Reward-Based Learning of a Memory-Required Task based on the Internal Dynamics of a Chaotic Neural Network

Toshitaka Matsuki and Katsunari Shibata

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

Abstract. We have expected that dynamic higher functions such as "thinking" emerge through the growth from exploration in the framework of reinforcement learning(RL) using a chaotic Neural Network(NN). In this frame, the chaotic internal dynamics is used for exploration and that eliminates the necessity of giving external exploration noises. A special RL method for this framework has been proposed in which "traces" were introduced. On the other hand, reservoir computing has shown its excellent ability in learning dynamic patterns. Hoerzer et al. showed that the learning can be done by giving rewards and exploration noises instead of explicit teacher signals. In this paper, aiming to introduce the learning ability into our new RL framework, it was shown that the memory-required task in the work of Hoerzer et al. could be learned without giving exploration noises by utilizing the chaotic internal dynamics while the exploration level was adjusted flexibly and autonomously. The task could be learned also using "traces", but still with problems.

Keywords: Chaotic Neural Network, Reservoir Computing, Reward-Modulated Hebbian Learning, Traces, Dynamic Higher Functions

1 Introduction

In recent years, Deep Learning, in which a large-scale neural network(NN) with many layers learns to process raw sensor signals in parallel, has surpassed existing systems in various fields. That suggests the difficulty in understanding the phenomenal performance of our parallel brain through our sequential consciousness and then developing an appropriate program by hand for such massively parallel processing. For a long time, our group has pointed out this difficulty and has suggested the necessity to develop a system in which the whole process from sensors to motors consists of a NN and necessary functions or useful internal representations emerge through reinforcement learning(RL) with explorations and rewards[1][2]. Recently, a recurrent NN (RNN) has been employed to deal with dynamics, and it was confirmed that the function of "memory" or "prediction" emerges in a simple task[3][4]. However, there seems to be a limitation for a non-chaotic "silent" RNN to form multi-stage state transitions through learning[5].

Thus, we thought that the complex dynamics is not formed from scratch in a non-chaotic "silent" RNN but is reformed from rich and chaotic internal dynamics in a chaotic NN. The dynamics is used also for exploration in RL, and that eliminates the need for giving exploration noises from outside. It is expected to become purposeful through learning reflecting the causal relations of the world and finally reach dynamic higher functions such as "thinking". In this new RL framework, since exploration components cannot be separated from the outputs, training signals cannot be derived. Therefore, instead of using error back propagation as in the conventional RL, a special learning method, in which "traces" are introduced, was proposed and confirmed to work in an easy task[6][7].

On the other hand, recently, reservoir computing such as Echo State Network[8] and Liquid State Machine[9] has been focused on. In this trends, Sussillo et al. trained reservoir networks by the new learning procedure called FORCE Learning[10]. In this procedure, the outputs are returned to the network along its feedback pathway and only readout weights are modified to match the network outputs with target patterns. Then, the network can learn to generate complex dynamic patterns amazingly easily and rapidly.

Hoerzer et al. showed that a reservoir network can learn through Reward-Modulated Hebbian Learning in which instead of explicit teacher signals, exploration noises and the reward to show the improvement of the performance derived from the error between outputs and targets were given[11]. In this research, it was shown that the network learned various dynamic patterns or a working memory task.

From the above, we think that it is essential to introduce the learning ability of dynamical patterns into our new RL framework to realize dynamic higher functions such as "thinking". In this paper, as the first step of this attempt, we examine whether the working memory task, which a reservoir network learned from reward signals in Hoerzer's work, can be learned without giving external exploration noises by utilizing the internal chaotic dynamics of the network as well as in our new RL framework. Next, we also examine whether the network can be trained with "traces", which are used in our new RL.

2 Method

2.1 Network

In this paper, we use the network as in Fig.1, which has basically the same structure and connection weights in the previous researches[10][11]. The network is composed of $N = 1000$ neurons, and they are sparsely and recurrently connected (connection probability $p = 0.1$). There are four external inputs each of which is fed to all the neurons. There are two output units called readout units, and are connected by all the network neurons. Each output from the corresponding readout unit is returned to all the network neurons along its feedback pathway. The model of each network neuron is a dynamical firing-rate model. The internal activity(membrane potential) of the j-th network neuron at time t is given as

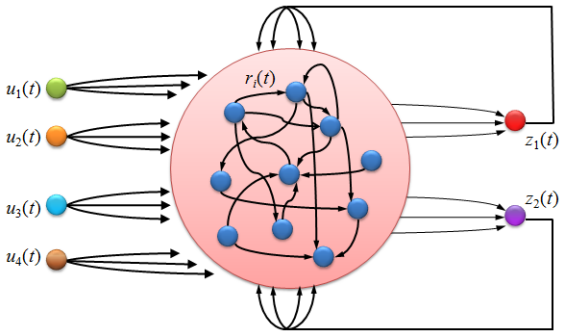


Fig. 1. The network model. It has 4 inputs,u_1(green),u_2(orange),u_3(cyan),u_4(brown) and 2 outputs,z_1(red),z_2(purple). In the network, 1000 neurons(blue) are recurrently connected(connection probability $p = 0.1$).

$$x_j(t) = \Big(1 - \frac{\Delta t}{\tau}\Big)x_j(t - \Delta t) + \frac{\Delta t}{\tau}\Big(\lambda \sum_{i=1}^{N} w_{ji}^{rec} r_i(t) + \sum_{i=1}^{I} w_{ji}^{in} u_i(t) + \sum_{i=1}^{O} w_{ji}^{fb} z_i(t)\Big), \tag{1}$$

where the step size $\Delta t = 1$[ms] and the time constant $\tau = 10$[ms]. λ is the parameter that gives the scale of recurrent connection weights between the network neurons, whose value is 1.8 or 1.5(the latter is used in the training with "traces"). Larger λ produces more chaotic activities of the network. w_{ji}^{rec} is the weight of recurrent connection from the i-th neuron to the j-th neuron. These are set to a value generated randomly from a Gaussian distribution with zero mean and variance $1/pN$. I is the number of inputs. w_{ji}^{in} is the weight from the i-th input to the j-th neuron. u_i is the i-th input value. O is the number of readout units. w_{ji}^{fb} is the weight from i-th readout unit to j-th neuron. These are set to a value generated randomly from a uniform distribution between -1 and 1. $z_i(t)$ is the output of the j-th readout unit. The output of network neuron $r_j(t)$ is computed from its internal activity $x_j(t)$ as

$$r_j(t) = tanh\Big(x_j(t)\Big). \tag{2}$$

$z_j(t)$ at time t is derived from $r_i(t)$ and the corresponding readout weight w_{ji} as

$$z_j(t) = \sum_{i=1}^{N} w_{ji} r_i(t). \tag{3}$$

Initially, w_{ji} is set to a value generated randomly from a Gaussian distribution with zero mean and variance $1/N$.

2.2 Learning

In this paper, only readout weights w_{ji} are trained. Excepting that exploration noises are not added, we basically followed the learning procedure by Hoerzer et al. in[11]. The network is trained with reward or penalty which is given dependently on whether the current performance of the network $P(t)$ is improved as compared to its running average $\overline{P}(t)$ with time constant 5ms. $P(t)$ is defined as

$$P(t) = -\sum_{j=1}^{O}\Big(z_j(t) - f_j(t)\Big)^2, \tag{4}$$

where f_j is the target for the j-th output of the network.

We use two learning methods in this research. First one is Reward-Modulated Hebbian Learning in [11] with a little modification. The modulatory signal $M(t)$ is defined using $P(t)$ and $\overline{P}(t)$ as

$$M(t) = \begin{cases} 1 & P(t) > \overline{P}(t) \\ -1 & P(t) \leq \overline{P}(t). \end{cases} \tag{5}$$

In [11], $M(t)$ took the value of 1 or 0, but here 1 or -1 is used. The readout weights are modified with $M(t)$ as

$$\Delta w_{ji} = \eta\Big(z_j(t) - \overline{z}_j(t)\Big)M(t)r_i(t). \tag{6}$$

where η is a learning constant and here $\eta = 0.0005$. $\overline{z}(t)$ is the running average of z with time constant 5ms.

Second, we use a learning method with traces that are used in our new RL. In this learning, to limit the value range, the output $z(t)$ is derived as

$$z_j(t) = tanh\Big(\sum_{i=1}^{N} w_{ji}r_i(t)\Big). \tag{7}$$

The readout weights are modified with $P(t)$ and $\overline{P}(t)$ as

$$\Delta w_{ji} = \eta\Big(P(t) - \overline{P}(t)\Big)c_{ji}(t), \tag{8}$$

where $c_{ji}(t)$ is the trace which expresses the correlation between the output increase and the i-th input in the j-th readout unit and $\eta = 0.05$ here. $c_{ji}(t)$ is given by

$$\Delta z_j(t) = z_j(t) - z_j(t - \Delta t). \tag{9}$$

$$c_{ji}(t) = \Big(1 - \frac{|\Delta z_j(t)|}{2}\Big)c_{ji}(t - \Delta t) + \frac{\Delta z_j(t)}{2}r_i(t), \tag{10}$$

where 2 is the value range of each readout unit. This equation computes the value similar to the running average of the input, but the time constant is derived from the output change of the unit. Therefore, when the output change is large, the signal $r_i(t)$ is considered to be important and taken into the trace largely. When the output does not change, the past value is kept in the trace.

2.3 Task

The network learns the task that requires working memory[11]. The network has four inputs and two outputs. Input pulses with the average rate of 0.5Hz are given on each input signal independently. It goes up to 1.0 taking 50ms and then goes down with time constant 50ms. Each signal has a different meaning. u_1 and u_2 are respectively ON and OFF signals for the output z_1, and u_3 and u_4 are for the output z_2. An ON or OFF signal makes the corresponding output to be 1.0 or -1.0 respectively with time constant 20ms, and the value is kept until the opposite signal for the corresponding output comes in.

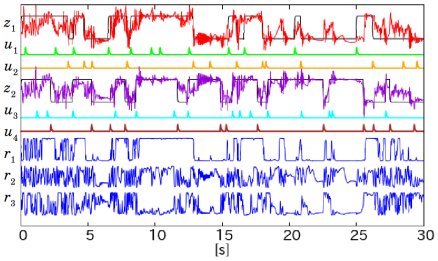
3 Results

Fig.2 shows the network activity: outputs, inputs and activities of some neurons. Fig.2(a) shows the activities for the first 30 seconds of learning. At first, the network did not know its desired behavior. However, noise-like fluctuations originated from the chaotic internal dynamics appeared in the output even without external noises. The chaotic internal dynamics performs the role of exploration, and the outputs looks to follow the targets with a lag. However, when learning was stopped at this timing, the output could not follow the target.

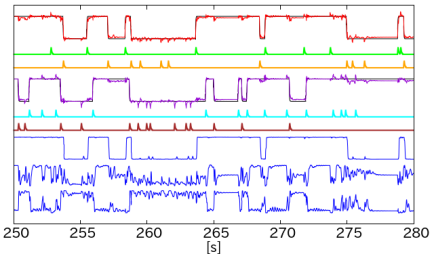
Fig.2(b) shows the activities of the network for 30 seconds of testing after 250 seconds of learning. The outputs almost match the target with no lag. This result shows that the task can be learned without exploration noises. It is interesting that as the learning progresses with successively given reward and penalty ($M(t)$), the sharp change disappeared gradually. The activity of the network seems to transit from exploration mode to stable mode, and the exploration component from the internal dynamics decreases autonomously.

To observe whether the network can adjust the exploration level autonomously when encountering unknown situation, the rule of learning task was changed suddenly. Fig.2(c) shows the network activities when the ON signal and OFF signal were swapped between u_1 and u_2 and between u_3 and u_4 after 250 seconds of learning. The chaotic activities appeared again and exploration was resumed even without any direction from outside. The network activities for 300 seconds of learning after the rule change are shown in Fig.2(d) in a compressed time scale, and Fig.2(e) shows 30 seconds of testing after that. It shows that the network could resume to explore and successfully learn the task even though the environment was changed suddenly in the middle of learning.

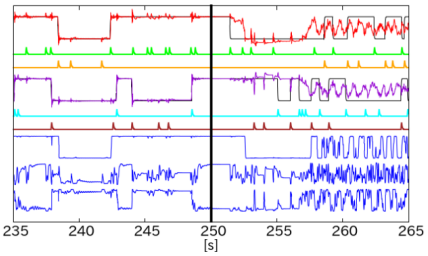
To show how the chaotic activities of network neurons change during learning, the output errors and the output change of network neurons were recorded as



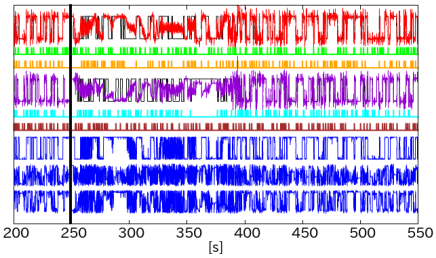
(a) First 30 seconds(training)



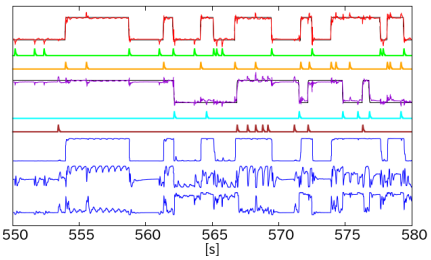
(b) 30 seconds (testing) after 250 seconds learning



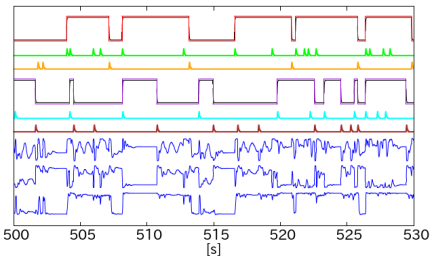
(c) 30 seconds (training) around the rule change at 250 seconds(vertical line)



(d) 350 seconds (training) around the rule change at 250 seconds(vertical line) in a compressed time scale



(e) 30 seconds (testing) after 300 seconds from the rule change



(f) Learning with traces: 30 seconds (testing) after 500 seconds of learning

Fig. 2. Network activities. Output z_1,z_2 are in red, purple respectively. The target value is in black. Input u_1,u_2,u_3,u_4 are in green, orange, cyan, brown respectively. The activities of 3 sample neurons from the network are in blue.

in Fig.3. These values are the mean absolute error of the network output and the mean absolute one-step change in each neuron output over all neurons over every 10000 time steps. In Fig.3, it is seen in the term of 250 seconds from the start of learning when the network learned under the first rule, the output error decreased and the network activities decreased gradually. As soon as the rule was changed at 250 seconds, the output error increased and, a little later, the output change in the network neurons also increased. Then the output change was large though it decreased sometimes, before the change decreased again as the error decreased. It shows that the network can make the internal dynamics chaotic autonomously to explore in unknown situation.

The result after learning procedure with traces is shown in Fig.2(f). In this experiment, we used 0.9 and -0.9 as the maximum and minimum value of targets to prevent reaching a limit of the outputs. Fig.2(f) shows that the reservoir network could be trained with the traces. It seems that the outputs follow the target precisely at a glance, but its values are close to -1.0 or 1.0 that is the upper or lower limit of the output. That means that the output values before the transformation by *tanh* are very large due to large readout weights w_{ji}. In addition, the network parameters needed very sensitive adjustment to learn successfully, and occasionally the output deviated largely. In this case, because the outputs stick upper or lower limit, it was difficult to output intermediate values and was impossible to resume to explore when the rule of task was changed during learning. There still remain problems to be solved.

4 Conclusion

In the Reward-Based Learning of Memory Required Task in a reservoir network, it was confirmed that the internal chaotic dynamics can perform the role of exploration on behalf of the exploration noises added from the outside. As the learning progressed, noise-like fluctuations in the outputs originated from the internal dynamics decreased and the network activities autonomously transited from exploration mode to stable mode gradually. It was also shown that when the task setting was changed during learning, the network adaptively resumed exploration and learned appropriately after that. Using the traces, which is used to train a chaotic NN in the newly-proposed novel RL, the same task could be trained as well, but further investigations are necessary. From these results, it is expected that the learning ability of the reservoir computing can be taken into our approach, and that enables the emergence of higher functions as the result of developing internal dynamics through RL.

Acknowledgement

The authors wish to thank Prof. Hiromichi Suetani for introducing FORCE Learning and the work of Hoerzer et al. to us. This work was supported by JSPS KAKENHI Grant Number 15K00360.

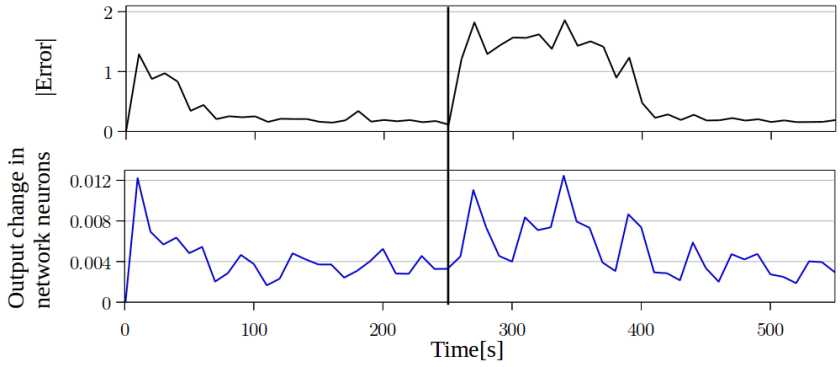


Fig. 3. The mean absolute error(upper) and the mean absolute output change of network neurons(lower) during learning. The vertical line is the timing of rule change.

References

1. K.Shibata and Y.Okabe : Reinforcement Learning When Visual Signals are Directly Given as Inputs, Proc. of ICNN '97, Vol. 3, pp.1716-1720(1997)
2. K.Shibata : Emergence of Intelligence through Reinforcement Learning with a Neural Network, Abdelhamid Mellouk(Ed.), "Advances in Reinforcement Learning" In-Tech, pp.99-120(2011)
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. 2013, ID 15 (2013)
5. Y.Sawatsubashi, et al. : Emergence of discrete and Abstract State Representation in Continuous Input Task, Robot Intelligence Technology and Applications 2012, pp.13-22(2012)
6. K.Shibata and Y.Sakashita : Reinforcement Learning with Internal-Dynamics-based Exploration Using a Chaotic Neural Network, Proc. of Int'l Joint Conf. on Neural Networks (IJCNN)2015, 2015.7
7. Y.Goto and K.Shibata : Emergence of Higher Exploration in Reinforcement Learning using a Chaotic Neural Network, Proc. of ICONIP2016 (2016)(to appear)
8. H.Jaeger : The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, pp.43(2001)
9. W.Maass, T.Natschlger, H.Markram : Real-time computing without stable states: a new framework for neural computation based on perturbations. NEURAL COMPUTATIOM, Vol.14, No.11, pp.2531-2560(2002)
10. D.Sussillo, L.F.Abbott : Generating coherent patterns of activity from chaotic neural networks. Neuron Article, Vol.63, No.4, 544-557(2009)
11. G.M.Hoerzer, R.Legenstein and W.Maass : Emergence of complex computational structures from chaotic neural networks through Reward-Modulated Hebbian Learning. Cerebral Cortex, Vol.24 No.3, pp.677-690 (2014)