Neural Processing Letters **13:** 213–220, 2001.
© 2001 *Kluwer Academic Publishers. Printed in the Netherlands.*

Reinforcement Learning Using the Stochastic Fuzzy Min–Max Neural Network

ARISTIDIS LIKAS
Department of Computer Science, University of Ioannina, GR 45110 Ioannina, Greece; e-mail: arly@cs.uoi.gr

Abstract. The fuzzy min–max neural network constitutes a neural architecture that is based on hyperbox fuzzy sets and can be incrementally trained by appropriately adjusting the number of hyperboxes and their corresponding volumes. An extension to this network has been proposed recently, that is based on the notion of random hyperboxes and is suitable for reinforcement learning problems with discrete action space. In this work, we elaborate further on the random hyperbox idea and propose the stochastic fuzzy min–max neural network, where each hyperbox is associated with a stochastic learning automaton. Experimental results using the pole balancing problem indicate that the employment of this model as an action selection network in reinforcement learning schemes leads to superior learning performance compared with the traditional approach where the multilayer perceptron is employed.

Key words: fuzzy min–max neural network, pole balancing problem, reinforcement learning, stochastic automaton.

1. Introduction

In the general framework of reinforcement learning, a system accepts inputs from the environment, selects and executes actions and receives a reinforcement signal r that is usually a scalar value rewarding or penalizing the selected actions. A popular approach to deal with such problems is the adaptive heuristic critic method (AHC) based on the method of temporal differences [2, 3, 6]. This method employs two networks: the *action selection network* which provides the action to be executed at each step and the *evaluation network* (or critic) which provides as output a prediction r_{pred} of the evaluation of the current state. The evaluation network is usually a feedforward network that is trained using the error values specified by the method of temporal differences [15]. The action network accepts as input the current problem state and provides the action probabilities p_i $(i = 1, \dots, K)$ (when K distinct actions are assumed) with which the action to be executed is selected [10]. The most widely used model of action selection network is the multilayer perceptron with stochastic output units. Other types of networks have also been proposed belonging to the neurofuzzy family like the fuzzy-ART network [11].

In [9], the fuzzy min–max neural network [13, 14] has been proposed as a model for the action network in the case of reinforcement problems with discrete action space. The operation of the network was suitably adapted in order to be able to cope with the specific requirements imposed by the reinforcement learning framework. For this reason, the notion of *random hyperbox* was introduced, to deal with states of high uncertainty. In the present work,we extend the idea of the random hyperbox and present the stochastic fuzzy min–max network where each hyperbox is associated with a stochastic automaton. More clearly, in the original formulation of the fuzzy min–max network, each hyperbox is characterized by its location and the corresponding class (or action) label. In the proposed extension, the class (or action) label is replaced by a stochastic automaton whose probability vector determines the corresponding action through random selection. Reinforcement learning in the stochastic fuzzy min–max network consists in adjusting not only the location and the boundaries of each hyperbox, but also the probability vector of each stochastic automaton. Details concerning the training of the network are presented in the next section.

2. The Stochastic Fuzzy Min–Max Network

The fuzzy min–max *classification* neural network [13] is an on-line learning classifier based on *hyperbox* fuzzy sets. A hyperbox constitutes a region in the pattern space that can be completely defined once the minimum and the maximum points along each dimension are given. Each hyperbox is associated with exactly one from the pattern classes and all patterns that are contained within a given hyperbox are considered to have full class membership. In the case where a pattern is not completely contained in any of the hyperboxes, a properly computed fuzzy membership function (taking values in [0, 1]) indicates the degree to which the pattern falls outside of each of the hyperboxes. During operation, the hyperbox with the maximum membership value is selected and the class associated with the winning hyperbox is considered as the decision of the network. Learning in the fuzzy min–max classification network is an on-line incremental *expansion–contraction* process that consists of partitioning the input space by creating and adjusting hyperboxes (the minimum and maximum points along each dimension) and also associating a class label to each of them. Details concerning the learning process are provided in [13]. An important issue is that there is only one parameter θ (maximum hyperbox size) that must be specified at the beginning of the learning process. On the other hand, performance is sensitive to the choice of this parameter, which must be empirically specified.

In [9] a modification has been proposed so that the fuzzy min–max network can be used as an action selection network in reinforcement learning problems with discrete action space. Two types of hyperboxes are considered: deterministic, which are associated with a specific action label, and random, in which the corresponding action is selected through *uniform* random selection. In this sense, learning is considered as a process of adding random hyperboxes that later become deterministic as learning proceeds. After an adequate number of steps it is expected that no random hyperboxes will exist any more. Random hyperboxes give the learning system the ability to explore the discrete output space to discover the best action. When such an action is found (according to the evaluation of the critic) it is assigned to the random hyperbox which now becomes deterministic.

In the proposed stochastic fuzzy min–max network, *all* hyperboxes are considered to be random and there is a stochastic automaton associated with each hyperbox. The role of the automaton is to control the degree of randomness in the action selection process. If K distinct actions are assumed, the automaton i corresponding to hyperbox i is characterized by a probability vector $p_i = (p_{i1}, \dots, p_{iK})$ (with $\sum_{j=1}^{K} p_{ij} = 1$). If at a specific time instance the winning hyperbox is i, then the output of the network is determined through random selection using the probability vector p_i.

In order to update the action probabilities of a stochastic automaton i, we have selected the *linear reward–penalty* (L_{R-P}) reinforcement scheme [12]. Assuming that at time instant t, the decision of network is the action k provided from automaton i (i.e. the winning hyperbox is i), then the probability vector p_i is updated as follows:

In the case where the action is rewarded:

$$p_{ij}(t+1) = \begin{cases} p_{ij}(t) + \alpha(1 - p_{ij}(t)) & \text{if } j = k \\ (1-\alpha)p_{ij}(t) & \text{if } j \neq k \end{cases} \tag{1}$$

In the case where the action is penalized:

$$p_{ij}(t+1) = \begin{cases} (1-\beta)p_{ij}(t) & \text{if } j = k \\ \frac{\beta}{K-1} + (1-\beta)p_{ij}(t) & \text{if } j \neq k \end{cases} \tag{2}$$

It holds that $0 < \alpha, \beta < 1$. Moreover, it must be noted that only the parameters of the winning automaton i are modified at step t.

The above probability update equations increase the probability of the selected action in the case of reward and tend to make all actions equiprobable (equal to $1/K$) in the case where penalty is received, since, in the latter case, we actually do not know which is the appropriate action to be reinforced. The parameters α and β control the magnitude of the updates. In the case where their values is close to one, the probabilities are adapted in a fast way following the reinforcement signals, while low values of the parameters lead to slower but more consistent action learning.

The stochastic fuzzy min–max network has been derived from the necessity to overcome some drawbacks of the original formulation based on random hyperboxes. The first drawback is that in a random hyperbox all actions are equiprobable, therefore we cannot express the favor towards a specific action. Once a rewarded action is selected, the random hyperbox becomes immediately deterministic and is labeled with the corresponding action label. This immediate transition from stochastic to deterministic causes problems in many cases, since the rewarded action may not be the best one or the action may not be rewarded again in the future. In the proposed approach, the favor over a specific action (in a given hyperbox) is gradually increased (or decreased) through proper adaptation of the corresponding probability vector. In addition, there is the flexibility to reduce the selection probability of a given action in case this action is not rewarded by the environment any more. On the contrary, in the original formulation [9], there is no available mechanism to change the action label of a given hyperbox. The only available adaptation mechanism is to create a new random hyperbox inside the original hyperbox and appropriately shrink both of them to avoid overlapping. This leads to the construction of an excessive number of hyperboxes with small volume which are more difficult to be adapted by the learning algorithm.

The proposed method completely distinguishes between the two procedures related to learning in reinforcement environments. The first is the adjustment of the position and volume of each hyperbox that is performed using the original expansion-contraction process (governed by the parameter θ) of the fuzzy min–max network. The second is the assignment of the action label corresponding to each hyperbox. This is based on the adjustment of the parameters of the associated stochastic automaton using the reinforcement value r provided by the environment and the evaluation r_{pred} provided by the critic. In this way, a penalized action does not lead to the creation of a new hyperbox, but in most cases leads only to the appropriate adjustment of the corresponding probability values.

The on-line training algorithm for the stochastic fuzzy min–max network can be summarized as follows: Assume a new input is presented to the network.

2.1. ACTION SELECTION

- If i is the winning hyperbox then the action is selected using the probability vector of the automaton i.
- If no winning hyperbox is found for that input point (i.e. no hyperbox meets the expansion criterion) then a new hyperbox is added centered at the specific point and the probabilities of the corresponding automaton are set equal to $1/K$. The network output is selected using these probability values.

2.2. ADAPTATION

Let r be the reinforcement signal provided by the environment and r_{pred} the output of the critic after execution of the selected action.

- if $|r - r_{\text{pred}}| < \delta$ (where δ is a small value) no learning takes place, since it is not safe to classify the evaluation of the action as reward or penalty.
- In the case of reward ($r - r_{\text{pred}} > \delta$) or penalty ($r - r_{\text{pred}} < -\delta$)
  - The probability values of the corresponding automaton are adjusted using the learning Equations (1), (2).

- If the winning automaton i has been expanded to include the input point, then the usual expansion–contraction process for the fuzzy min–max network takes place to avoid overlapping hyperboxes.

3. Application to the Pole Balancing Problem

The pole balancing problem constitutes the best-studied reinforcement learning benchmark. It consists of a single pole hinged on a cart that may move left or right on a horizontal track of finite length. The pole has only one degree of freedom (rotation about the hinge point). The control objective is to push the cart either left or right with a force so that the pole remains balanced and the cart is kept within the track limits.

Four state variables are used to describe the status of the system at each time instant: the horizontal position of the cart (x), the cart velocity ($\dot{x}$), the angle of the pole (ϕ) and the angular velocity ($\dot{\phi}$). At each step the action network must decide the direction and magnitude of force F to be exerted on the cart. Details concerning the equations of motion of the cart-pole system can be found in [2, 11]. Through Euler's approximation method we can simulate the cart-pole system using discrete-time equations with time step $\Delta\tau = 0.02$ sec. We assume that the system's equations of motion are not known to the controller, which perceives only the state vector at each time step. Moreover, we assume that a failure occurs when $|\phi| > 12$ degrees or $|x| > 2.4\,\mathrm{m}$ and that a cycle has been successfully completed if the pole remains balanced for more than 120,000 consecutive time steps. Two versions of the problem exist concerning the magnitude of the applied force F. We are concerned with the case where the magnitude is fixed (equal to $10\,\mathrm{N}$) and the controller must decide only on the direction of the force at each time step. Obviously the control problem is more difficult compared to the case where any value for the magnitude is allowed. Therefore, comparisons will be presented only with fixed magnitude approaches and we will not consider architectures like the RFALCON [11], which are more efficient but assume continuous values for the force magnitude and the control problem is easier to solve.

Experiments have been conducted to assess the performance of the AHC method with the stochastic fuzzy min–max (SFMM) network as an action network. For comparison purposes we have also implemented the AHC approach using the multilayer perceptron (MLP) with stochastic output units as well as the previous fuzzy min–max (FMM) network [9] as action networks. The equation of motion, system parameters and the architecture of the multilayer perceptron (with five hidden nodes) were exactly the same as those reported in [1, 2]. In addition, in all approaches we used exactly the same multilayer perceptron architecture as a critic. Training speed is measured in terms of the number of cycles required to achieve pole balancing. A series of 50 experiments were conducted using each method, with each cycle starting with random initial state variables.

Table I. Training performance in terms of required number of training cycles when the stochastic fuzzy min–max (SFMM) (for several values of θ), the multilayer perceptron (MLP) and the fuzzy min–max (FMM) [9] are used as action networks in the AHC framework. Also the percentage of successful runs and the average number of created hyperboxes are displayed.

			Number of Cycles				
Network	θ	Success (%)	Best	Worst	Mean	SD	No. Hyperboxes
SFMM	0.15	100	1852	9952	4210	1350	62
SFMM	0.25	92	89	10897	3450	1400	28
SFMM	0.35	70	40	12093	2520	2130	14
SFMM	0.45	62	15	14375	2970	3540	10
MLP		72	4123	12895	5175	2284	
FMM	0.1	60	3545	13755	4872	3270	275

The termination criterion for each experiment was the following: When a successful cycle (lasting more than 120,000 steps) was encountered, the system was placed at the zero initial state and a new cycle was started without learning, i.e. adaptation of the network parameters. If this cycle was also successful, then the experiment was terminated, otherwise a new learning cycle was started from random initial state. This criterion was set to ensure that after training, the system was able to successfully operate starting from the zero initial state.

In the employed stochastic fuzzy min–max network we have used the following parameter values: $\delta = 0.1$, $\alpha = 0.9$ and $\beta = 0.9$. In addition, at each cycle we started with stochastic action selection and after 200 steps we switched to deterministic action selection, ie. selection of the action with highest probability value. This modification has been found to increase learning performance [7] and has also been used in the experiments with the multilayer perceptron.

Obtained results are summarized in Table I, for several values of the learning parameter θ of the stochastic fuzzy min–max network. The table provides the percentage of successful experiments, the statistics of required number of training cycles and the average number of created hyperboxes. For each method the displayed results are only taken from the successful experiments. It is clear that the stochastic fuzzy min–max network exhibits significantly better performance compared to the multilayer perceptron in terms of the required number of training cycles. For comparison purposes we have also conducted experiments using the fuzzy min–max (FMM) network that is based on random hyperboxes [9]. Best results using this network (obtained for the value of $\theta = 0.1$) are displayed in Table I and indicate the superiority of the proposed stochastic fuzzy min–max network. A serious drawback of the FMM approach is that it led to the creation of an excessive number of small volume hyperboxes as already noted in Section 2.

In what concerns the of the stochastic fuzzy min–max network it is clear that the performance is sensitive to the value of parameter θ, which specifies the maximum allowed volume for every hyperbox. For small values of θ (eg. $\theta = 0.15$), many hyperboxes are created and the training algorithm is more reliable, since it always provides a solution. As expected, in order for the position and volume of many hyperboxes to be adjusted, many training cycles are required and, therefore, training time is longer. As the value of θ increases, less hyperboxes are needed to cover the state space and this results in an increase in training speed, but training is less reliable and the number of unsuccessful experiments increases. Therefore, in order to select a value for θ, one has to appropriately weigh the above mentioned conflicting aspects.

Finally, it must be noted that the basic characteristic of the fuzzy min–max network is that it provides a partitioning of the problem input space using hyperboxes and assigns an action label to each hyperbox. Since each hyperbox actually defines a rule in the input space, the proposed stochastic fuzzy min–max network can be considered as a technique for deriving rule-based controllers in reinforcement learning problems. Consequently, the proposed learning method can be viewed as a rule extraction technique in the reinforcement learning framework. Since the significance of rule-based model descriptions is widely acknowledged, the proposed network has an additional advantage over the multilayer perceptron, which needs considerable post processing to achieve rule extraction [5].

References

1. Anderson, C. W.: Strategy learning with multilayer connectionist representations, Technical Report TR87-509.3, GTE Labs, Waltham, MA.
2. Anderson, C. W.: Learning to control an inverted pendulum using neural networks, *IEEE Control Systems Magazine*, **2** (1989), 31–37.
3. Barto, A. G., Sutton, R. S. and Anderson, C. W.: Neuron like elements that can solve difficult control problems, *IEEE Trans. on Systems, Man and Cybernetics*, **13** (1983), 835–846.
4. Berenji, H. R. and Khedkar, P.: Learning and tuning fuzzy logic controllers using reinforcements, *IEEE Trans. on Neural Networks*, **3** (1992), 724–740.
5. Bigus, J. P.: *Data Mining with Neural Networks*, McGraw-Hill, (1996).
6. Kaelbing, L., Littman, M. and Moore, A.: Reinforcement learning: A survey, *J. Artificial Intelligence Res.*, **4** (1996), 237–285.
7. Kontoravdis, D., Likas, A. and Stafylopatis, A.: Efficient reinforcement learning strategies for the pole balancing problem, In: M. Marinaro and P. Morasso (eds), *Proc. ICANN'94*, Springer-Verlag, (1994), pp. 659–662.
8. Likas, A., Blekas, K. and Stafylopatis, A.: Application of the fuzzy min-max neural network classifier to problems with continuous and discrete attributes, In: *Proc. IEEE Workshop on Neural Networks for Signal Processing*, Ermioni, Greece, (1994), pp. 163–170.
9. Likas, A. and Blekas, K.: A reinforcement learning approach based on the fuzzy min-max neural network, *Neural Processing Letters*, **4**(3) (1996), 167–172.
10. Lin, L.: Self-improving reactive agents based on reinforcement learning, planning and teaching, *Machine Learning*, **8** (1992), 293–321.

11. Lin, C-J. and Lin, C-T.: Reinforcement learning for an ART-based fuzzy adaptive learning control network, *IEEE Trans. on Neural Networks*, **7** (1996), 709–731.
12. Narendra, K. S. and Thathachar, M. A.: *Learning Automata*, Prentice-Hall, (1989).
13. Simpson, P. K.: Fuzzy min-max neural networks–Part 1: Classification, *IEEE Trans. on Neural Networks*, **3**(5) (1992), 776–786.
14. Simpson, P. K.: Fuzzy min-max neural networks–Part 2: Clustering, *IEEE Trans. on Fuzzy Systems*, **1**(1) (1993), 32-45.
15. Sutton, R. and Barto, A.: *Reinforcement Learning: An Introduction*, MIT Press, (1998).