Chaos-based Reinforcement Learning when Introducing Refractoriness in Each Neuron

Katsuki Sato, Yuki Goto, and Katsunari Shibata

Oita University, 700 Dannoharu, Oita, Japan
{shibata}@oita-u.ac.jp

Abstract. Aiming for the emergence of "thinking", we have proposed new reinforcement learning using a chaotic neural network. Then we have set up a hypothesis that the internal chaotic dynamics would grow up into "thinking" through learning. In our previous works, strong recurrent connection weights generate internal chaotic dynamics. On the other hand, chaotic dynamics are often generated by introducing refractoriness in each neuron. Refractoriness is the property that a firing neuron becomes insensitive for a while and observed in biological neurons. In this paper, in the chaos-based reinforcement learning, refractoriness is introduced in each neuron. It is shown that the network can learn a simple goal-reaching task through our new chaos-based reinforcement learning. It can learn with smaller recurrent connection weights than the case without refractoriness. By introducing refractoriness, the agent behavior becomes more exploratory and Lyapunov exponent becomes larger with the same recurrent weight range.

Keywords: Reinforcement Learning, Chaotic Neural Network, Goal Reaching, Refractoriness

1 Introduction

Our group has proposed the end-to-end reinforcement learning approach in which the entire process from input sensors to output motors that consists of a neural network is trained through reinforcement learning, and then various functions emerge in it[1]. DeepMind group showed the successful learning result of TV games in this approach[2], and that has consolidated the effectiveness of the approach. The higher functions that we human have, for example, memory, prediction, logical thinking need time-series processing. Our group has used a recurrent neural network that can learn to deal with time-series data, and has shown the emergence of memory or prediction function through reinforcement learning[3][4]. However, what we can call logical thinking, which is a typical higher function has not emerged yet.

We can think one after another without any input from outside, and so logical thinking can be thought of as internal dynamics that transit among states autonomously. Exploration, which is essential in reinforcement learning, is similar to thinking in terms of dynamics with autonomous state transitions. From this

similarity between logical thinking and exploration, we have set up a hypothesis that exploration, which is generated as chaotic dynamics, grows up into logical thinking through learning. So, we have proposed new reinforcement learning using a chaotic neural network (ChNN). Here, an agent explores according to its internal chaotic dynamics without adding random noises from outside, and can learn a simple goal-reaching task or an obstacle avoidance task[5][6]. In our previous works, strong recurrent connection weights generate internal chaotic dynamics in a recurrent neural network.

On the other hand, it is often the case that chaotic dynamics are generated by introducing refractoriness in each neuron[8]. Refractoriness is the property that neurons that have fired do not fire for a while, and is also the property that biological neurons actually have. Chaotic itineracy, which we think very important property for inspiration or discovery, can be observed when associative memory is implemented. It is also shown that there is a difference in degree of chaos between known and unknown patterns on an associative memory using a ChNN, and after an unknown pattern is learned, association to the pattern is formed as well as the other known patterns[7].

In this paper, we introduce refractoriness in each neuron, and apply our new reinforcement learning to the refractoriness-originated chaotic neural network (RChNN). We examine whether the RChNN can learn a simple goal-reaching task. We compare the learning results between the cases of introducing refractoriness and without refractoriness, and observe Lyapunov exponent for both cases varying the range of the recurrent connection weights.

2 Reinforcement Learning (RL) using a Refractoriness-originated Chaotic Neural Network (RChNN)

In RL, an agent learns actions autonomously to get more a reward and less punishment. To realize autonomous learning, exploration is necessary, and in general, an agent explores stochastically using external random noises from outside. However here, an agent explores according to its internal chaotic dynamics in its ChNN without adding external random noises. For the learning of motor-level continuous motion signals, actor-critic is used. The actor-net, which generates actions, is made up of a ChNN, and the critic-net, which generates state value, is made up of a non-chaotic layered NN. The chaotic neuron model with refractoriness used in the actor net is dynamic as

$$u_{j,t}^{a,\xi} = \left(1 - \frac{\Delta t}{\tau}\right) u_{j,t-\Delta t}^{a,\xi} + \frac{\Delta t}{\tau} \sum_{i=1}^{N^{in}} w_{j,i}^{a,h} \cdot o_{i,t}^{a,in} \tag{1}$$

$$u_{j,t}^{a,\eta} = \left(1 - \frac{\Delta t}{\tau}\right) u_{j,t-\Delta t}^{a,\eta} + \frac{\Delta t}{\tau} \sum_{i=1}^{N^{h}} w_{j,i}^{a,REC} \cdot o_{i,t-\Delta t}^{a,h} \tag{2}$$

$$u_{j,t}^{a,\zeta} = \left(1 - \frac{\Delta t}{\kappa\tau}\right) u_{j,t-\Delta t}^{a,\zeta} - \frac{\Delta t}{\kappa\tau} \cdot \alpha \cdot o_{j,t-\Delta t}^{a,h} \tag{3}$$

$$u_{j,t}^{a,h} = u_{j,t}^{a,\xi} + u_{j,t}^{a,\eta} + u_{j,t}^{a,\zeta} \tag{4}$$

$$o_{j,t}^{a,h} = \frac{1}{1+\exp(-g \cdot u_{j,t}^{a,h})}. \tag{5}$$

The model is originated from the Aihara model[8], but we use the expression where the time constants are explicitly written. Each of Eq. (1)(2)(3) shows forward, recurrent, or refractoriness term that appears in Eq. (4) respectively. $u_{j,t}^{a,h}$ and $o_{j,t}^{a,h}$ are the internal state and the output of the j-th hidden neuron at time t. $o_{i,t}^{a,in}$ is the i-th input signal. a indicates the actor-net, $h(=1)$ and $in(=0)$ indicate the hidden and input layer respectively. $w_{j,i}^{a,h}$ is the connection weight from the i-th neuron in the input layer to the j-th neuron in the hidden layer, and $w_{j,i}^{REC}$ is the recurrent connection weight from the i-th to the j-th neuron in the hidden layer. All the weights are decided by uniform random numbers. Here, we use step size $\Delta t = 1$, time constant $\tau = 1.25$, scaling parameter for time constant $\kappa = 8$ referring to [8], α is the scaling parameter of refractoriness, and g is the gain of sigmoid function. Here we use $\alpha = 3$ and $g = 2$. The neuron model used in the output layer($L = 2$) in the actor-net or each neuron in the critic-net is static as

$$u_{j,t}^{n,l} = \sum_{i=1}^{N^{(l-1)}} w_{j,i}^{n,l} \cdot o_{i,t}^{n,l-1} \tag{6}$$

$$o_{j,t}^{n,l} = \frac{1}{1+\exp(-u_{j,t}^{n,l})} - 0.5. \tag{7}$$

$n = a$ or c, and a represents the actor-net and c represents critic-net respectively. TD-error $\hat{r}_t$ used for learning is computed as

$$\hat{r}_t = r_{t+\Delta t} + \gamma \cdot V_{t+\Delta t} - V_t \tag{8}$$

where $r_{t+\Delta t}$ is the reward given at time $t + \Delta t$, γ is a discount factor, and here 0.96 is used. $V_{t+\Delta t} = O_{t+\Delta t}^{c,L}$ is the critic output, and $L(=2)$ represents the output layer. The training signal T_{V_t} for the output in the critic-net at time t is computed as

$$T_{V_t} = r_{t+\Delta t} + \gamma \cdot V_{t+\Delta t}. \tag{9}$$

The critic NN is trained once by regular error backpropagation using T_{V_t}.

In the proposed method, there is no external random number added to the actor outputs. The weights $w_{j,i}^{a,l}$ $(l = L(2), h(1))$ in the RChNN are modified using the causality trace $c_{j,i,t}^{l}$[5] and a learning rate η as

$$\Delta w_{j,i,t}^{a,l} = \eta \cdot \hat{r}_t \cdot c_{j,i,t}^{l} \tag{10}$$

where $\Delta w_{j,i,t}^{a,l}$ is the update of the weight $w_{j,i}^{a,l}$. The trace $c_{j,i,t}^{l}$ is put on each connection, and takes and maintains the input through the connection according

to the change in its output $\Delta o^l_{j,t} = o^l_{j,t} - o^l_{j,t-1}$ as

$$c^l_{j,i,t} = (1 - |\Delta o^{a,l}_{j,t}|)c^l_{j,i,t-\Delta t} + \Delta o^{a,l}_{j,t} o^{a,l-1}_{i,t}. \tag{11}$$

Here, only the connection weight $w^{a,l}_{j,i}$ from inputs to hidden neurons or from hidden neurons to output neurons are trained, and the recurrent connection weight $w^{a,REC}_{j,i}$ is not trained.

3 Simulation

In this paper, in order to examine whether our new reinforcement learning works also in an RChNN, we set a simple goal-reaching task as shown in Fig. 1.

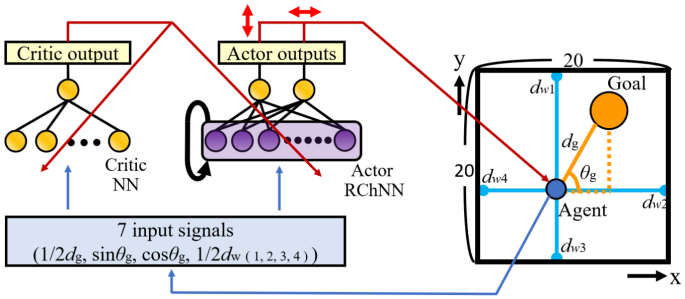


Fig. 1. Chaos-based reinforcement learning system in an agent and a goal-reaching task

Table 1. Parameters used in the simulation

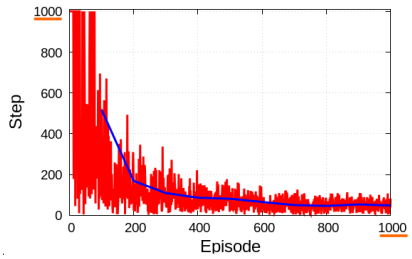
		Actor	Critic
Number of hidden neurons		100	10
Gain of sigmoid function: g	Output	1	
	Hidden	2	1
Learning rate: η	Output	0.01	1
	Hidden(FW)	0.01	1
	Hidden(REC)	0	-
Range of initial weights	Hidden(FW)	[-1,1]	
	Hidden(REC)	varied	-
	Output	[-1,1]	

In this simulation, as shown in Fig. 1, there is a 20×20 field. An agent with a radius of 0.5 and a goal with a radius of 1.0 are located randomly at the beginning of each episode. The agent catches 7 input signals representing the distance and direction from the agent to the goal and the distance to each wall, and inputs them to each neural network as in Fig. 1. Each of the two actor outputs represents the agent's movement in one step in the x-direction or the y-direction respectively, but they are normalized without changing its moving direction so that the movable range becomes a circle with a radius of 0.5. When

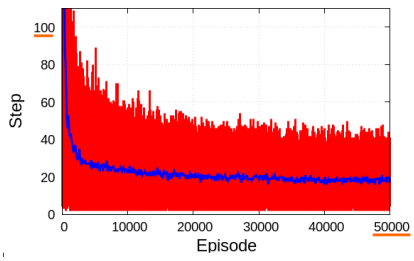
the agent reaches the goal, it gains 0.4 reward. When the agent hits the wall, the agent is given a penalty of -0.01. One episode finishes when the agent reaches the goal or reaches 1,000 steps that is the upper limit of the step. The agent learned 50,000 episodes in total.

Fig. 2 shows the learning curve. In order to see the early stage of learning, the horizontal scale is expanded in (a), while in order to see the late stage of learning the vertical scale is expanded in (b). Both red and blue lines show the number of steps to reach the goal but the red one is plotted at each episode while the average value over each 100 episodes is plotted as the blue one. Fig. 3 (a) shows sample trajectories after learning for 8 patterns when the goal was located at the center. Fig.3 (b) shows the changes in the critic value for 8 trajectories in Fig. 3 (a).

As the number of episodes increases, the number of steps to the goal decreases. The agent after learning moves toward the goal, and the critic becomes higher as the agent approaches the goal not depending on the goal position. We can see that an agent having an RChNN can learn a simple goal-reaching task with new reinforcement learning.

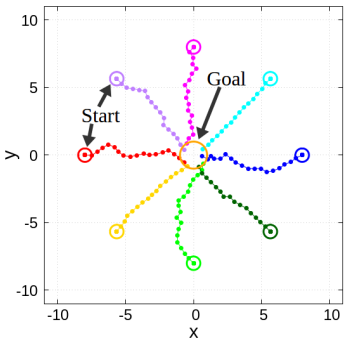


(a) Learning curve
(horizontal scale is expanded)

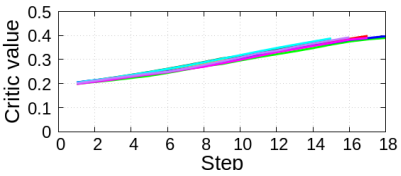


(b) Learning curve
(vertical axis is expanded)

Fig. 2. Learning curve



(a) Trajectories



(b) Critic value change

Fig. 3. 8 sample trajectories after learning and the critic value change for each trajectory

Next, we compare the learning success rate between an RChNN and a regular ChNN without refractoriness (the refractoriness term in equation (3) is removed). The range$[-w_{max}^{REC}, w_{max}^{REC}]$ of the recurrent connection weights w_{max}^{REC} was changed from 0.3 to 2, and the number of successful learning runs in 20 runs for each different weight range is compared as shown in Fig. 4. Regardless of having refractoriness, the number of unsuccessful runs increases as the recurrent connection weight decreases, but the success rate decreases faster in the case of without refractoriness.

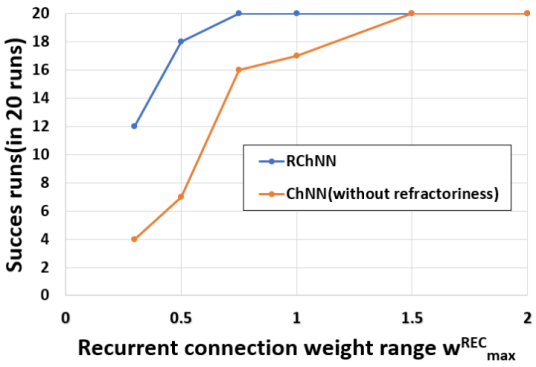
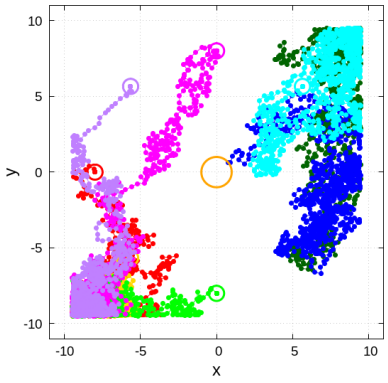
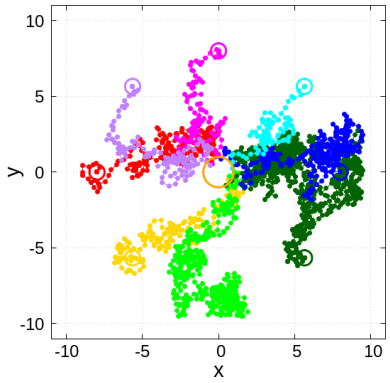


Fig. 4. Comparison of the learning success rate for the various range of recurrent connection weights in the hidden layer between the cases with and without refractoriness.

The agent behaviors before learning and after 100 episodes of learning for both cases when the weights range of the recurrent connection weights w_{max}^{REC} is 2.0 are shown in Fig. 5. We can see that the agent behavior is more exploratory with refractoriness than in the case without refractoriness. For the case without refractoriness, the agent moves toward the wall before learning (b-1). However, the agent gets punished by crashing, and after 100 episodes, the agent becomes more exploratory. In Fig. 4, when $w_{max}^{REC} = 2$, the agent succeeded in all the 20 runs finally in both cases.

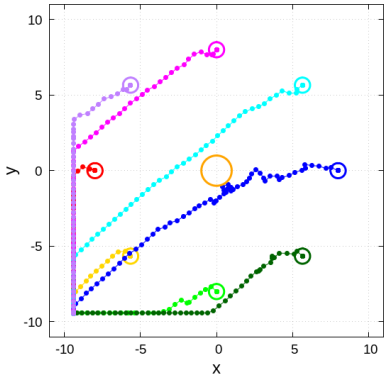


(a-1) Before learning

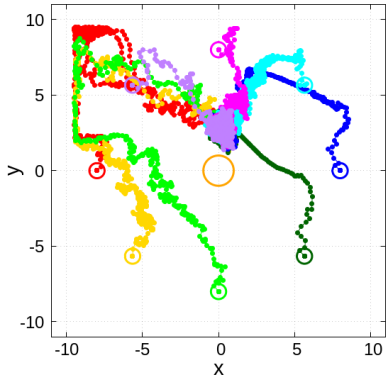


(a-2) After 100 episodes of learning

(a) With refractoriness



(b-1) Before learning



(b-2) After 100 episodes of learning

(b) Without refractoriness

Fig. 5. Comparison of the agent's exploration at the beginning of learning. ($w_{max}^{REC} = 2$)

In order to observe the relationship between degree of chaos and learning success rate, we calculate Lyapunov exponent. Lyapunov exponent is an index for degree of chaos in a dynamic system. If Lyapunov exponent is positive, the system is chaotic. Here, the network state is updated for 50 steps using only the recurrent connections, and is compared between the cases when a small perturbation whose size is 0.001 is added or not to initial state. Then Lyapunov exponent λ is calculated for 20 networks with different weights as

$$\lambda = \frac{1}{50 \cdot 20} \sum_{p=1}^{20} \sum_{t=1}^{50} \ln \frac{d_{p,t+\Delta t}}{d_{p,t}} \tag{12}$$

d is the distance in hidden state between the cases when a perturbation is added or not.

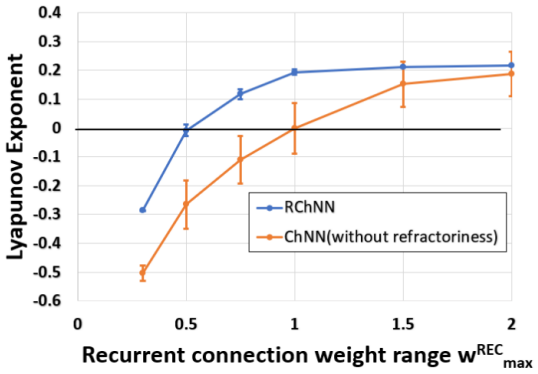


Fig. 6. Comparison of the Lyapunov exponent between the cases with or without refractoriness for the various ranges of the recurrent connection weights in the hidden layer.

In, Fig. 6, as the recurrent connection weights decreases, the degree of chaos decreases. The dynamics is more chaotic in the case with refractoriness than without refractoriness. The degree of chaos becomes stronger by introducing refractoriness. From the similarity of the trend between Fig. 4 and Fig. 6, introduction of refractoriness makes the degree of chaos strong and largely influences to the learning performance. We think that in the case with refractoriness, more exploratory behavior based on high degree of chaos is the source of the high successful learning rate for small recurrent connection weight ranges. Lyapunov exponent increases by using symmetrical activation function, but here we use Eq. (5).

4 Conclusion

In this paper, we used a refractoriness-originated chaotic neural network in our new reinforcement learning, and showed that the network can learn a simple goal-reaching task. As the recurrent connection weights decrease, the success rate decreases more slowly than the case without refractoriness, and that is very similar to the change in Lyapunov exponent. From the observation of the agent behavior at an early stage of learning, it is known that the introduction of refractoriness makes the degree of chaos strong and leads more exploratory behavior. That would be the reason why the success rate is larger in the case with refractoriness for the same range of the recurrent connection weights.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 15K00360

References

1. K. Shibata and Y. Goto : New Reinforcement Learning Using a Chaotic Neural Network for Emergence of "Thinking"-"Exploration" Grows into "Thinking" through Learning -, arXiv : 1705.05551 (2017)
2. Volodymyr, M., et al.: Playing Atari with Deep Reinforcement Learning, NIPS Deep Learning Workshop 2013 (2013)
3. K. Shibata and H. Utsunomiya : Discovery of Pattern Meaning from Delayed Rewards by Reinforcement Learning with a Recurrent Neural Network, Proc. of IJCNN. 2011, pp. 1445-1452,(2011)
4. K. Shibata and K. Goto : Emergence of Flexible Prediction-Based Discrete Decision Making and Continuous Motion Generation through Actor-Q-Learning, Proc. of ICDL-Epirob.ID 15 (2013)
5. K. Shibata and Y. Sakashita : Reinforcement Learning with Internal-Dynamics-based Exploration Using a Chaotic Neural Network, Proc. of IJCNN. (2015)
6. Y. Goto and K. Shibata : Emergence of Higher Exploration in Reinforcement Learning Using a Chaotic Neural Network, Proc. of ICONIP 2016, pp. 40-48, (2016)
7. Y. Osana and M. Hagiwara : Successive Learning in Chaotic Neural Network, Proc. of IJCNN 1998, Vol. 2, pp. 1510-1515 (1998)
8. Aihara, K., Takabe, T., Toyoda, M.: Chaotic neural networks, Physics Letters A, **144**(6-7), pp. 333-340 (1990)