A Memory-based Neural Network Model for Efficient Adaptation to Dynamic Environments

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

***Abstract*— When environments are dynamically varied for agents, the knowledge acquired from an environment would be useless in the future environments. Thus, agents should be able to not only acquire new knowledge but also modify old knowledge in learning. However, modifying all acquired knowledge is not always efficient. Because the knowledge once acquired may be useful again when the same (or similar) environment reappears. Moreover, some of the knowledge can be shared among different environments. To learn efficiently in such a situation, we propose a neural network model that consists of the following four modules: resource allocating network, long-term memory, association buffer, and environmental change detector. We apply this model to a simple dynamic environment in which several target functions to be approximated are varied in turn.**

I. Introduction

A most distinctive feature of human is to learn continuously from many experiences in the lifetime. Once a concept is acquired, it is not only accumulated in memory but also used to generalize other concepts from extremely few examples [1]. In other words, we can transfer the knowledge about a concept into other concepts to be formed through the so-called lifelong learning. To allow machines to imitate such a human ability, we should incorporate at least the following abilities:

1) the ability to represent many experiences in an efficient form of knowledge,
2) the ability to generalize the knowledge to cope with unknown situations,
3) the ability to learn new knowledge incrementally without unexpected forgetting,
4) the ability to discriminate different tasks,
5) the ability to utilize proper knowledge for handing a task experienced before,
6) the ability to extract sharable knowledge from different tasks and to transfer it into unknown tasks.

The first ability gives an efficient way to memorize the huge number of knowledge those are acquired through lifelong learning. The second ability is needed to make proper decision even from not so many experiences. To realize these two abilities, neural networks are often used. In neural networks, however, knowledge is dispersively stored in their connection weights. Hence, when a new training sample is given, 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’ [2] and it makes difficult to realize the third ability. To solve the problem of catastrophic interference, there have been proposed several approaches [2], [3], [4], [5]. 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) [5]. Although RAN-LTM is one of memory-based neural networks, it does not need so much memory capacity and it realizes robust incremental learning ability. In this sense, RAN-LTM has the first three abilities.

The fourth and fifth abilities are needed to realize the autonomous agents that can learn efficiently in multi-task environments. For this purpose, we have proposed *Resource Allocating Network with Associative Long-Term Memory* (RAN-ALTM) [6], in which two extra modules are added to RAN-LTM. This model was applied to the learning in a class of dynamical environments where a stationary environment continues for a while and then it changes to another environment. In this problem, a task (e.g., function approximation, pattern recognition) is given from each stationary environment.

The last ability realizes so-called *knowledge transfer* to facilitate the learning of future tasks. To realize this ability, sharable knowledge among different tasks should be identified from all knowledge acquired before, and then it should be utilized as a bias of the subsequent learning. In this paper, we propose an extended RAN-ALTM model in which such a knowledge transfer mechanism is incorporated into the original RAN-ALTM model. More concretely, an additional information called ‘invariability parameter’ is introduced into each memory item to measure the retrieval frequency of the memory item. Then the memory items with large invariability are used to determine the initial structure of a neural network for the subsequent environment.

II. A Memory-based Neural Network Model

A. Assumption for Dynamic Environment

In general, there are various definitions of dynamic environments depending on what sensory signals are processed by agents; that is, the time scales of the environmental changes could be ranged from seconds to lifetime. Therefore, when we discuss a learning mechanism of agents, first we should define the relation between the time scales of the agents’ adaptation and the environmental changes.

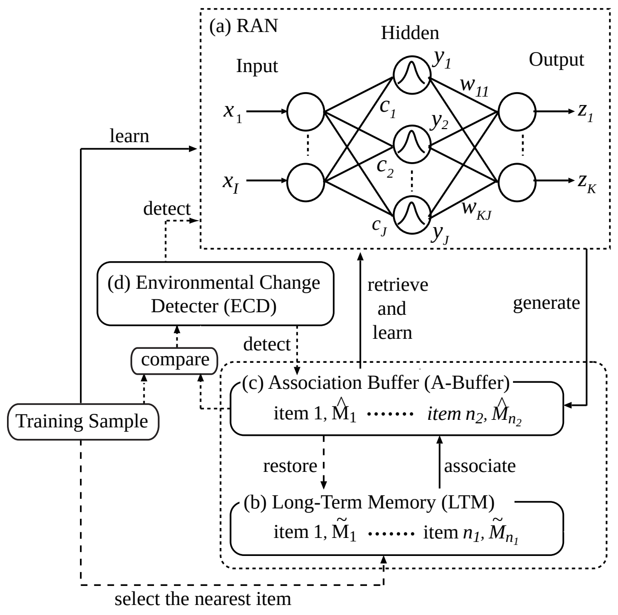

(a) RAN
Hidden
Input
Output
x_1
x_I
y_1
y_2
y_J
c_1
c_2
c_J
w_{11}
w_{KJ}
z_1
z_K
learn
detect
(d) Environmental Change Detecter (ECD)
retrieve and learn
generate
compare
detect
Training Sample
(c) Association Buffer (A-Buffer)
item 1, $\hat{M}_1$ ······ item n_2, $\hat{M}_{n_2}$
restore
associate
(b) Long-Term Memory (LTM)
item 1, $\tilde{M}_1$ ······ item n_1, $\tilde{M}_{n_1}$
select the nearest item

Fig. 1. Structure of RAN-ALTM.

In this paper, we make an assumption that the time constant of the agents' adaptation is small enough as compared with the time constants of the environmental changes. This assumption leads to a class of dynamic environments where a stationary environment continues for a while and then it changes to another stationary environment. Under this condition, we can assume that the duration of a stationary environment is long enough to be learned.

Another assumption we make here is that agents should learn continuously from successively appeared stationary environments, and that they may encounter the environments experienced before. That is to say, the proposed learning algorithm should possess the properties of *lifelong learning*.

B. Resource Allocating Network with Associative Long-Term Memory

In order to adapt efficiently under this class of dynamic environments, we have proposed a memory-based neural network model called *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 role of each module.

1) Resource allocating network (RAN): RAN [7] is an extended version of Radial-Basis Function (RBF) network [8]. 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 z_k are calculated as follows:

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

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

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

After the calculations in Eqs. (1)-(2), a mean squared error E between the outputs $\boldsymbol{z}$ and the targets $\boldsymbol{T}$ is evaluated. Then, either of the following operations are selected based on E and the activations of hidden units: the allocation of a new hidden unit and the modification of connection weights and RBF centers. When the latter operation is selected, some memory items are retrieved from A-Buffer in order to learn with the training sample.

2) Long-Term Memory (LTM): LTM is a place to store memory items $\{\tilde{M}_1, \cdots, \tilde{M}_{n_1}\}$, which are extracted through the learning and utilized for the following purposes:

1) suppressing the forgetting caused by incremental learning,
2) adapting to the current environment quickly by retrieving useful ones acquired in the past.

To realize the above functions, memory items are composed of not only an input-output pair but also 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) Hessian Information, $H(\tilde{\boldsymbol{x}}_j)$
The complexity 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 such a domain. Therefore, we quantify the functional complexity by the Hessian. If this information is large, the corresponding memory item should be recalled with high probability.

iii) Linkage Information Table (LIT)
When a memory item $\tilde{M}_k$ ($k \neq j$) is generated in the same environment as $\tilde{M}_j$, $\tilde{M}_k$ is added to the LIT of $\tilde{M}_j$, and the relevance parameter Q_{jk} is also registered. If Q_{jk} is large, it means that $\tilde{M}_j$ has strong relevance to $\tilde{M}_k$ and it should be 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. This value corresponds to the term of validity. 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 returned 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).

Memory items are generated in the region where the outputs are accurately approximated. In order to prevent memory items from increasing too much, it is generated only when the distance from the nearest memory item is larger than a constant value. When the above two conditions are satisfied, the RBF center $\boldsymbol{c}_j^*$ of the most activated hidden unit is given to RAN, and the outputs $\boldsymbol{z}$ are calculated. This input-output pair $(\boldsymbol{c}_j^*, \boldsymbol{z})$ is generated as a new memory item (see *Generation of Memory Items* in Subsection II-D).

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_{jk} 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 $F_k = 1$ (see *Association of Memory Items*). Through this association mechanism, several memory items are moved to A-Buffer from a single training sample. The correct retrieval of these memory items is the key mechanism to attain the fast adaptation in RAN-ALTM.

3) Association Buffer (A-Buffer): A-Buffer is the place where the memory items generated in the same environment are temporarily stored. To suppress the forgetting caused by incremental learning, memory items in A-Buffer $\{\hat{M}_1, \cdots, \hat{M}_{n_2}\}$ are retrieved to train in RAN. Since it is not efficient to use all the memory items at every learning step, the retrieved memory items are restricted to suppress the interference effectively (see *Retrieval of Memory Items*).

Although the association is done from a memory item based on its relevance parameter Q_{jk}, the associated items could belong to different environments each other. This can happen when the memory item is generated around an intersection point of the input-output functions for two different environments. Therefore, the association for such a misleading memory item should be prohibited. To find such a misleading memory item, check the inconsistency among all memory items in A-Buffer and set the association flag F_j set to 1 (see Step 5 of *Learning Algorithm*).

4) Environmental Change Detector (ECD): Since memory items stored in A-Buffer were generated in the same environment, the training sample has the inconsistency to the memory items when the environment changes. Therefore, the environmental changes can be detected by checking the inconsistency between the current training sample and the most similar memory items in A-Buffer.

To realize such a mechanism, we introduce a confidence factor for environmental changes, and this inconsistency is accumulated into the confidence factor (see *Detection of Dynamical Changes*). When this factor becomes larger than a predetermined threshold value, the detection signal from ECD is propagated to RAN and A-Buffer, and then these two modules are initialized (see *Initialization*).

C. Incorporation of Knowledge Transfer Mechanism to RAN-ALTM

In the original RAN-ALTM model, when an environmental change is detected, all memory items stored in A-Buffer is cleared regardless of their usefulness in the subsequent stationary environment. Needless to say, this initialization is wasteful if some of the memory items are also useful in the next environment.

To avoid this inefficiency, we should predict the usefulness of memory items in the subsequent environment. Unfortunately, this prediction cannot be done completely in general. In many actual situations, however, we can expect that there are some sharable knowledge among stationary environments (tasks) as described in Section I. Therefore, the expected usefulness of a memory item can be estimated by the probability of retrieving the memory items over the recent environments.

To evaluate this probability, we introduce an invariability parameter Z_j as an additional element of the jth memory item $\tilde{M}_j$. Then, when an environmental change is detected, Z_j of memory items in A-Buffer are increased while those in LTM are decreased as follows:

$$[\mathrm{LTM}] \qquad Z_j \leftarrow \max[Z_j - 1, 0] \tag{3}$$

$$[\mathrm{A-Buffer}] \qquad Z_j \leftarrow \min[Z_j + 1, \theta_z] \tag{4}$$

where θ_z is a maximum value of Z_j.

Unless the subsequent environment happens to be completely changed, it is expected that memory items with large Z_j are also useful in the new environment. Let the set of these memory items be Ω^I. The memory items $\tilde{M}_j$ $(j \in \Omega^I)$ are left in A-Buffer after the detection of an environmental change, and then the input information $\tilde{\boldsymbol{x}}_j$ of $\tilde{M}_j$ are utilized for determining the initial structure of RAN: that is, the RBF centers are initialized by $\tilde{\boldsymbol{x}}_j$ (see Step 4 in *Initialization*). This operation incorporates a knowledge transfer mechanism into RAN-ALTM that enhances the adaptability to the new environment.

D. A New Learning Algorithm of RAN-ALTM

Now that we can show a new learning algorithm of RAN-ALTM incorporating a knowledge transfer mechanism.

[Learning Algorithm]

(1) When a training sample $(\boldsymbol{x}, \boldsymbol{T})$ is given, execute *Detection of Dynamical Changes*.
(2) If an environmental change is detected, execute *Initialization*. Else if the vigilance mode is detected, go back to Step 1 (i.e., skip the training process). Otherwise, go to Step 3.
(3) Calculate the outputs $\boldsymbol{z}$ from Eqs. (1)-(2) in RAN.
(4) Execute *Association of Memory Items*.

(5) Find the inconsistency among all memory items in A-Buffer. For all inconsistent memory items, set the association flag F_j to 1.
(6) Calculate the mean square error E between the outputs $\boldsymbol{z}$ and the targets $\boldsymbol{T}$.
(7) If $E > \varepsilon$ and $\|\boldsymbol{x} - \boldsymbol{c}^*\| > \delta(t)$, then a hidden unit is added (i.e., $J \leftarrow J+1$). The centre vector $\boldsymbol{c}_J$, connection weight w_{kJ}, and variance σ_J^2 are initialized as follows: $\boldsymbol{c}_J = \boldsymbol{x}$, $\boldsymbol{w}_k = \boldsymbol{T} - \boldsymbol{z}$, and $\sigma_J = \kappa\|\boldsymbol{x} - \boldsymbol{c}^*\|$, where κ is a positive constant and $\delta(t)$ is decreased with time t. Otherwise, the following procedure is carried out:
    a) Execute *Retrieval of Memory Items*.
    b) For the training sample and all the retrieved memory items, calculate RAN's outputs and the mean square error E.
    c) Update the weight connections, RBF centers, and biases based on the learning algorithm of RAN [7].
(8) Execute *Generation of Memory Items*.
(9) For all memory items in A-Buffer, decrease the validity information V_j by a constant value.
(10) If V_j becomes zero, return the corresponding memory item $\tilde{M}_j$ to LTM.
(11) Go back to Step 1.

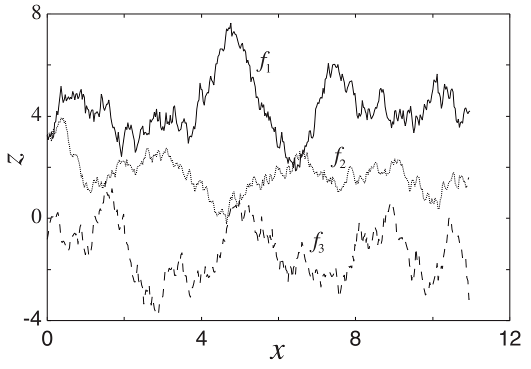
[Detection of Dynamical Changes]
(1) Find the most similar memory item in A-Buffer, $\hat{M}^*$: $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{z}}^*)$, to the training sample $(\boldsymbol{x}, \boldsymbol{T})$.
(2) Update the confidence B for environmental changes as follows:
i) If $\|\boldsymbol{x} - \hat{\boldsymbol{x}}^*\| < k_1/g'(\hat{\boldsymbol{x}}^*)$,
$$B^{NEW} = \|\boldsymbol{T} - \hat{\boldsymbol{z}}^*\| + k_2 B^{OLD}.$$
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 $k_3\sigma_z \le B < k_4\sigma_z$, then enter the vigilance mode. Else if $B \ge k_4\sigma_z$, then propagate the detection signal to RAN and A-Buffer. Here, σ_z is the variance of $\hat{\boldsymbol{z}}_j$, and k_3 and k_4 are positive constants.

[Initialization]
(1) Initialize all hidden units and weight connections in RAN.
(2) Update the invariability parameter Z_j as shown in Eqs. (3)-(4).
(3) Return the memory items to LTM whose invariability parameter Z_j is less than θ_z.
(4) For each memory item $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j)$ in A-Buffer, execute the same procedure of allocating hidden units in RAN (see Step 6 in *Learning Algorithm*).

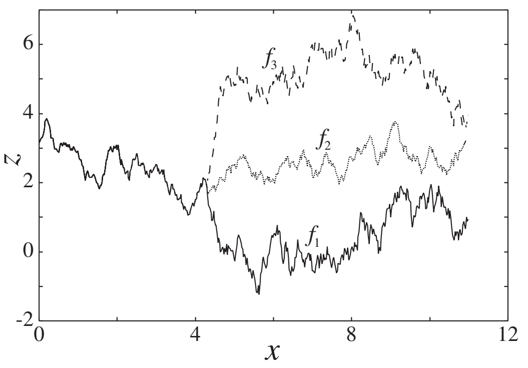
[Association of Memory Items]
(1) Search for the most similar memory item $\tilde{M}_j$ to the training sample $(\boldsymbol{x}, \boldsymbol{T})$, and then move it to A-Buffer. Set the validity information V_j of $\tilde{M}_j$ to infinity.
(2) Increase the relevance parameter Q_{jk} in LIT of $\tilde{M}_j$ to the recently generated / retrieved memory items $\tilde{M}_k$.
(3) If the association flag F_j is zero, go to Step 4. Otherwise, terminate this procedure.
(4) Refer to LIT of $\tilde{M}_j$, and move the memory items $\tilde{M}_k$ with the m_1 largest Q_{jk} to A-Buffer. Set the validity information V_k of $\tilde{M}_k$ to a constant value.
(5) For each $\tilde{M}_k$, carry out the same operation in Step 4 (i.e., move m_2 memory items to A-Buffer).

[Retrieval of Memory Items]
(1) For the jth hidden unit y_j, find a memory item $\tilde{M}_j$: $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j, H(\tilde{\boldsymbol{x}}_j))$ 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.
(2) 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.
(3) If the validity information V_j of $\tilde{M}_j$ is infinity, put the input-output pair $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j)$ into a retrieval candidate set Ω with probability P_j. Otherwise, check the consistency of $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j)$ for the input-output relation of RAN. If it is consistent, put $(\tilde{\boldsymbol{x}}_j, \tilde{\boldsymbol{z}}_j)$ into Ω with probability P_j.

[Generation of Memory Items]
(1) If all hidden outputs y_j are less than a threshold value θ_c, then terminate this procedure. Otherwise, go to Step 2.
(2) 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.
(3) If $r_j > \beta$ for $j \in \mathcal{I}_1$, then initialize r_j and go to Step 4. Otherwise, terminate this procedure. Here, β is a positive constant.
(4) Calculate the minimum distance between the jth center $\boldsymbol{c}_j$ and memory items $\tilde{\boldsymbol{x}}_m$ $(m = 1, \cdots, n_2)$ in A-Buffer 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 5. Otherwise, terminate this procedure.
(5) 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



(a) Experiment 1



(b) Experiment 2

Fig. 2. Two sets of one-dimensional functions, each of which corresponds to a stationary environment.

as its input, and obtain the output $\boldsymbol{z}$. Calculate the determinant of Hessian matrix $H(\boldsymbol{c}_j)$ whose element (i, i') is given as follows:

$$\frac{\partial^2 \tilde{z}_k}{\partial c_{ji} \partial c_{ji'}} = \sum_{l=1}^{J} \frac{w_{kl}}{\sigma_j^4}(c_{ji} - c_{li})(c_{ji'} - c_{li'}) y_l$$

(6) Increase the relevant parameter Q_{jk} by a constant value for the recently generated or retrieved memory items $\tilde{M}_k$, and set the others to zero.
(7) Set V_j to infinity, and set F_j and Z_j to zero.
(8) Generate the following seven-fold information $(\boldsymbol{c}_j, \boldsymbol{z}, H(\boldsymbol{c}_j), Q_{jk}, V_j, F_j, Z_j)$ as a new item.

III. Experiments

A. Experimental Setup

In the defined class of dynamic environments, a stationary environment continues for a while and then it changes to another stationary environment in turn. To evaluate the adaptability, let us consider a simple problem where a stationary environment is represented by a one-dimensional function.

Figures 2(a)(b) are the target functions to be approximated. These three one-dimensional functions $f_1 \sim f_3$ are temporally interchanged. That is, each function corresponds to the desired input-output relation for an agent 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 function continues for a while, and then the target function is changed to another one (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.

As you can see from Fig. 2(b), the target functions in the second set have a region where all functions have the same outputs z. Therefore, training samples drawn from this region are unchanged even if the environmental changes occur. This means that memory items generated in this region could be shared in all environments. Thus, if these items are transferred to the learning of the subsequent environment, faster adaptation must be realized.

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 past knowledge when the corresponding environment reappears,
(4) whether the fast adaptation is realized by extracting sharable memory items and transferring them to the subsequent learning.

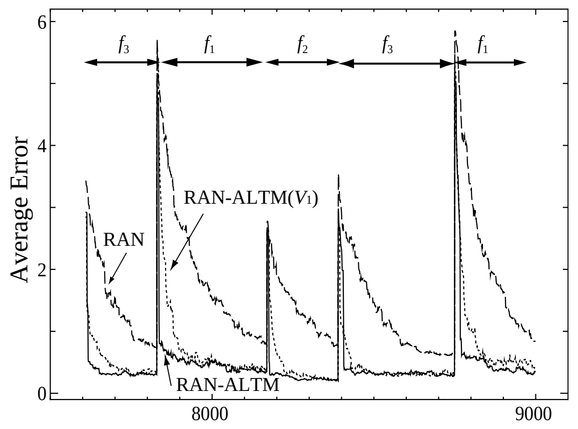
To verify the above four capabilities in RAN-ALTM, we adopt two variants of RAN-ALTM: the one in which both the functions of association and knowledge transfer are left out, and the other in which only the function of knowledge transfer is removed. For notational convenience, these variants are noted as RAN-ALTM(V_1) and RAN-ALTM(V_2), respectively.

B. Experimental Results

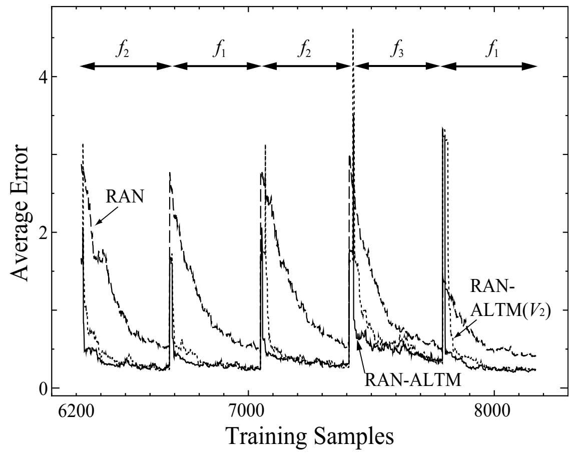
The evaluation is conducted for a series of 30 stationary environments, each of which corresponds to one of the above three target functions $f_1 \sim f_3$. Since the order of the transitional environments and the duration are randomly determined, we assume that agents do not know what environments are presented next.

Figures 3 (a)(b) show the time courses of the average errors for a fixed set of test samples. Note that these time courses are illustrated as a typical learning process in the whole learning period, and that all environments shown here have been experienced at least once by an agent. The arrows depicted in the upper part of Figs. 3 (a)(b) show the terms of each stationary environment. For comparative purposes, the learning processes for the original RAN are also demonstrated, in which only adaptation to the training samples is conducted without detecting any environmental changes.

As you can see from Figs. 3 (a)(b), although the error suddenly increases after environmental changes, the error decreases immediately in RAN-ALTM and its variant models. On the other hand, the error of RAN is not decreased so rapidly. From these results, we can say that environmental changes are correctly detected by ECD and the initialization triggered by



(a) Experiment 1



(b) Experiment 2

Fig. 3. Time courses of average errors for two different dynamic environments.

the detection enhances the adaptation speed. Furthermore, after the environmental changes, the output errors almost always decrease monotonously. This result suggests that RAN-ALTM and its variant models can learn stably even in incremental settings under dynamic environments.

In Fig. 3 (a), we see that RAN-ALTM can adapt quickly to the reappeared environments as compared with RAN-ALTM(V_1), in which no association function is introduced. This result suggests that RAN-ALTM can accumulate the knowledge of the experienced environments and can utilize it for the later learning through the association function. Furthermore, from the result in Fig. 3 (b), we can say that the knowledge transfer mechanism works well in RAN-ALTM because it can adapt quickly to the reappeared environments as compared with RAN-ALTM(V_2).

To verify such results more precisely, the average numbers of training samples to detect environmental changes and to reach the learning convergence after the detection are investigated over the whole training period (i.e., the series of 30 transitional environments). Table I shows these results. As seen from Table I, although there are little differences among RAN-ALTM and the variant models in the detection performance, RAN-ALTM achieves the best convergence in both two experiments. Especially, we can see that the convergence is quite fast in Experiment 2 where there are knowledge to be shared among different stationary environments.

TABLE I
THE AVERAGE NUMBERS OF TRAINING SAMPLES NEEDED FOR (A) THE DETECTION OF ENVIRONMENTAL CHANGES AND (B) THE LEARNING CONVERGENCE.

(a)

	RAN-ALTM(V_1)	RAN-ALTM(V_2)	RAN-ALTM
Exp. 1	6.6	6.8	6.8
Exp. 2	17.0	15.8	15.8

(b)

	RAN-ALTM(V_1)	RAN-ALTM(V_2)	RAN-ALTM
Exp. 1	148.3	102.5	101.1
Exp. 2	196.8	192.0	134.8

IV. Conclusion

In this paper, we proposed an extended version of Resource Allocating Network with Associative Long-Term Memory (RAN-ALTM), in which the knowledge transfer mechanism was augmented from the previous model. 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 we assume that the duration of an environment is long enough to be learned but it is unknown for agents, stationary environments appear repeatedly over a long period of time, and some environments may partially share the same input-output relations.

To examine the adaptation performance of RAN-ALTM, we defined a simple form of dynamic environments; i.e., three one-dimensional target functions were temporally interchanged. From the experimental results, we verified that the proposed RAN-ALTM has the excellent capabilities shown in Section I.

References

[1] S. Thrun and L. Pratt, *Learning to learn*, Kluwer Academic Pub. (1998)
[2] 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)
[3] 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)
[4] 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)
[5] 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)
[6] K. Tsumori and S. Ozawa, "Incremental learning in dynamic environments using neural network with long-term memory," *Proc. Int. Conf. on Neural Networks 2003*, 2583/2588 (2003)
[7] J. Platt, "A resource allocating network for function interpolation," *Neural Computation*, **3**, 213/225 (1991)
[8] T. Poggio and F. Girosi, "Networks for approximation and learning," *IEEE Trans. on Neural Networks*, **78**, 9, 1481/1497 (1990)