2009 International Symposium on Nonlinear Theory and its Applications
NOLTA'09, Sapporo, Japan, October 18-21, 2009

Reinforcement Learning using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution

Yuko OSANA†

†School of Computer Science, Tokyo University of Technology
1404-1 Katakura, Hachioji, Tokyo, 192-0982, Japan Email: osana@cs.teu.ac.jp

Abstract—In this paper, we propose a reinforcement learning method using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD). The proposed method is based on the actor-critic method, and the actor is realized by the KFMPAM-WD. The KFMPAM-WD is based on the self-organizing feature map, and it can realize successive learning and one-to-many associations. The proposed method makes use of this property in order to realize the learning during the practice of task. We carried out a series of computer experiments, and confirmed the effectiveness of the proposed method in path-finding problem.

1. Introduction

The reinforcement learning is a sub-area of machine learning concerned with how an agent ought to take actions in an environment so as to maximize some notion of long-term reward[1]. Reinforcement learning algorithms attempt to find a policy that maps states of the world to the actions the agent ought to take in those states.

Temporal Difference (TD) learning is one of the reinforcement learning algorithm. The TD learning is a combination of Monte Carlo ideas and dynamic programming (DP) ideas. TD resembles a Monte Carlo method because it learns by sampling the environment according to some policy. TD is related to dynamic programming techniques because it approximates its current estimate based on previously learned estimates. The actor-critic method[2] is the method based on the TD learning, and consists of two parts; (1) actor which selects the action and (2) critic which evaluate the action and the state.

On the other hand, neural networks are drawing much attention as a method to realize flexible information processing. Neural networks consider neuron groups of the brain in the creature, and imitate these neurons technologically. Neural networks have some features, especially one of the important features is that the networks can learn to acquire the ability of information processing. The flexible information processing ability of the neural network and the adaptive learning ability of the reinforcement learning are combined, some reinforcement learning method using neural networks are proposed[3]-[5].

In this paper, we propose the reinforcement learning method using Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD). The proposed method is based on the actor-critic method, and the actor is realized by the KFMPAM-WD. The KFMPAM-WD is based on the self-organizing feature map[6], and it can realize successive learning and one-to-many associations. The proposed method makes use of this property in order to realize the learning during the practice of task.

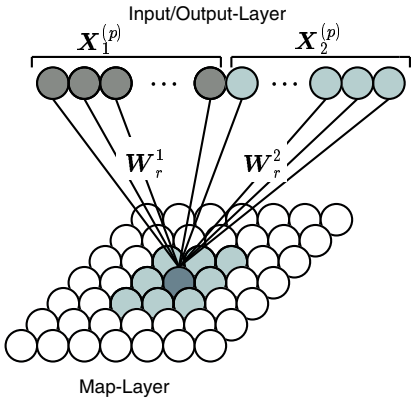


Figure 1: Structure of KFMPAM-WD.

2. KFM Probabilistic Associative Memory based on Weights Distribution

Here, we explain the Kohonen Feature Map Probabilistic Associative Memory based on Weights Distribution (KFMPAM-WD).

2.1. Structure

Figure 1 shows the structure of the KFMPAM-WD. As shown in Fig.1, this model has two layers; (1) Input/Output(I/O)-Layer and (2) Map-Layer, and the I/O-Layer is divided into some parts.

2.2. Learning Process

In the learning algorithm of the KFMPAM-WD, the connection weights are learned as follows:

(1) The initial values of weights are chosen randomly.

(2) The Euclid distance between the learning vector $\boldsymbol{X}^{(p)}$ and the connection weights vector $\boldsymbol{W}_i$, $d(\boldsymbol{X}^{(p)}, \boldsymbol{W}_i)$ is calculated.

(3) If $d(\boldsymbol{X}^{(p)}, \boldsymbol{W}_i) > \theta^t$ is satisfied for all neurons, the input pattern $\boldsymbol{X}^{(p)}$ is regarded as an unknown pattern. If the input pattern is regarded as a known pattern, go to (8).

(4) The neuron which is the center of the learning area r is determined as follows:

$$r = \underset{\substack{i\,:\,D_{iz}+D_{zi}<d_{iz} \\ (\text{for } \forall z \in F)}}{\operatorname{argmin}} d(\boldsymbol{X}^{(p)}, \boldsymbol{W}_i) \tag{1}$$

where F is the set of the neurons whose connection weights are fixed. d_{iz} is the distance between the neuron i and the neuron z whose connection weights are fixed. In Eq.(1), D_{ij} is the radius of the ellipse area whose center is the neuron i for the direction to the neuron j, and is given by

$$D_{ij} = \begin{cases} a_i, & (d_{ij}^y = 0) \\ b_i, & (d_{ij}^x = 0) \\ \sqrt{\dfrac{{a_i}^2 {b_i}^2}{{b_i}^2 + {m_{ij}}^2 {a_i}^2}({m_{ij}}^2 + 1)}, & (\text{otherwise}) \end{cases} \tag{2}$$

where a_i is the long radius of the ellipse area whose center is the neuron i and b_i is the short radius of the ellipse area whose center is the neuron i. In the KFMPAM-WD, a_i and b_i can be set for each training pattern. m_{ij} is the slope of the line through the neurons i and j. In Eq.(1), the neuron whose Euclid distance between its connection weights and the learning vector is minimum in the neurons which can be take areas without overlaps to the areas corresponding to the patterns which are already trained. In Eq.(1), the size of the area for the learning vector are used as a_i and b_i.

(5) If $d(\boldsymbol{X}^{(p)}, \boldsymbol{W}_r) > \theta^t$ is satisfied, the connection weights of the neurons in the ellipse whose center is the neuron r are updated as follows:

$$\boldsymbol{W}_i(t+1) = \begin{cases} \boldsymbol{W}_i(t) + \alpha(t)(\boldsymbol{X}^{(p)} - \boldsymbol{W}_i(t)), & (d_{ri} \le D_{ri}) \\ \boldsymbol{W}_i(t), & (\text{otherwise}) \end{cases} \tag{3}$$

where $\alpha(t)$ is the learning rate and is given by

$$\alpha(t) = \frac{-\alpha_0(t-T)}{T}. \tag{4}$$

Here, α_0 is the initial value of $\alpha(t)$ and T is the upper limit of the learning iterations.

(6) (5) is iterated until $d(\boldsymbol{X}^{(p)}, \boldsymbol{W}_r) \le \theta^t$ is satisfied.

(7) The connection weights of the neuron r $\boldsymbol{W}_r$ are fixed.

(8) (2)~(7) are iterated when a new pattern set is given.

2.3. Recall Process

In the recall process of the KFMPAM-WD, when the pattern $\boldsymbol{X}$ is given to the I/O-Layer, the output of the neuron i in the Map-Layer, x_i^{map} is calculated by

$$x_i^{map} = \begin{cases} 1, & (i = r) \\ 0, & (\text{otherwise}) \end{cases} \tag{5}$$

where r is selected randomly from the neurons which satisfy

$$\frac{1}{N^{in}} \sum_{k \in C} g(X_k - W_{ik}) > \theta^{map} \tag{6}$$

where θ^{map} is the threshold of the neuron in the Map-Layer, and $g(\cdot)$ is given by

$$g(b) = \begin{cases} 1, & (|b| < \theta^d) \\ 0, & (\text{otherwise}). \end{cases} \tag{7}$$

In the KFMPAM-WD, one of the neurons whose connection weights are similar to the input pattern are selected randomly as the winner neuron. So, the probabilistic association can be realized based on the weights distribution.

When the binary pattern $\boldsymbol{X}$ is given to the I/O-Layer, the output of the neuron k in the I/O Layer x_k^{io} is given by

$$x_k^{io} = \begin{cases} 1, & (W_{rk} \ge \theta_b^{io}) \\ 0, & (\text{otherwize}) \end{cases} \tag{8}$$

where θ_b^{io} is the threshold of the neurons in the I/O-Layer.

When the analog pattern $\boldsymbol{X}$ is given to the I/O-Layer, the output of the neuron k in the I/O Layer x_k^{io} is given by

$$x_k^{io} = W_{rk}. \tag{9}$$

3. Reinforcement Learning using KFMPAM-WD

Here, we explain the proposed reinforcement learning method using KFMPAM-WD.

3.1. Outline

In the proposed method, the actor in the Actor-Critic[2] is realized by the KFMPAM-WD. In this research, the I/O-Layer in the KFMPAM-WD is divided into two parts corresponding to the state $\boldsymbol{s}$ and the action $\boldsymbol{a}$, and the actions for the states are memorized.

In this method, the critic receives the states which are obtained from the environment, the state is estimated and the value function is updated. Moreover, the critic outputs Temporal Difference (TD) error to the actor. The KFMPAM-WD which behaves as the actor (we call this "actor network") is trained based on the TD error, and selects the action from the state of environment. Figure 2 shows the flow of the proposed method.

3.2. Actor Network

In the proposed method, the actor in the Actor-Critic[2] is realized by the KFMPAM-WD.

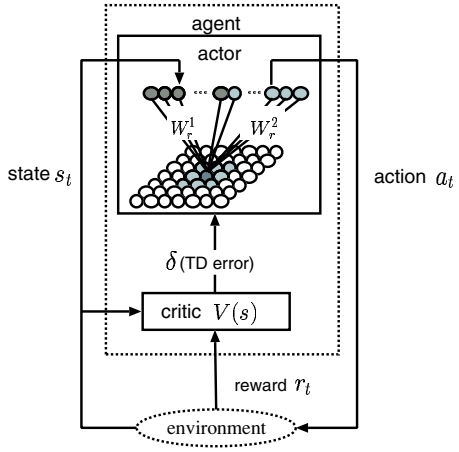


Figure 2: Flow of Proposed Method.

3.2.1. Dynamics

In the actor network, when the state $\boldsymbol{s}$ is given to the I/O-Layer, the corresponding action $\boldsymbol{a}$ is recalled. In the proposed method, the other action is also selected randomly (random selection), and the more desirable action from the recalled action and the action selected in the random selection is chosen as the action finally.

When the pattern $\boldsymbol{X}$ is given to the network, the output of the neuron i in the Map-Layer at the time t $x_i^{map}(t)$ is given by Eq.(5), and the output of the neuron k in the I/O-Layer at the time t $x_k^{io}(t)$ is given by Eq.(8). In the actor network, only the state information is given, so the input pattern is given by

$$\boldsymbol{X} = (\boldsymbol{s}(t), \boldsymbol{0})^T \tag{10}$$

where $\boldsymbol{s}(t)$ is the state at the time t.

3.2.2. Learning

The actor network is trained based on the TD error from the critic.

The learning vector at the time t $\boldsymbol{X}^{(t)}$ is given by the state $\boldsymbol{s}(t)$ and the corresponding action $\boldsymbol{a}(t)$ as follows.

$$\boldsymbol{X}^{(t)} = (\boldsymbol{s}(t), \boldsymbol{a}(t))^T \tag{11}$$

(1) When State and Action are Stored

When the pair of the state and the selected action are memorized in the actor network, the area size corresponding to the pair is updated. If the TD error is larger than 0, the area is expanded. If the TD error is smaller than 0, the area is reduced.

(2) When State and Action are not Stored and TD error is Larger than 0

When the pair of the state and the selected action are not memorized in the actor network and the TD error is larger than 0, the pair is trained as new pattern.

3.3. Reinforcement Learning using KFMPAM-WD

The flow of the proposed reinforcement learning method using KFMPAM-WD is as follows:

(1) The initial values of weights in the actor network are chosen randomly.

(2) The agent observes the environment $\boldsymbol{s}(t)$, and the actor $\boldsymbol{a}(t)$ is selected by the actor network or the random selection.

(3) The state $\boldsymbol{s}(t)$ transits to the $\boldsymbol{s}(t+1)$ by action $\boldsymbol{a}(t)$.

(4) The critic receives the reward $r(\boldsymbol{s}(t+1))$ from the environment $\boldsymbol{s}(t+1)$, and outputs the TD error δ to the actor.

$$\delta = r(\boldsymbol{s}(t+1)) + \gamma V(\boldsymbol{s}(t+1)) - V(\boldsymbol{s}(t)) \tag{12}$$

where γ $(0 \leq \gamma \leq 1)$ is the decay parameter, $V(\boldsymbol{s}(t))$ is the value function for the state $\boldsymbol{s}(t)$.

(5) The eligibility $e_t(\boldsymbol{s})$ is updated.

$$e(\boldsymbol{s}) \leftarrow \begin{cases} \gamma\lambda e(\boldsymbol{s}) & (\text{if } \boldsymbol{s} \neq \boldsymbol{s}(t+1)) \\ \gamma\lambda e(\boldsymbol{s}) + 1 & (\text{if } \boldsymbol{s} = \boldsymbol{s}(t+1)) \end{cases} \tag{13}$$

where γ $(0 \leq \gamma \leq 1)$ is the decay parameter, and λ is the trace decay parameter.

(6) All values for states $V(\boldsymbol{s})$ are updated based on the eligibility $e_t(\boldsymbol{s})$ $(\boldsymbol{s} \in S)$.

$$V(\boldsymbol{s}) \leftarrow V(\boldsymbol{s}) + \xi\delta e_t(\boldsymbol{s}) \tag{14}$$

where ξ $(0 \leq \xi \leq 1)$ is the learning rate.

(7) The connection weights in the actor network are updated based on the TD error (See **3.2.2**).

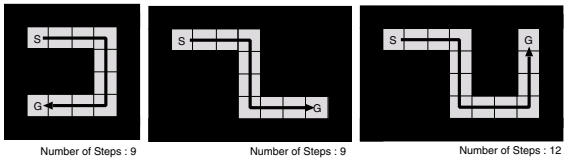
(8) Back to (2).

4. Computer Experiment Results

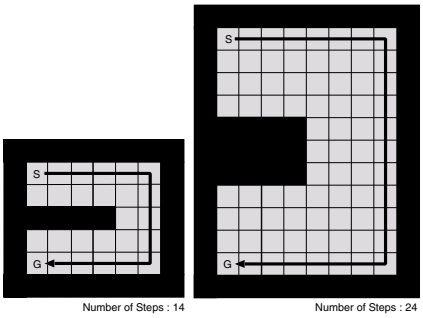
Here, we show the computer experiment results to demonstrate the effectiveness of the proposed method.

We applied the proposed method to the path-finding problem. In this experiment, a agent moves from the start point (S) to the goal point (G). The agent can observe the states of three cells in the lattice, and can move forward/left/right. As the positive reward, we gave 3 when the agent arrived at the goal and 2 when the agent moves. And as the negative reward, we gave -1 when the agent hit against the wall.

Figure 3 shows an example of map and the trained route(arrow). Figure 4 shows the transition of number of steps from the start to the goal in the same trial. Figure 5 shows an example of trained relation between the state and the action.

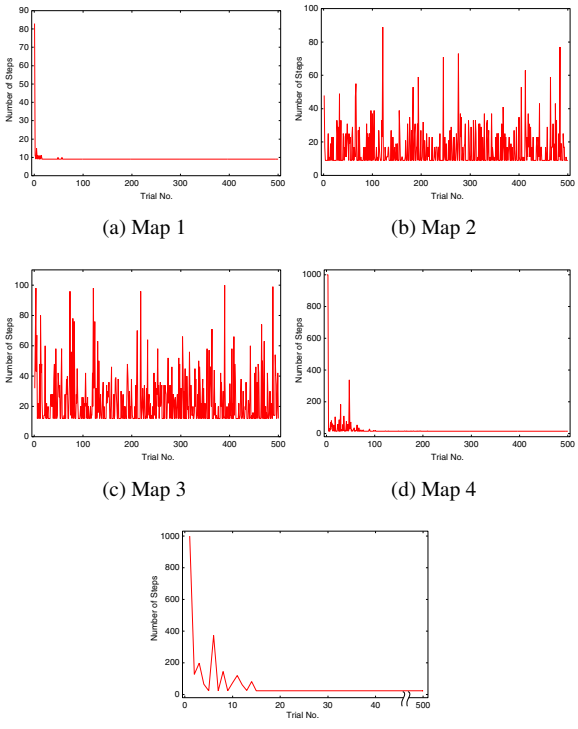


(a) Map 1 (b) Map 2 (c) Map 3



(d) Map 4 (e) Map 5

Figure 3: Map and Trained Route.



(a) Map 1 (b) Map 2

(c) Map 3 (d) Map 4

(e) Map 5

Figure 4: Transition of Steps.

forward	forward	left	forward	right
35 a_i : 4.00 b_i : 3.00	11 a_i : 2.56 b_i : 1.56	3 a_i : 1.90 b_i : 0.90	35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00

(a) Map 1

right	forward	forward	forward	left	right
11 a_i : 2.51 b_i : 1.51	35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00	11 a_i : 2.23 b_i : 1.23	35 a_i : 4.00 b_i : 3.00

(b) Map 2

forward	forward	left	right	left	forward
35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00	11 a_i : 2.51 b_i : 1.51	3 a_i : 1.04 b_i : 0.04	11 a_i : 2.53 b_i : 1.53	35 a_i : 4.00 b_i : 3.00

(c) Map 3

left	forward	right
35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00

(d) Map 4

left	right	forward	forward
35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00	35 a_i : 4.00 b_i : 3.00

(e) Map 5

Figure 5: An example of Trained Relation between State and Action.

5. Conclusion

In this paper, we have proposed the reinforcement learning method using KFMPAM-WD. The proposed method is based on the actor-critic method, and the actor is realized by the KFMPAM-WD. We carried out a series of computer experiments, and confirmed the effectiveness of the proposed method in path-finding problem.

References

[1] R. S. Sutton and A. G. Barto : Reinforcement Learning, An Introduction, The MIT Press, 1998.

[2] I. H. Witten : "An adaptive optimal controller for discrete-time Markov environments," Information and Control, Vol.34, pp. 286–295, 1977.

[3] K. Shibata, M. Sugisaka and K. Ito : "Fast and stable learning in direct-vision-based reinforcement learning," Proceedings of the 6th International Sysmposium on Artificial Life and Robotics, Vol.1, pp.200–203, 2001.

[4] S. Ishii, M. Shidara and K. Shibata: "A model of emergence of reward expectancy neurons by reinforcement learning," Proceedings of the 10th International Sysmposium on Artificial Life and Robotics, GS21–5, 2005.

[5] A. Shimizu and Y. Osana : "Reinforcement learning using Kohonen feature map assocaitive memory with refractoriness based on area representation," Proceedings of International Conference on Neural Information Processing, Auckland, 2008.

[6] T. Kohonen : Self-Organizing Maps, Springer, 1994.