammatical Description in Deterministic Chaos: An Experiment in Recurrent Neural Learning

Jun TANI and Naohiro FUKUMURA

Sony Computer Science Laboratory Inc.

Takanawa Muse Building, 3-14-13 Higashi-gotanda, Shinagawa-ku,Tokyo, 141 JAPAN

Tel 3-5448-4380, Fax 3-5448-4273, Email tani@csl.sony.co.jp

(Pubished in *Biological Cybernetics*, 72, 365-370, 1995)

Abstract

We consider the modeling process of a "biological" agent by combining the concepts of neuroinformatics and deterministic chaos. We assume that an agent observes a target process as a stochastic symbolic process, which is restricted by grammatical constraints. Our main hypothesis is that an agent would learn the target model by reconstructing an equivalent quasi-stochastic process on its deterministic neural dynamics. We employed a recurrent neural network (RNN), which is regarded as an adjustable deterministic dynamical system. Then, we conducted an experiment to observe how the RNN learns to reconstruct the target process, represented by a stochastic finite state machine in the simulation. The result revealed the capability of the RNN to evolve, by means of learning, toward chaos which is able to mimic a target's stochastic process. We precisely analyzed the evolutionary process as well as the internal representation of the obtained neural dynamics. This analysis enabled us to clarify an interesting mechanism of the self-organization of chaos by means of neural learning, and also showed how grammar can be embedded into the evolved deterministic chaos.

1 Introduction

An intelligent agent should have the ability to build a dynamical model of its own environment. The agent, using the model, can conduct lookahead prediction and planning. The essential question raised here is how an agent can acquire the model.

Crutchfield (1991) presented an interesting concept for the modeling process, based on the dynamical system's view. In his formulation, the target process to be modeled is viewed as stochastic system defined on real numbers. An agent observes the target process through a measuring apparatus that offers finite measuring resolution. The measuring resolution is usually determined by the modeler's bias toward what is worth observing against noise. If noise affects the target process or the measuring process substantially, any attempt at high resolution measurement becomes meaningless since precise information relating to the original process has already been lost. The measuring apparatus acts as a transducer that samples the process value and maps it onto a finite symbol. Therefore, it can be said that the modeler observes the original dynamical process as a stochastic symbolic process. Crutchfield (1991) assumed such a stochastic symbolic process to be internally preserving stable grammatical constraints. He formulated an algorithm by which the modeler can extract essential grammar, in the form of statistical finite state machine (SFSM), from the observed sequence.

In this paper, we present another interpretation of the above modeling process that would be more biologically plausible (see Figure 1).

– Fig. 1 –

We assume that an agent possesses an internal neural system that is , mathematically, a deterministic dynamical system with an adjustable parameter set. In learning the target process, its dynamical structure is modulated, by means of weight adjustment, such that its time evolution mimics the original process. If we adopt Crutchfield's view of "quantization by observation", we face a basic problem: how a deterministic dynamical system can mimic the observed stochastic symbolic process. The answer can be found in a mathematical framework of symbolic dynamics (an excellent introduction can be found in (Hao Bai-lin, 1989)), which we briefly review below.

Symbolic dynamics can be defined for a given dynamical system based on real numbers with finite divisions of its state space, a unique symbol being assigned to each cell. The trajectory in the original state space is transformed into symbolic sequences by this definition. If the original dynamical system is characterized as a fixed point or limit cycling dynamics, the symbolic dynamics also converge to one symbol or to a periodic sequence of symbols. If the original dynamical system is chaotic (i.e. the trajectory is expansive in certain directions), the symbolic dynamics could be ergodistic, generating stochasticity. Such a stochastic symbolic process is subject to grammatical constraint, and it eventually composes language. It was shown that a simple parameterized unimodal map such as "logistic map" can generate varieties of language (including different classes of language) (Crutchfield and Young, 1989; Friedman 1991).

This notion of symbolic dynamics leads us to a hypothetical answer: the internal neural system evolves, by means of learning, toward a chaotic regime which is able to mimic a grammatically constrained stochastic symbolic process. We will conduct a simple experiment to test this hypothesis in the following. We employ a recurrent neural network (RNN) (Jordan, 1986; Pineda, 1987; Elman, 1990; Sato, 1990) that is regarded as a deterministic dynamical system. The main contrast of our research from previous one on RNN learning of grammar (Elman,

1990; Pollack, 1991; Giles, Sun, Chen and Lee, 1992) is that we discuss capability of RNN as a generator rather than a recognizer of language. We focuss on an RNN with self-feedback loop without inputs which realizes an autonomous dynamical system. Figure 2 shows illustrates our experiment. We assumed that an SFSM represents an observed process.

– Fig. 2 –

The sequence generated in the SFSM is fed to the RNN as example training data, after which the RNN adjusts the connective weights, in an attempt to mimic the sequence. This is a search process, on the neural manifold spanned across weight space, to find an appropriate dynamical function that can generate a specified language. We will examine the evolutional process of the RNN's internal dynamics, and investigate how grammatical constraint can be embedded into its internal representation.

2 Recurrent Network

The RNN employed in this study is a three-layered feed-forward network with state feed-back loops. It has an output unit but no input units (see Figure 3).

– Fig. 3 –

The dynamics of the RNN is given by the recurrent application of a map which is parameterized by the connective weight W, realizing an autonomous dynamics.

The state vector of current time, S_t, in the input layer is transformed to that of the next time, S_{t+1}, in the output layer,

$$S_{t+1} = F_s(S_t, W) \tag{1}$$

Output O_{t+1} is also mapped from the S_t, but by a different function.

$$O_{t+1} = F_o(S_t, W) \tag{2}$$

Note that S_t and O_t are a vector and a scalar of a real number. F_s and F_o can be embodied by means of a neural activation function:

$$\begin{cases} h_i = f(\sum_j w^h_{ij} s_{j,t} + b^h_i) \\ s_{i,t+1} = f(\sum_j w^s_{ij} h_j + b^s_i) \\ o_{t+1} = f(\sum_j w^o_j h_j + b^o_i) \end{cases} \tag{3}$$

where $s_{j,t}$ is the jth unit of the state vector in the input layer, $s_{i,t+1}$ is the ith unit of the state vector in the output layer, h_i is the ith unit in the hidden layer vector and o_{t+1} is the output. w^h_{ij}, w^s_{ij}, w^o_{ij} and b^h_i, b^s_i, b^o_i are connective weights and biases for hidden units, state units and output unit respectively. $f()$ is a standard sigmoid function.

The objective of learning is to find an optimal value of W that minimizes the energy E defined as the learning error between two sequences: the target sequence y_P and the output sequence o_P over the period P. With $o_p = F_o {F_s}^{p-1}(s_0)$ the energy function is,

$$E = \sum_{p=1}^{P} (y_p - F_o {F_s}^{p-1}(s_0))^2 \tag{4}$$

The optimal value of W, minimizing E, can be obtained through the iterative calculation of back-propagation through time (BPTT) (Rumelhart, Hinton and Williams, 1986). In this calculation the recurrent network is transformed into a cascaded feed-forward network without loops by duplicating the original three-layered network in the time direction. The generalized delta rule (Rumelhart, Hinton and Williams, 1986) is applied to the cascaded network to find the weight update vector at each sequence pattern l as $\Delta_l w_{ij}$, which is:

$$\Delta_l w_{ij} = \delta_{i,l} a_{j,l} \tag{5}$$

where $a_{j,l}$ is the output of the jth unit of the cascaded network. For the output unit, $\delta_{i,l}$ is calculated as

$$\delta_{i,l} = (y_{i,l} - o_l) f_i'(net_{o,l}) \tag{6}$$

where $f_i'(net_{o,l})$ is the derivative of the sigmoid function with an input summation of the output unit. For hidden units for which there are no teaching signals, $\delta_{i,l}$ is calculated recursively as the error signals back-propagate through the connections to the unit:

$$\delta_{i,l} = \nu f_i'(net_{i,l}) \sum_k \delta_{k,l} w_{i,k} \tag{7}$$

where ν is a decay coefficient that is set to a value between 0.0 and 1.0. The details of the role of ν will be explained later. The update weight vector for all sequences Δw_{ij} is obtained as a summation of (5).

To accelerate the learning speed, a momentum term is included in the update as

$$\Delta w_{ij}(n+1) = \eta \delta_{i,l} a_{j,l} + \alpha \Delta w_{ij}(n) \tag{8}$$

where n indexes the iteration time in the learning, and η and α are the constants of the learning rate and the momentum, respectively.

3 Procedure

This section describes the procedure of the experiment.

3.1 Target process

An SFSM, as shown in Figure 4 was assumed as the target process to learn.

– Fig. 4 –

This process repeats a basic sequence: (1, 0, 1/0). The generation of "1" or "0" followed by "10" is stochastically determined with even probability.

3.2 Training procedure

In the following paragraphs, we describe the specific methodologies and conditions applied to RNN training, which have not been presented in previous reviews of the generalized methodology of recurrent learning.

The employed RNN was three-layered, with three state units in the bottom layer, four hidden units in the middle layer, and one output unit and three state units in the upper layer.

In preparing the training data, a finite sequence of symbols is sampled from the source process. The sequence is divided into a set of window segments, each of equal length. The network is trained with this set of window segments in a parallel manner. The necessity for this segmentation arises from a typical characteristic of chaos: initial sensitivity. If chaos is dominant in the reconstructed dynamics, the learning of a sequence exceeding a certain time frame cannot converge because the output and the target sequences start to diverge in the time frame even if optimal connective weights are assigned. This implies that the length of each learning sequence should be set to a value shorter than this time frame. In our experiment, the window length is heuristically determined to be 12. The length of the total sample sequence is 204.

The error at each time in the window is back-propagated as a delta value through the state units of the cascaded network. Since the delta value could become unexpectedly large once accumulated to the bottom of the cascaded network, an adequate decay coefficient ν of (7) for back-propagating the delta should be set. In the delta computation of (7), ν is set to 0.8 when delta is computed for state units, and to 1.0 otherwise. The delta decays every time it is back-propagated to the state units.

The training process of the network should determine, in addition to the connective weights common to all segments, the optimal initial state for each window segment which minimizes the learning error. The ith unit of the initial state vector $s_{0,i}$ is iteratively calculated by the steepest descent utilizing δ information at the initial level as:

$$\Delta s_{0,i}(n+1) = \eta \delta_{0,i} + \alpha \Delta s_{0,i}(n) \tag{9}$$

This iterative search of the initial state is conducted for each window segment, simultaneously with that for the connective weights.

The training procedure employs parameter control for the learning rate η. Our tests revealed that learning proceeds through a precise structure of bifurcation and that, if the learning speed is too high, the structure is destroyed with showing a sudden increase in the error. Thus we introduced a heuristic control scheme in which the learning rate η is adjusted according to the learning error:

$$\eta = \beta E^{\gamma} \tag{10}$$

By this, the learning speed is polynominally reduced as the learning error E becomes smaller.

4 Simulation Results

4.1 Evolutionary Process in Learning

The computational experiment was conducted. Parameter settings were $\gamma = 1.5$, $\beta = 10^{-5}$, and $\alpha = 0.93$. The initial values of connective weights and biases were randomly set to between -1.0 and 1.0. The exact values are listed in the Appendix.

In our experiment, learning was terminated after 20000 iterations, at which point the learning process was nearly saturated. Figure 5 shows the time histories of the mean square learning error, the output trace of the internal dynamics, and the largest Lyapunov exponent.

– Fig. 5 –

Here, the output is a set of steady state solutions of o_i of (3) at each temporal point in the evolution, assuming that the connective weights evolves negligibly slowly. We noted that the internal dynamics evolve toward complexity through bifurcation. The largest Lyapunov exponent was negative at the beginning of learning, later becoming positive as learning proceeded. This implies that the internal dynamics are evolving toward chaos.

Let us examine the evolution process more precisely. The dynamics of a fixed point (near 0.5) continue through 1570 steps. Then, sudden tri-furcation takes place and limit cycling with a periodicity of three starts. The strategy that emerges here is to mimic the target process by constructing a sequence such as "1.0, 0.0, 0.5" rather than one of all "0.5", which results in the error being dramatically reduced. This limit cycling structure continues through 11950 steps, after which it starts to bifurcate, generating a limit cycling having a longer period. With a representation of symbolic dynamics in which an output of less than 0.5 is assigned to "0" and that equal or larger than 0.5 is assigned to "1", the sequence is represented by cycling such as "(101)(100)- - -". Chaos emerges from the 15102 steps. The Lyapunov exponent becomes positive at this point. Figures 6 (a) and (b) show the steady state symbolic sequences immediately before and at the onset of chaos.

– Fig. 6 –

The symbol sequences sampled after transient periods are represented by white (denoting 0) and black (denoting 1) squares from left to right in each row, downwards. The sequences existing immediately before the onset of chaos exhibit cycling with a periodicity of 36. That of the onset of chaos exhibits intermittent chaos having disorder bursts akin to noise on the basic cycling having a periodicity of 12. Although the observed sequence at the onset of chaos is grammatically correct, it seemed extremely biased in that "0" or "1" at the disorder is determined by a complex mechanism depending on long-past sequences. (c) shows the symbol sequences at 16600 steps in which the generation of the sequence becomes very close to that of the target process. The stochastic part of "0" or "1" comes at every third period. The symbol sequence at 20000 steps, shown in (d), seems quite similar.

In our experiment, learning proceeded with the evolution of the internal dynamics from a fixed point, through limit cycling, and finally to chaos. A similar evolutionary process was observed in the study (Sato, Murakami and Joe, 1990) of the direct learning of chaos (an RNN is trained with a real number time series, generated by a Lorentz attractor, to learn its dynamics.) This type of evolution, from order to disorder, seems essential to learning chaotic dynamics.

4.2 Internal representation

In this section, we focus on the internal dynamics obtained at the end of learning, and investigate how the target SFSM is represented internally by means of evolved attractor dynamics.

Since the RNN has three hidden state units, the dimensionality of the internal state space is three. We graphically examined how the internal state evolves at each time step in this state space (the connective weights are fixed with that obtained in the end of learning). Figure 7 (a) shows the trajectory.

– Fig. 7 –

The transitions from current time to next time are indicated by connected lines, shown as a 2-D projection of the original 3-D internal state space. It appears that the trajectory goes around three regions periodically. This is not, however, limit cycling since each region has a certain width. (If it were limit cycling having a period of three, the trajectory would simply be a triangle connecting three points.)

To clarify this point, we closely examined the structure of this attractor. Figure 7 (b) shows the projection of the obtained attractor into two-dimensional space, which reveals the existence of three segmented curved lines. This implies the interesting fact that the dynamics of attractor is self-organized as a cycle of three one-dimensional maps. We tag the curve segments which are mapped onto each other by $R1$, $R2$, $R3$, $R3$ and parameterize each curve segment by the interval [0,1] in the direction given by the arrows in the figure. (Note that $R2$ is a hairpin curve with no crossing.) We investigated how each region is mapped to another, and also how it is mapped to the output.

Figure 8 (a) shows the obtained mapping from one region to the other. (The length of each region is unified as 1.0.)

– Fig. 8 –

$R1$ is mapped to $R2$, $R2$ is mapped to $R3$ and $R3$ is mapped back to $R1$. An important point to note is that mapping from $R2$ to $R3$ is surjection while all others are bijection. (b) shows the mapping from each region to the output. $R1$ is mapped throughout to 1.0, $R2$ does to 0.0, and $R3$ does continuously from 0.0 to 1.0.

Now, we can explain the mechanism by which the target grammar is embedded into the internal dynamics. Figure 9 shows the illustrative schematics relating the internal dynamics and the target SFSM.

– Fig. 9 –

It can be clearly seen that $R1$ corresponds to state 1 of the target SFSM, $R2$ corresponds to that of state 2 and $R3$ to that of state 3. So, how can we explain the stochasticity assumed in state 3 ? The answer lies in the mapping by surjection from $R2$ to $R3$, in which $R2$ is stretched, folded in the middle then mapped to $R3$. It is well known that the infinite repetition of stretching and folding ultimately generates chaos (explained in any introductory text on dynamical systems, such as (Wiggins, 1990)). The time evolution of the internal state becomes unpredictable after a certain time frame in terms of initial sensitivity. Therefore, the output of "0" or "1", which depends solely on the internal state value on $R3$, becomes stochastic.

5 Summary

We presented a hypothetical model that explains how a biological agent learns target models by observation. We assumed that an agent observes a target process as a stochastic symbolic process, and that it would reconstruct the observed process on its internal deterministic neural dynamics by adjusting its weight parameter set. We assumed RNN for this adjustable deterministic dynamical system, and investigated its learning by experiment. The result revealed the capability of RNN to evolve, by means of learning, toward chaos which is able to mimic a target's stochastic process. Analysis of the internal representation clarified how the grammar, constraining the target process, was embedded into the evolved deterministic chaos.

The study left us further problems to consider. Our experiment exhibited slow convergence of the learning process. Repeated simulations with the same learning parameter, but with different initial weight conditions, showed the probability of convergence to be about half. It was, however, observed that quick learning with a larger β in (10) results in a worse convergence probability. The learning process becomes unstable, accompanied by bursts in the mean square error in such learning. Although it is not yet clear whether this slow learning is essential to a successful convergence, Sato's research (Sato, Murakami, and Joe, 1990) into direct learning of chaos also revealed slow convergence of the same order. A formal analysis of learning chaos is expected to reveal a qualitative differences from learning other dynamics such as fixed point or limit cycling.

An experiment with other simple SFSMs exhibited similar evolution toward chaos in successful learning. However it was observed that various internal representations, having different dimensionality, were obtained according to the learning target as well as the initial weight conditions. The clarification of the correspondence between the type of grammar to be learned and topology of attractor to be self-organized is an open problem.

We have not conducted any experiments to learn upper classes of grammar (i.e. equivalent to infinite state machine). Recent researches on "computation on real numbers" (Crutchfield, 1989; Friedman, 1991) infer that neural function corresponding to such grammar could be realized only in a critical region on a neural weight manifold. Our observation of the onset of chaos in Figure 6 (b) might be an example. To solve this problem, we need to study relationship between neural informatics and complex dynamics in a more qualitative manner.

Appendix

A Initial weights

The initial connective weights and biases are:

$$w_{ij}^h = \begin{pmatrix} -0.345557 & 0.896520 & 0.537531 \\ -0.876996 & -0.466396 & 0.173198 \\ 0.406974 & -0.520104 & -0.318611 \\ -0.512321 & -0.728994 & -0.544173 \end{pmatrix}$$

$$b_i^h = (-0.540455 \quad 0.437992 \quad -0.926643 \quad 0.116201)$$

$$w_{ij}^s = \begin{pmatrix} -0.496082 & -0.683896 & 0.232361 & 0.635133 \\ -0.179600 & -0.138874 & -0.432633 & 0.026872 \\ -0.893048 & -0.660220 & 0.037103 & -0.142628 \end{pmatrix}$$

$$b_i^s = (0.298294 \quad 0.563765 \quad -0.546093)$$

$$w_{ij}^o = (-0.048378 \quad 0.978033 \quad 0.083583 \quad -0.113919)$$

$$b_i^o = (-0.621886)$$

Reference

Bai-lin Hao (1989) Elementary Symbolic Dynamic and Chaos in Dissipative System. World Scientific, Singapore

Crutchfield JP, Young K (1989) Inferring statistical complexity. Phys Rev Lett 63:105-108

Crutchfield JP (1991) Semantics and thermodynamics. In: Casdagli M, Eubank S (eds) Nonlinear Modeling. Addison-Wesley, Reading, Massachusetts, pp1-36

Elman J (1990). Finding structure in time. Cognitive Science 14:179-211

Friedman EJ (1991) Structure and uncomputability in one-dimensional maps. Complex Systems 5:335-350

Giles CL, Miller CB, Chen D, Chen HH, Sun GZ, Lee YC (1992) Learning and extracting finite state automata with second order recurrent neural networks. Neural Computation 4:393-405

Harnad S (1990) The symbol grounding problem. Physica D 42:335-346

Jordan M (1986) Attractor dynamics and parallelism in a connectionist sequence machine. In: Proc of Ninth Annual conference of Cognitive Science Society, Lawrence Erlbaum, pp.531-546

Norman DA (Ed.) (1980) Perspectives on cognitive science. Ablex Publishing Corp., Norwood, N.J.

Pineda FJ (1987) Generalization of back-propagation to recurrent neural networks. Phys Rev Lett 59:2229-2232

Pollack JB (1991) Higher order recurrent networks and grammatical inference. Machine Learning 7:227-252

Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation. In: D.E. Rumelhart, and J. L. McLelland (Eds.) Parallel Distributed Processing. MIT Press, Cambridge, MA, pp.318-362

Sato M, Murakami Y, Joe K (1990) Learning chaotic dynamics by recurrent neural networks. In: Proc. of Int. Conf. on Fuzzy Logic and Neural Networks, Iizuka, Japan, pp.601-604

Sato M (1990) A learning algorithm to teach spatiotemporal patterns to recurrent neural networks. Biol Cybern 62:259-263

Tani J, Fukumura N (in press) Embedding a grammatical description in deterministic chaos: an experiment in recurrent neural learning. Biol Cybern

Varela FJ, Maturana R, Uribe R (1974) Autopoiesis: the organization of living systems, its characterization and a model. BioSystems 5:187-196

Wiggins S (1990) Introduction to Applied Nonlinear Dynamical Systems and Chaos. Springer, Berlin Heidelberg New York, pp.420-438

List of Figures

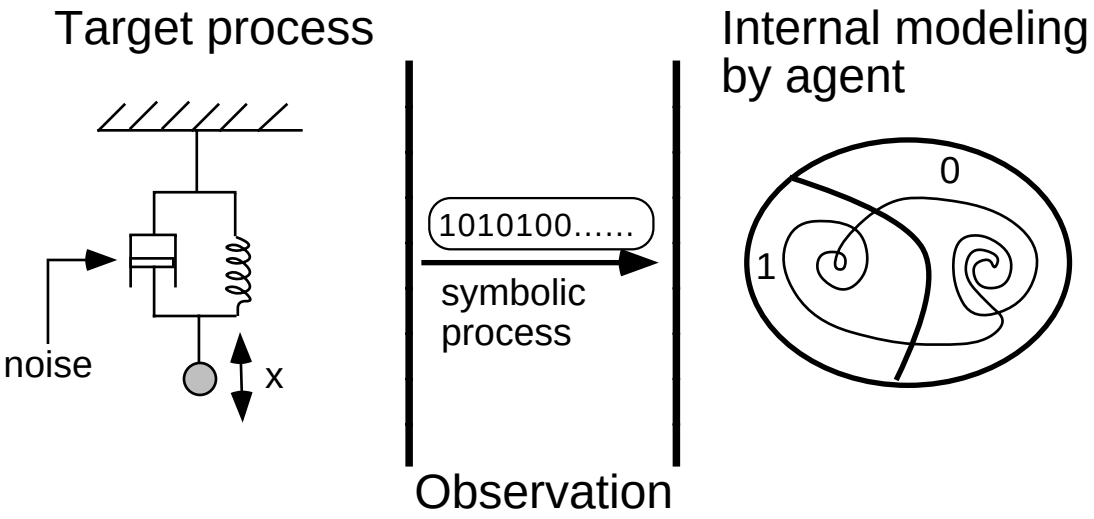

Figure 1: Observation and modeling of target process.

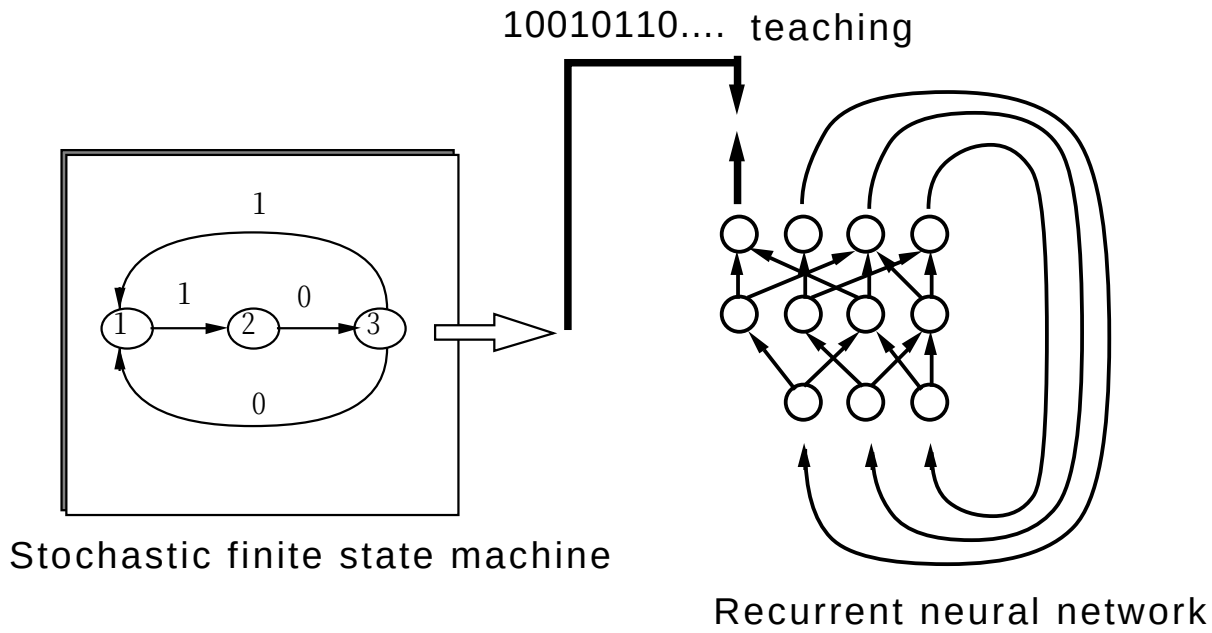

Figure 2: RNN reconstructs SFSM from observed stochastic symbolic sequence.

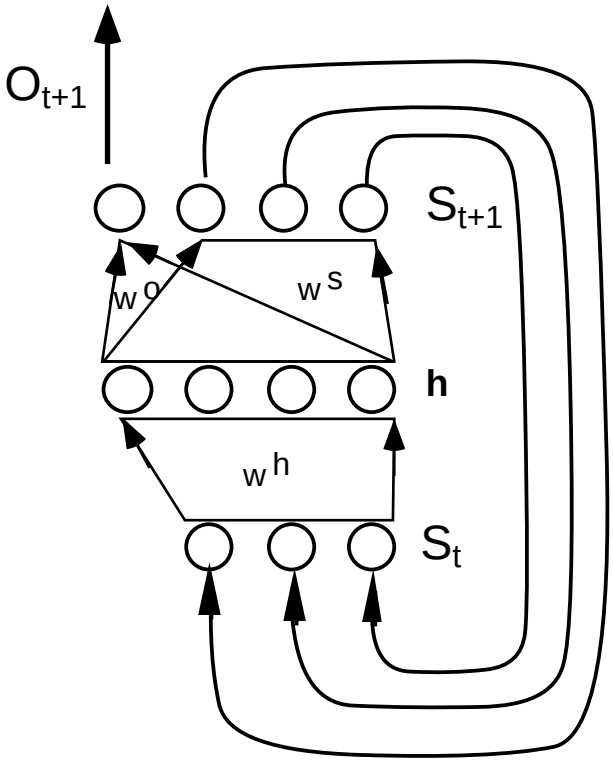

Figure 3: Structure of employed RNN.

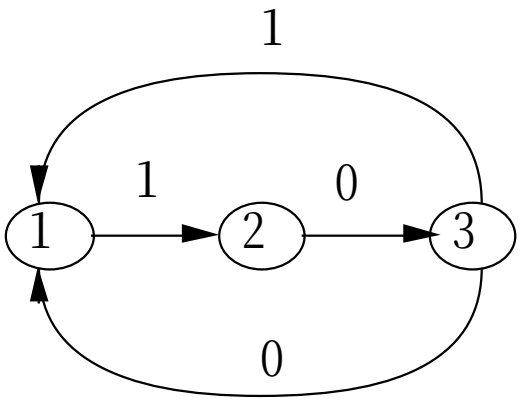

Figure 4: Target SFSM of learning.

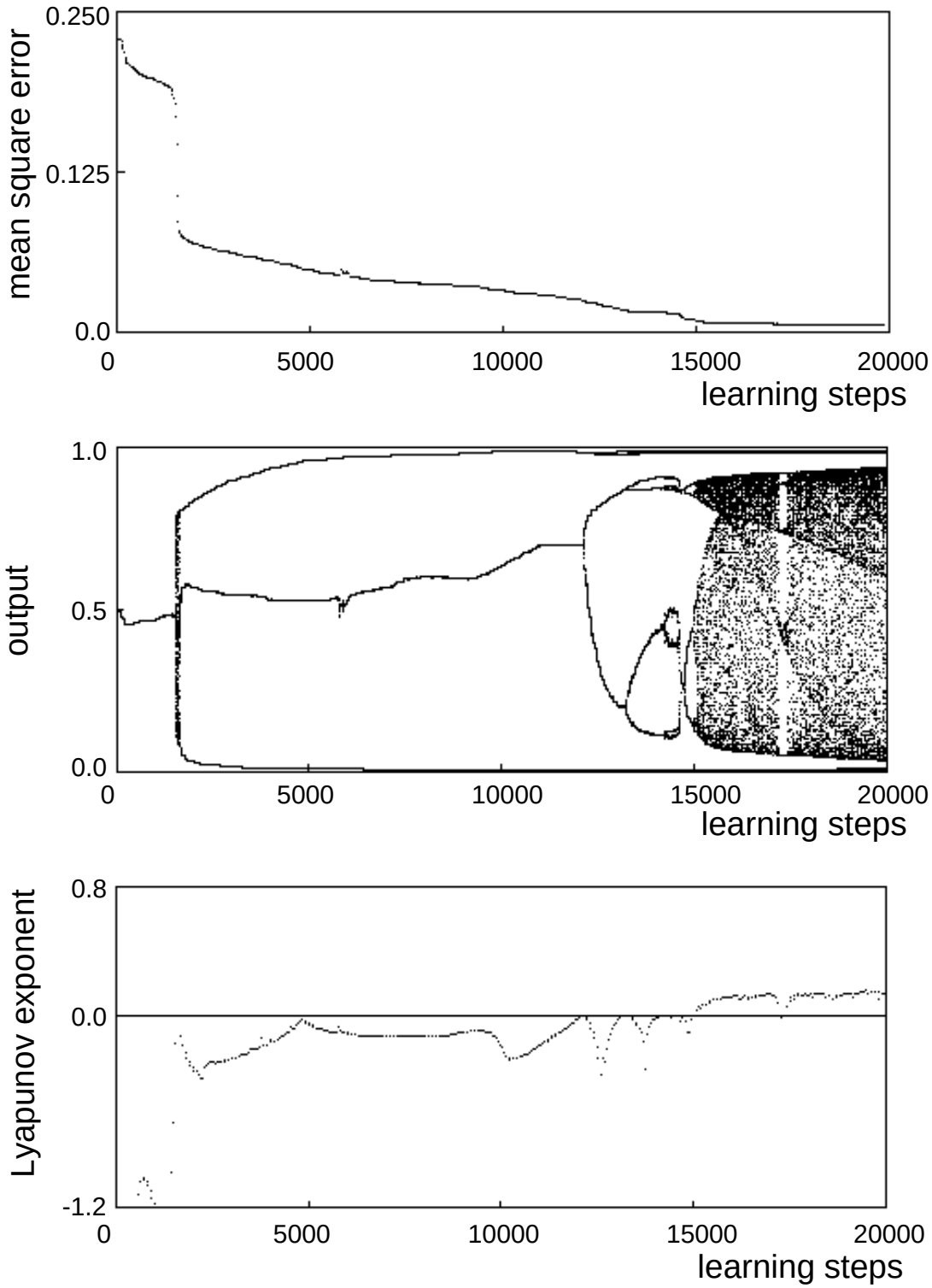

Figure 5: Evolutionary learning process, uppermost showing time history of the learning errors, the middle the output trace of internal dynamics, and the lowermost the Lyapunov exponent.

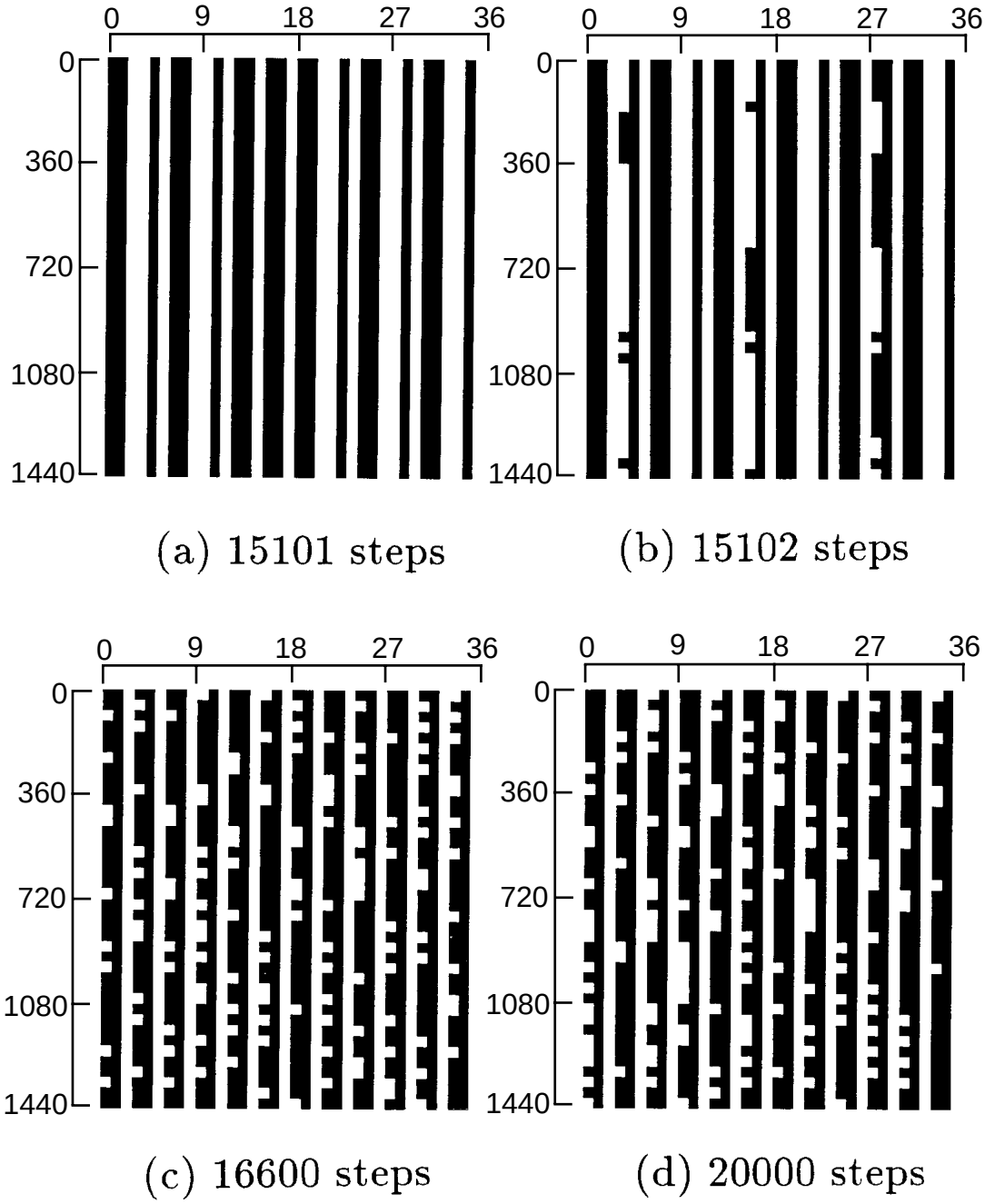

Figure 6: Symbol sequences generated by internal dynamics at each stage of the evolving process.

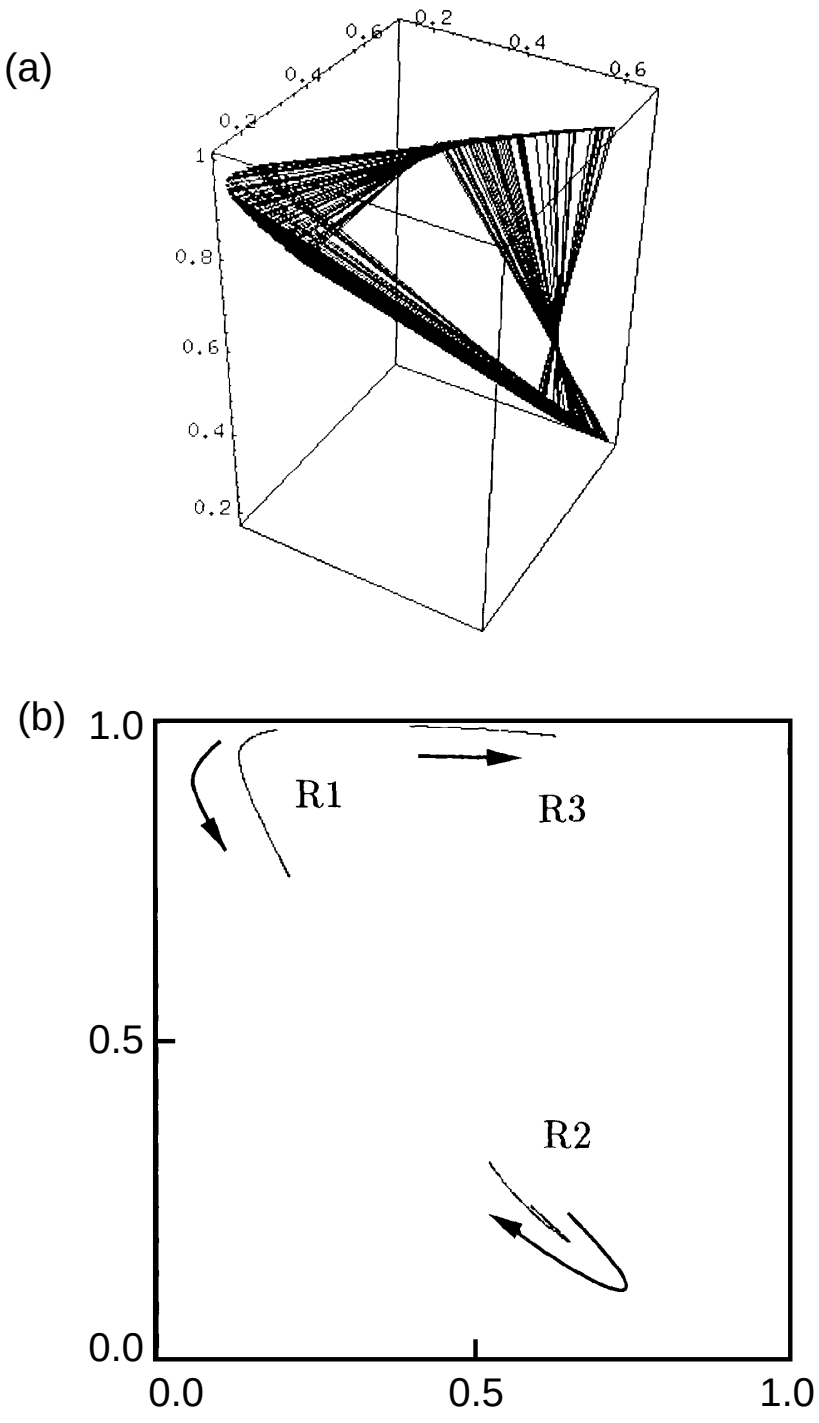

Figure 7: (a) shows trajectory of internal dynamics obtained at the end of learning, (b) Emergent attractor consisting of three regions of curved lines, indexed as R1, R2 and R3. For each region direction is defined by the attached arrow.

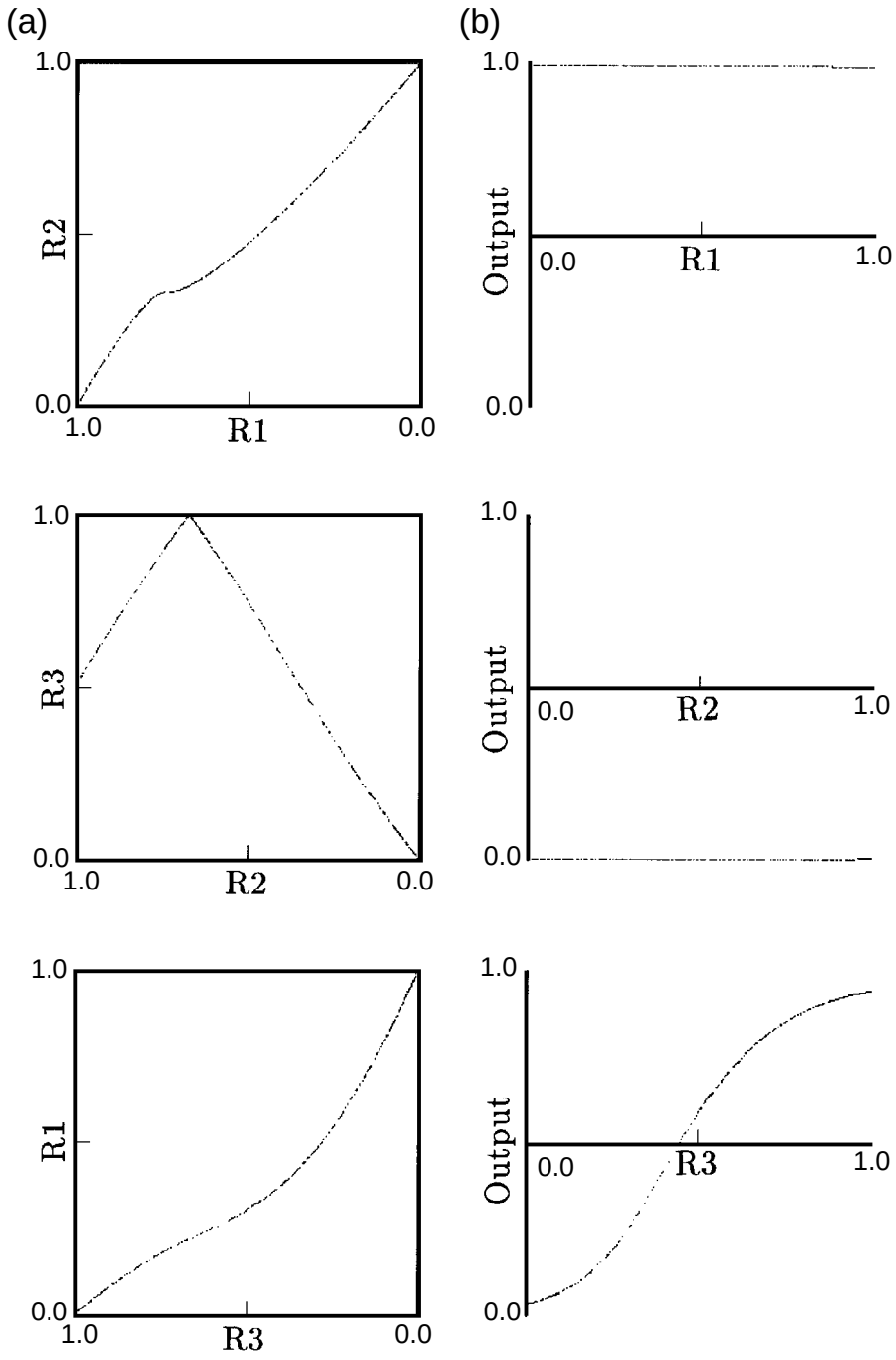

Figure 8: (a) shows internal mapping from one region to another, while (b) shows mapping from one region to the output.

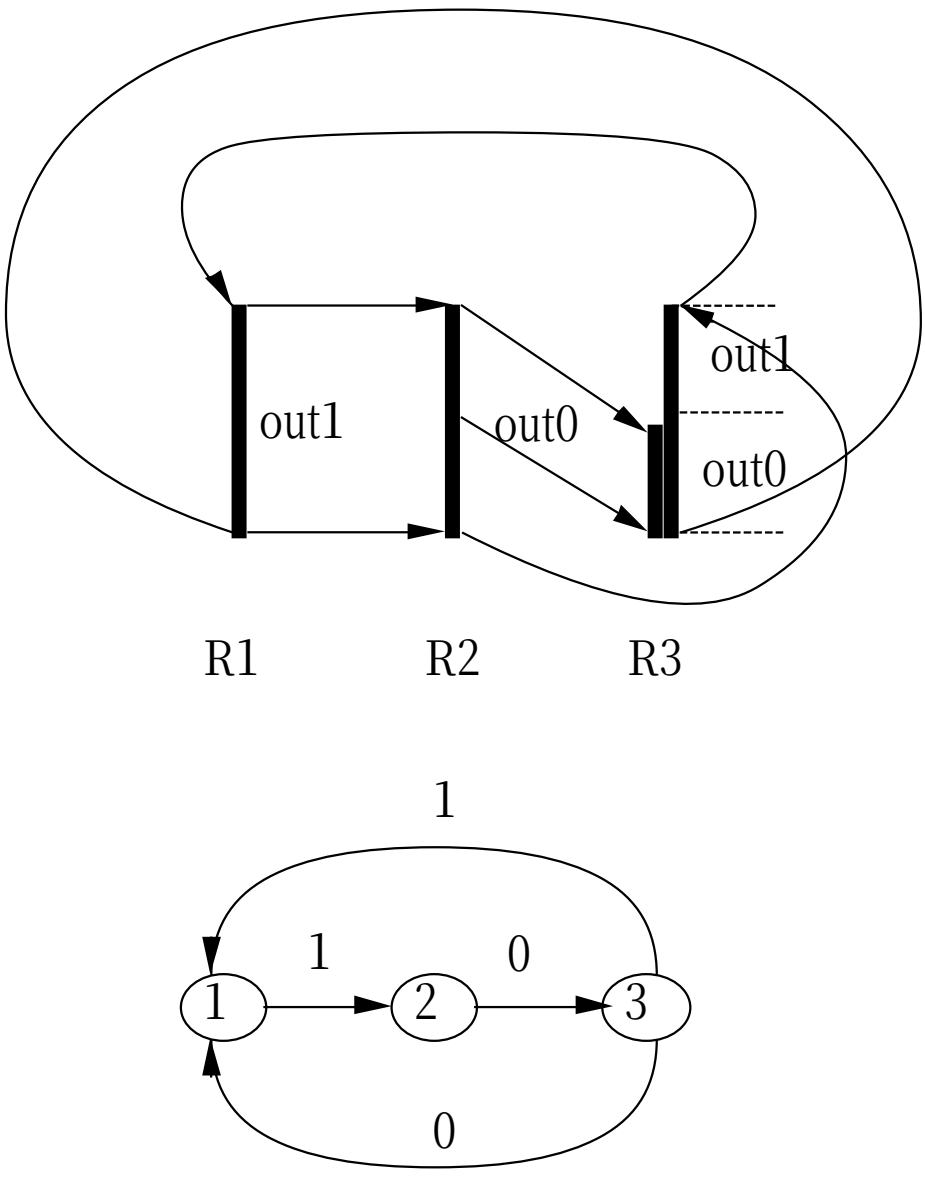

Figure 9: Schematics relating the evolved internal dynamics and target SFSM.