Incremental Learning in Dynamic Environments Using Neural Network with Long-term Memory

Kenji Tsumori and Seiichi Ozawa
Graduate School of Science and Technology, Kobe University
1-1 Rokko-dai, Nada, Kobe 657-8501, JAPAN
Email: tsumori@frenchblue.scitec.kobe-u.ac.jp
ozawa@eedept.kobe-u.ac.jp

***Abstract*— When the environment is dynamically changed for agents, knowledge acquired from an environment might be useless in the future environments. Therefore, agents should not only acquire new knowledge but also modify or delete old knowledge. However, these modification and deletion are not always efficient in learning. Because the knowledge once acquired in the past can be useful again in the future when the same environment reappears. To learn efficiently in this situation, agents should have memory to store old knowledge. In this paper, we propose an agent architecture that consists of four modules: resource allocating network (RAN), long-term memory (LTM), association buffer (A-Buffer), and environmental change detector (ECD). In LTM, not only acquired knowledge but also the information about which knowledge was produced in the same environment is stored. This information is utilized for recalling the knowledge acquired in the past when the same environment reappears. To evaluate the adaptability in a class of dynamic environments, We apply this model to a simple problem that some target functions to be approximated are changed in turn. As a result, we verify the following adaptability of RAN-ALTM: (1) incremental learning can be stably carried out, (2) environmental changes are correctly detected, (3) fast adaptation is realized by training some of the accumulated knowledge when the past environments reappear.**

I. Introduction

In general, real-world environments surrounding agents have a lot of uncertain and dynamical properties. In the environments, all of the knowledge acquired in the past is not always effective; hence the agents have to acquire new knowledge incrementally.

Several approaches to the agent learning under dynamic environments have been proposed so far [1], [2]. In many approaches, the adaptability of agents is realized by reconstructing their models when the environment changes. However, when the same (or similar) environments appear repeatedly over a long period of time, one can say that it is not effective to discard or modify the current knowledge to adapt to new environments. In this situation, even if the knowledge is useless at some point, agents should keep it themselves so as to retrieve it in the future if necessary. Furthermore, it is desired to store only essential knowledge because the quantity of knowledge could be huge for large tasks, e.g. continuing tasks. To do this, we should present a new agent architecture that is different from the conventional memory-based (or instance-based) learning approaches [3], [4].

In order to extract essential knowledge from training samples, multi-layer neural networks have been often used. In neural networks, however, knowledge is distributedly stored in their connection weights. Hence, when a new training sample is given to a network, the input-output relations acquired in the past are easy to be collapsed by the learning of the new sample. This disruption in neural networks is called 'catastrophic interference' that is caused by excessive adaptation of connection weights for a new training sample [5].

To solve the problem of catastrophic interference, there have been proposed several approaches [5], [6], [7], [8], [9]. A promising approach is that some representative input-output pairs are extracted from sequentially given training samples and some of them are trained with a current training sample. Based on this approach, we have proposed an extended version of RBF networks called *Resource Allocating Network with Long-Term Memory* (RAN-LTM) [9]. RAN-LTM does not need so much memory capacity and it realizes robust incremental learning ability. In this sense, RAN-LTM has some of the characteristics suitable for real-world environments except that RAN-LTM is not designed for dynamic environments.

In this paper, several new functions are introduced into the original RAN-LTM so that it can work well under dynamic environments. In Section 2, the assumption for dynamic environments is described, and then we propose new agent architecture in Section 3. In Section 4, the adaptability of the proposed model is evaluated through some simulation experiments. Finally we state conclusions in Section 5.

II. A Model of Learning Agent

A. Assumption for Dynamical Environments

In this paper, we assume a class of dynamic environments where a stationary environment continues for a while and then it changes to another stationary environment in turn. And it is assumed that the duration of an environment is long enough to be learned but it is unknown for agents. Furthermore, we assume that stationary environments appear repeatedly over a long period of time. That is to say, if agents hold their knowledge acquired in the past and retrieve it to learn in the future, the agents could adapt quickly to the reappeared environments.

B. Agent Architecture

In order to adapt efficiently under the dynamic environment defined in II-A, agents must possess at least the following capabilities:

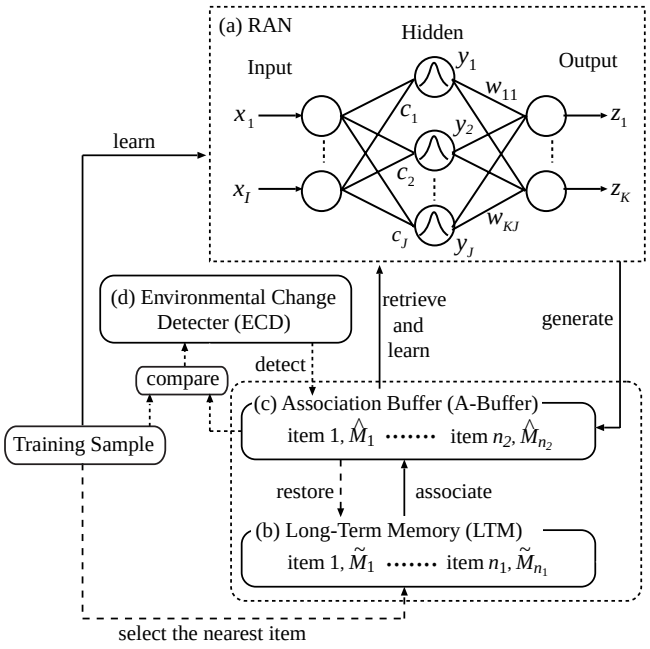

Fig. 1. Architecture of RAN-ALTM.

1) the capability to acquire knowledge from training samples given incrementally,
2) the capability to store the acquired knowledge in memory and to retrieve proper knowledge,
3) the capability to detect environmental changes.

The capability 1) is necessary to extract only essential knowledge from many training samples. The capability 2) is indispensable to do stable incremental learning under dynamic environments and to adapt quickly to the reappeared environments. The capability 3) is necessary to detect when a stationary environment changes to another environment.

As one of the learning agent models with the capabilities 1) and 2), we have proposed Resource Allocating Network with Long-Term Memory (RAN-LTM). To realize all of the above capabilities, we have to augment the capability 3) as well as modify the capability 2) in RAN-LTM. Here, we propose Resource Allocating Network with Associative Long-Term Memory (RAN-ALTM) whose architecture is depicted in Fig. 1. As you can see from Fig. 1, RAN-ALTM is composed of four modules. First, we briefly explain the basic functions of each module.

a) Resource Allocating Network (RAN)
Resource allocating network (RAN) [10] is an extended version of radial-basis function network (RBFN) [11], [12]. RAN can adaptively add its hidden units to extend the approximation ability when unknown training samples are given. When training samples are given incrementally, the forgetting caused by the interference is often problematic in the original RAN. To suppress the interference, some 'memory items' stored in long-term memory are learned with a training sample.

b) Long-Term Memory (LTM)
LTM is a memory whose items $\{\tilde{M}_1, \cdots, \tilde{M}_{n_1}\}$ are stored for a long time. Here, n_1 is the number of memory items in LTM. Under a dynamic environment, many memory items generated under different stationary environments are mingled in LTM. Memory items which were generated in the same environment as a current training sample are moved to Association Buffer using the association function.

c) Association Buffer (A-Buffer)
A-Buffer is a place where associated memory items of LTM are stored temporarily. Associated memory items in A-Buffer is denoted as $\{\hat{M}_1, \cdots, \hat{M}_{n_2}\}$. Here, n_2 is the number of memory items in A-Buffer. Some memory items that are effective in suppressing the interference are retrieved from A-Buffer and they are trained in RAN. When an environmental change is detected by Environmental Change Detector, A-Buffer is initialized.

d) Environmental Change Detector (ECD)
ECD is a module which detects environmental changes by checking the inconsistency between a training sample and the memory items in A-Buffer.

Next we describe the detail information processing of the above modules $a) \sim d)$.

C. Resource Allocating Network (RAN)

Let the numbers of units in input, hidden, and output layers be I, J, and K, respectively. When an input $\boldsymbol{x} = \{x_1, \cdots, x_I\}'$ is given to RAN (see Fig. 1 (a)), the jth hidden output y_j and the kth network output are calculated as follows:

$$y_j = \exp\left(-\frac{\| \boldsymbol{x} - \boldsymbol{c}_j \|^2}{2\sigma_j^2}\right), \quad j = 1, \cdots, J \tag{1}$$

$$z_k = \sum_{j=1}^{J} w_{kj} y_j + \gamma_k, \quad k = 1, \cdots, K \tag{2}$$

where $\boldsymbol{c}_j = \{c_{j1}, \cdots, c_{jI}\}'$ and σ_j^2 are the center and variance of the jth hidden unit. And w_{kj} and γ_k are respectively a connection weight from the jth hidden unit to the kth output unit and a bias of the kth output unit.

In RAN, connection weights and RBF centers can be modified to improve the approximation accuracy of network outputs. The modification of connection weights and centers is carried out based on the gradient descent method. After the mean squared error E between outputs $\boldsymbol{z}$ and targets $\boldsymbol{T}$ is evaluated, the training of RAN is conducted as follows:

(1) If E is larger than a positive constant ε and the distance between input $\boldsymbol{x}$ and its nearest center vector $\boldsymbol{c}^*$ is larger than a positive value $\delta(t)$ (i.e., $E > \varepsilon$ and $\|\boldsymbol{x} - \boldsymbol{c}^*\| > \delta(t)$), then add a hidden unit to RAN (i.e., $J \leftarrow J +$

1). Set the following values to the network parameters for the Jth hidden unit (center vector $\boldsymbol{c}_J$, connection weights w_{kJ}, and variance σ_J):

$$c_{Ji} = x_i, \quad i = 1, \cdots, I \tag{3}$$

$$w_{kJ} = T_k - z_k, \quad k = 1, \cdots, K \tag{4}$$

$$\sigma_J = \kappa \|\boldsymbol{x} - \boldsymbol{c}^*\| \tag{5}$$

where κ is a positive constant. Decrease $\delta(t)$ with time t as follows:

$$\delta(t) = \max\left[\delta_{max} \exp\left(\frac{t}{\tau}\right), \delta_{min}\right] > 0 \tag{6}$$

where τ is a decay constant.

(2) Otherwise
Modify the network parameters as follows:

$$w_{kj}^{NEW} = w_{kj}^{OLD} + \alpha(T_k - z_k) y_j \tag{7}$$

$$c_{ji}^{NEW} = c_{ji}^{OLD} + \frac{\alpha}{\sigma_j}(x_p - c_{ji}) y_j \sum_k (T_k - z_k)\sigma_{kj} \tag{8}$$

$$\gamma_k^{NEW} = \gamma_k^{OLD} + \alpha(T_k - z_k) \tag{9}$$

where α is a positive learning ratio.

D. Long-Term Memory (LTM)

1) Structure of Memory Item: As mentioned before, memory items are extracted from the mapping function of RAN. The purpose of learning these memory items is as follows:

(1) to suppress the forgetting caused by learning incrementally,
(2) to retrieve memory items generated in the past environments that are the same as (or similar to) the current environment, and to adapt to the current environment quickly.

To realize the above functions, memory items, $\tilde{M}_j$ $(j = 1, \cdots, n_1)$, are composed of not only an input-output pair but also some information for the interference suppression and the association. The jth memory item, $\tilde{M}_j$ $(j = 1, \cdots, n_1)$, consists of the following information:

i) Input-Output Pair, $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j)$
An input-output pair extracted from the mapping function acquired by RAN.

ii) Curvature Information, $H(\tilde{\boldsymbol{x}}_j)$
Curvature information of the mapping function at $\tilde{\boldsymbol{x}}_j$. In general, many RBF centers tend to be allocated in a complicated domain; hence, the interference could be serious in this domain. Therefore, we quantify the functional complexity by the curvature. If this curvature information is large, the corresponding memory item is recalled with high probability.

iii) Linkage Information Table (LIT)
A set of memory items, $\tilde{M}_k$ $(k \neq j)$, which were generated in the same environment as $\tilde{M}_j$, are registered in LIT. The degree of relations Q_k with the jth memory item $\tilde{M}_j$ is also resisted in LIT. If Q_k is large, it means that $\tilde{M}_j$ has strong relations with $\tilde{M}_k$. Several $\tilde{M}_k$ with large Q_k are moved to A-Buffer.

iv) Validity Information, V_j
This information V_j is set to a constant value when $\tilde{M}_j$ is moved to A-Buffer. If $\tilde{M}_j$ is the most similar memory item to the current training sample, V_j is set to a large value. Otherwise, V_j is set to a small value that decreases by the times of recursive associations (see also II-D.3). This value corresponds to the term of validity within A-Buffer. Whenever a new training sample is given, V_j decreases by 1. If V_j becomes 0, the corresponding memory item $\tilde{M}_j$ is move back to LTM. When an environmental change is detected in ECD, all V_j are initialized and all memory items in A-Buffer are restored to LTM.

v) Association Flag, F_j
This flag indicates whether the association of $\tilde{M}_j$ is permitted or not: $F_j = 1$ (permitted), $F_j = 0$ (not permitted).

2) Generation of Memory Items: The generation of memory items is carried out only for the input domain where the accuracy of function approximation is ensured. In order to prevent memory items from increasing too much, it is generated only when the distance from any other memory items is more than a constant value. If the above conditions are satisfied, the RBF center $\boldsymbol{c}_j^*$ of the most activated hidden unit is given to RAN, and the output $\boldsymbol{z}$ is calculated. This input-output pair $(\boldsymbol{c}_j^*, \boldsymbol{z})$ is generated as a new memory item. The procedure of generating memory items is shown below.

[Procedure of Generation]

(1) When a training sample $(\boldsymbol{x}, \boldsymbol{T})$ is given to RAN, calculate the hidden outputs, y_j $(j = 1, \cdots, J)$.
(2) If all hidden outputs are less than a threshold value θ_c, then go to Step 7. Otherwise, go to Step 3.
(3) Obtain all indices j of hidden units whose outputs y_j are larger than θ_c, and define a set $\mathcal{I}_1$ of these indices. Update the following approximation criterion r_j for all $j \in \mathcal{I}_1$:

$$r_j^{NEW} = r_j^{OLD} + 2\exp(-\rho|E|) - 1$$

where E is the output error and ρ is a positive constant.
(4) If $r_j > \beta$ for $j \in \mathcal{I}_1$, then initialize r_j and go to Step 5. Otherwise, go to Step 7. Here, β is a positive constant.
(5) Calculate the minimum distance between the jth center $\boldsymbol{c}_j$ and memory items $\tilde{\boldsymbol{x}}_m$ $(m = 1, \cdots, M)$ as follows:

$$d_j^* = \min_m \|\boldsymbol{c}_j - \tilde{\boldsymbol{x}}_m\|.$$

If $d_j^* > \eta$ for $j \in \mathcal{I}_1$, then go to Step 6. Otherwise go to Step 7.
(6) Increase the number of memory items M by one (i.e., $M \leftarrow M + 1$). Give the jth center vector $\boldsymbol{c}_j$ to RAN as its input, and obtain the output $\boldsymbol{z}$. Calculate the determinant $H(\boldsymbol{c}_j)$ of Hessian matrix whose element

(i, i') is given as follows:

$$\frac{\partial^2 \tilde{z}_k}{\partial c_{ji} \partial c_{ji'}} = \frac{1}{\sigma_i^2 \sigma_{i'}^2} \sum_{l=1}^{J} w_{kl}(c_{ji} - c_{li})(c_{ji'} - c_{li'}) y_l + \gamma_k.$$

Store the triplet $(\boldsymbol{c}_j, \boldsymbol{z}, H(\boldsymbol{c}_j))$ in LTM as the Mth memory item $(\tilde{\boldsymbol{x}}_M, \tilde{\boldsymbol{z}}_M, H(\tilde{\boldsymbol{x}}_M))$. Initialize LIT, V_j, and F_j.

(7) Go back to Step 1.

3) Association of Memory Items: When a training sample $(\boldsymbol{x}, \boldsymbol{T})$ is given to RAN-ALTM, the most similar memory item $\tilde{M}_j : (\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{y}}_j)$ is moved from LTM to A-Buffer. Then, the LIT of $\tilde{M}_j$ is referred, and memory items $\tilde{M}_k$ with the largest m_1 of Q_k are associated and moved to A-Buffer. For every $\tilde{M}_k$, the same association is conducted recursively and m_2 memory items are moved to A-Buffer. Such a recursive operation is repeated designated times. Note that the association is not done for memory items with association flag $F_k = 0$.

Using this association mechanism, many memory items related to a given training sample are move to A-Buffer, and some of them are retrieved and learned with the training sample. The associated memory items correspond to the knowledge acquired in the past environment similar to the current environment. Hence, we can say that this is the key mechanism to adapt quickly to new environments in RAN-ALTM.

E. Association Buffer (A-Buffer)

1) Retrieval of Memory Items: To suppress the forgetting by incremental learning, memory items in A-Buffer are retrieved and trained with a training sample in RAN. However, it is not efficient to retrieve and train all the memory items. Thus we should restrict the memory items which can suppress the interference effectively. The algorithm of retrieving memory items is shown below.

[Procedure of Retrieval]

(1) When a training sample $\boldsymbol{x}$ is given to RAN, calculate hidden outputs, y_j $(j = 1, \cdots, J)$.

(2) For the jth hidden unit, find a memory item $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j, H(\tilde{\boldsymbol{x}}_j))$ whose input vector $\tilde{\boldsymbol{x}}_j$ is the nearest to the jth RBF centers $\boldsymbol{c}_j$. Repeat the above operation for all hidden units.

(3) For each hidden unit, calculate the recall probability P_j from y_j and $H(\tilde{\boldsymbol{x}}_j)$:

$$P_j = \frac{1}{1 + \exp[-\nu\{y_j + H'(\tilde{\boldsymbol{x}}_j)\} + \lambda]}.$$

Here, ν and λ are positive constants, and $H'(\tilde{\boldsymbol{x}}_j)$ is Hessian information obtained as follows:

$$H'(\tilde{\boldsymbol{x}}_j) = \min\{\frac{|H(\tilde{\boldsymbol{x}}_j)|}{H_0}, 1\}$$

where H_0 is a positive constant.

(4) Retrieve input-output pairs, $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j)$ $(j = 1, \cdots, J)$, with probability P_j, and train the retrieved pairs as well as the current training sample $(\boldsymbol{x}, \boldsymbol{T})$ in RAN.

(5) Go back to Step 1.

2) Update of LIT: Usually, it is considered that memory items in A-Buffer were generated in the same (or similar) environments. Therefore, they should be mutually registered into LIT and increase the degree of relation Q_k each other. However, if an associated memory item in A-Buffer simultaneously belongs to several different environments, the memory items associated from the above memory item are not ensured that all of them belong to the current environment. Therefore, for this misleading memory item the recursive association should be prohibited. To find misleading memory items, check the consistency among the input-output pairs of all memory items in A-Buffer. If such memory items are found, the association flags F_j are set to 0.

F. Environmental Change Detector (ECD)

Environmental changes are detectable by checking the inconsistency between a training sample and memory items in A-Buffer. The procedure in ECD is shown below.

[Procedure in ECD]

(1) Find the most similar memory item in A-Buffer, $\hat{M}^* : (\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{z}}^*)$, to the training sample $(\boldsymbol{x}, \boldsymbol{T})$.

(2) The index B for detecting environmental changes is updated as follows:

i) If $\|\boldsymbol{x} - \hat{\boldsymbol{x}}^*\| < k_1 / g'(\hat{\boldsymbol{x}}^*)$,

$$B^{NEW} = \|\boldsymbol{T} - \hat{\boldsymbol{z}}^*\| + k_2 B^{OLD},$$

otherwise $\quad B^{NEW} = k_2 B^{OLD}$

where $g'(\cdot)$ is the derivative of the RAN's mapping function, k_1 and k_2 are positive constants.

(3) If $B > k_3 \sigma_z$, then an environmental change is detected. A-Buffer is reset. Here, σ_z is the variance of $\hat{\boldsymbol{z}}_j$ in A-Buffer and k_3 is a positive constant.

(4) Go back to Step 1.

III. Simulation Experiments

A. Evaluation Method

As stated before, we assume here a class of dynamic environments where a stationary environment continues for a while and then it changes to another stationary environment in turn. To evaluate the adaptability under such dynamic environments, let us examine the following capabilities of RAN-ALTM:

(1) whether incremental learning can be stably carried out,

(2) whether environmental changes are correctly detected,

(3) whether the fast adaptation is realized by training some of the accumulated knowledge when the past environments reappear.

Here, we define a simple form of dynamic environments; i.e., three one-dimensional target functions $f_1 \sim f_3$ in Fig. 2 are temporally interchanged. That is, each function corresponds to the desired input-output relations for agents under a stationary environment. A training sample (x, z) is randomly drawn from one of the target functions ($f_1 \sim f_3$), and it gives to RAN-ALTM incrementally. The learning of this target

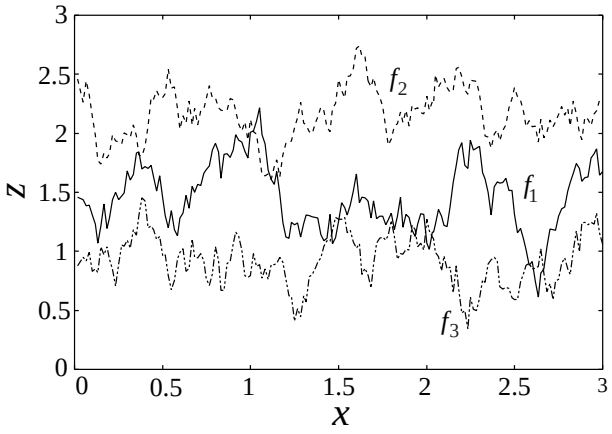

Fig. 2. Three one-dimensional functions: f_1, f_2, and f_3. Each function corresponds to the desired input-output relations for agents under a stationary environment.

function continues for a while, and then the target function is changed to another (this means that an environmental change occurs). Here, we should note that agents do not know the duration of stationary environments and when the environmental changes occur at all.

To check the above three capabilities of RAN-ALTM, we adopt the modified RAN-LTM in which only environmental detector is introduced. For notational convenience, this modified RAN-LTM is simply called RAN-LTM.

B. Experimental Results

Simulation experiments are conducted for 25 different series of three target functions $f_1 \sim f_3$: the series are different in the order of target functions to be presented and their duration. The order of target functions and the duration are determined at random. Needless to say, agents do not know what series are presented.

In the experiments, the following parameters of RAN-ALTM are adopted:

[RAN] $\delta = 0.05$, $\sigma = 0.1$, $\epsilon = 0.0001$, $\alpha = 0.0001$, $\kappa = 0.1$
[LTM] $\theta = 0.9$, $\beta = 9.75$, $\eta = 0.05$, $\rho = 1.0$
[A-Buffer] $\nu = 9.75$, $\lambda = 8.0$
[ECD] $k_1 = 0.02$, $k_2 = 0.3$, $k_3 = 3.0$

Figures 3 (a)-(c) show the temporal transitions of the average errors for the following three series:

(a) $f_2 \rightarrow f_3 \rightarrow f_2 \rightarrow f_3 \rightarrow f_1$
(b) $f_1 \rightarrow f_2 \rightarrow f_3 \rightarrow f_2 \rightarrow f_3$
(c) $f_1 \rightarrow f_3 \rightarrow f_1 \rightarrow f_3 \rightarrow f_2$.

The vertical dotted lines in Figs. 3 (a)-(c) indicate the time of environmental changes.

As you can see from Figs. 3 (a)-(c), although the error suddenly increases after environmental changes, the error decreases immediately. From the results, we can say that

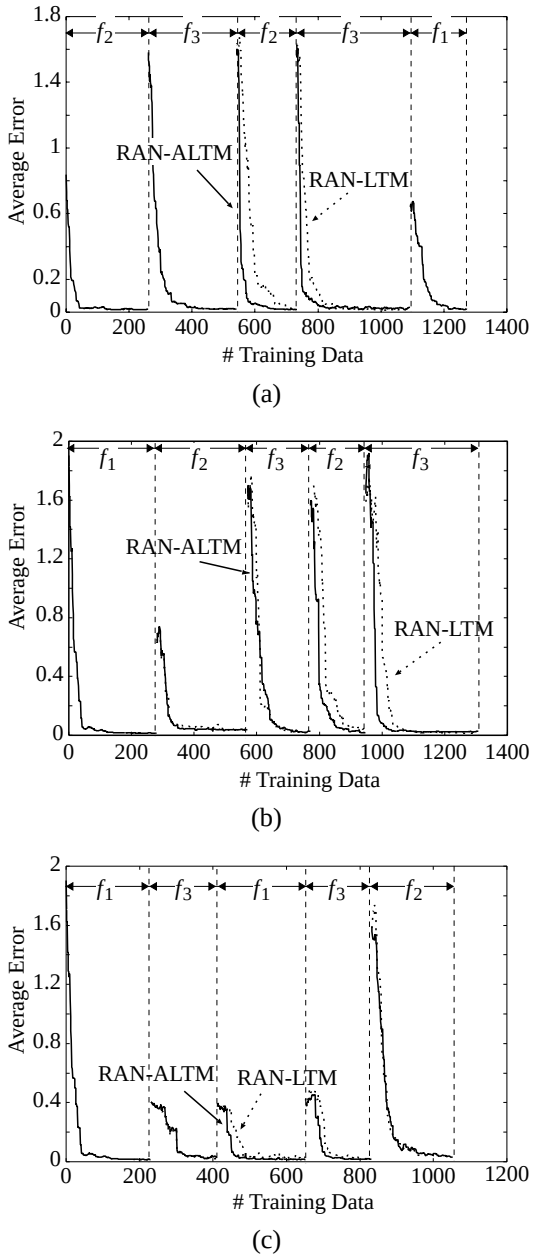

Fig. 3. Temporal transitions of average errors for three different dynamic environments.

environmental changes are correctly detected in ECD. Furthermore, after environments change, the output errors almost always decrease monotonously. This result suggests that both RAN-LTM and RAN-ALTM can learn stably even in incremental settings under dynamic environments. Comparing RAN-ALTM with RAN-LTM, you can see that the former can adapt quickly to the reappeared environments (e.g., f_2 and f_3 in Fig. 3 (a)). To verify this result more precisely, the averages and standard deviations of the speedup enhancement for 25 different dynamic environments. The results are shown in Table I. As seen from Table I, about 27-29 % speedup

TABLE I

THE IMPROVEMENT IN ADAPTATION SPEED OF RAN-ALTM.

	f_1	f_2	f_3
Average (%)	27.13	27.65	29.23
Std. Dev. (%)	2.452	2.333	2.666

enhancement is attained for $f_1 \sim f_3$ on average.

From the above experiments, we can conclude that RAN-ALTM has the high-performance adaptability under dynamic environments.

IV. CONCLUSION

In this paper, we propose an incremental learning model under dynamic environments called Resource Allocating Network with Associative Long-Term Memory (RAN-ALTM). The dynamic environment assumed here is that a stationary environment continues for a while and then it changes to another stationary environment in turn. And it is assumed that the duration of an environment is long enough to be learned but it is unknown for agents. Furthermore, we assume that stationary environments appear repeatedly over a long period of time. Hence, if agents hold their knowledge acquired in the past and retrieve it properly in learning, the agents should be able to adapt quickly to the successive environments.

To verify such adaptation ability of the proposed RAN-ALTM, we defined a simple form of dynamic environments; i.e., three one-dimensional target functions were temporally interchanged. From the experimental results under these dynamic environments, we verified the following adaptability of RAN-ALTM: (1) incremental learning could be stably carried out, (2) environmental changes were correctly detected, (3) fast adaptation was realized using the accumulated knowledge when the past environments reappeared.

However, several problems are still left. One is that we need to adjust adequately the parameter of ECD depending on the complexity of target functions. Hence, we cannot say that ECD is a robust model at the current state. We will improve this point in near future.

ACKNOWLEDGMENT

The authors would like to thank Prof. Shigeo Abe for his useful discussions and comments. This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Young Scientists (B).

REFERENCES

[1] S. B. Thrun and T. M. Mitchell: "Lifelong Robot Learning," *Robotics and autonomous systems*, **15**, 25/46 (1995)

[2] M. A. Wiering: "Reinforcement learning in dynamic environments using instantiated information," *Proc. 18th International Conf. on Machine Learning*, 585/592 (2001)

[3] A. McCallum: "Instance-based utile distinctions for reinforcement learning with hidden state," *Int. Conf. on Machine Learning*, 387/395 (1995)

[4] C. G. Atkeson, A. W. Moore and S. Schaal: "Locally weighted learning," *Artificial Intelligence Review*, **11**, 75/113 (1997)

[5] G.A. Carpenter and S. Grossberg: "The ART of adaptive pattern recognition by a self-organizing neural network," *IEEE Computer*, **21**, 3, 77/88 (1988)

[6] H. Nakayama and M. Yoshida: "Additional learning and forgetting by potential method for pattern classification," *Proc. Int. Conf. on Neural Networks 97*, 1839/1844 (1997)

[7] K. Yamauchi, N. Yamaguchi, and N. Ishii: "Incremental learning methods with retrieving of interfered patterns," *IEEE Trans. on Neural Networks*, **10**, 6, 1351/1365 (1999)

[8] H.-C. Fu, Y.-P. Lee, C.-C. Chiang, and H.-T. Pao: "Divide-and-conquer learning and modular perceptron networks," *IEEE Trans. on Neural Networks*, **12**, 2, 250/263 (2001)

[9] M. Kobayashi, A. Zamani, S. Ozawa and S. Abe: "Reducing computations in incremental learning for feedforward neural network with long-term memory," *Proc. Int. Joint Conf. on Neural Networks*, 1989/1994 (2001)

[10] J. Platt: "A resource allocating network for function interpolation," *Neural Computation*, **3**, 213/225 (1991)

[11] T. Poggio and F. Girosi: "Networks for approximation and learning," *IEEE Trans. on Neural Networks*, **78**, 9, 1481/1497 (1990)

[12] M. J. L. Orr: "Introduction to radial basis function networks," *Technical Report of Institute for Adaptive and Neural Computation*, Division of Informatics, Edinburgh University (1996)