MFM: Multiple Forward Model Architecture for Sequence Processing *

Raju S. Bapi† **Kenji Doya**
Computational Neurobiology Group
Kawato Dynamic Brain Project, ERATO, JST
2-2 Hikaridai, Seika, Soraku, Kyoto 619-0288, Japan
email: {rajubapi, doya}@erato.atr.co.jp

Abstract

A multiple forward model (MFM) architecture is proposed for sequence identification, learning and production. MFM is inspired by Wolpert and Kawato's [1998] multiple paired forward and inverse models architecture for motor control. In particular, learning of sequence-specific modules and switching among multiple sequences are demonstrated. Appropriate sequence modules are chosen and maintained on the basis of feed-forward prediction errors and softmax responsibility estimation. Simulations are conducted in order to reproduce the findings from Tanji and Shima's [1994] sequence experiments on monkeys. Simulations demonstrate that the network is capable of learning four simple three-step sequences in a supervised fashion initially using external cues and eventually reproducing them without external cues as in the monkey experiments. Preliminary hypotheses are made about how various components of the network mimic the overall function of the cortex and the basal ganglia in mammals. Finally, general implications of this architecture for hierarchical sequence processing are discussed.

1 Introduction

Serial order is an important aspect of human and animal behavior and continues to receive attention. In spite of research over many years, it is not yet completely understood how the brain solves the serial order of motor behavior problem [Rosenbaum, 1991]. However, in recent years there has been rapid progress in the domain of sequence processsing both in the neuroscience literature (electrophysiology on animals and imaging studies on humans) and in computational learning literature.

In the sequence learning literature, various researchers used neural networks (For example, [Bapi and Doya, 1998], [Dominey, 1995], [Elman, 1990], etc.) to learn multiple sequences. Problems with such approaches are long training times and catastrophic forgetting. Recently, Wolpert and Kawato [1998] proposed multiple paired forward-inverse models architecture for human motor learning and successfully applied it to the problem of manipulation of multiple objects [Haruno *et al.*, 1998]. They argued that such paired models enable learning and retrieval of the appropriate models based on environmental context and facilitate smooth interpolation for generalization. These models also avoid the catastrophic forgetting problem since simulations demonstrate that the model switching is robust [Haruno *et al.*, 1998]. In the current work we extend this approach to sequence learning. We propose a multiple forward model (MFM) architecture for sequence processing. The key idea is that sequence learning and switching is based on prediction errors. After introducing the details of the proposed architecture, we will review the experimental findings of Tanji and Shima [1994] on monkeys. The aim of the current work is to take these experimental results as an example case and test the proposed MFM architecture.

2 MFM Architecture

A forward model is defined as one that predicts the response output $y(t+1)$, given the inputs $x(t)$ to the system. In the sequence processing context, a forward model predicts the next action $y(t+1)$, given the previous response $y(t)$. Figure 1 shows the MFM architecture.

In the following we describe the processing steps in order. Firstly, the network is presented with the the previous response $y(t)$. Each of the *Sequence Modules* predicts the next action $y^k(t+1)$ based on the previous response. These predictions are gated by the responsibility factors λ^k in the *Responsibility Gating* module to generate a predicted response. The *Response Selection* module combines the predicted response with the input to generate the actual action output $y(t+1)$. Errors resulting from the comparison of the desired target to the predicted response of each module are sent to the *Prediction Error* module. *ResponsibilityEstimation* module transforms the prediction errors into responsibility fac-

*To be presented at the IJCAI'99 workshop on Sequence Learning, Stockholm, Sweden, July 31–Aug 6, 1999

†Current Address: Dept. of Computer/Information Sciences, University of Hyderabad, Hyderabad, A.P., India 500 046

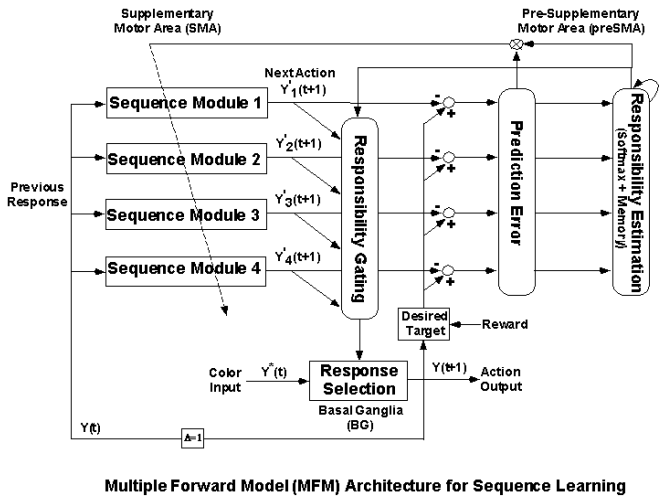

Figure 1: Neural Network Architecture

tors λ^k. λ^k signify the relative appropriateness of modules in the current context and drive hebbian learning in the sequence modules. The above processing steps are repeated till the sequence switching becomes stable. We now present the detailed equations.

2.1 Sequence Module Output

Each sequence module k predicts the next action $\hat{y}^k(t+1)$ based on the previous response $y(t)$.

$$\hat{y}^k_j(t+1) = f\left(\sum_i w^k_{ji}(t) y_i(t)\right) \quad (1)$$

2.2 Response Selection

The predicted response is used in two ways. Firstly, it is used to generate the actual action output $y_j(t+1)$. Predicted response from each sequence module $\hat{y}^k(t)$ is gated by the corresponding responsibility factor $\lambda^k(t)$ and the final output is chosen by $g(\cdot)$, either the maximum function or a suitable stochastic function.

$$y_j(t+1) = g\left(\sum_{k=1} \lambda^k(t)\hat{y}^k_j(t)\right) \quad (2)$$

2.3 Responsibility Estimation

The second use of the predicted responses is to calculate the responsibility factors.

Prediction Error

The prediction error $E^k(t)$ of each module is calculated based on the predicted response $\hat{y}^k_j(t)$ and the desired target $y^{\star\star}_j(t)$.

$$error^k_j(t) = (y^{\star\star}_j(t) - \hat{y}^k_j(t)) \quad (3)$$

$$E^k(t) = \frac{1}{2}\sum_j \left(error^k_j(t)\right)^2 \quad (4)$$

Responsibility

We assume that the probability $p^k(t)$ of the k^{th} module being the appropriate predictor is proportional to the exponential of the error $E^k(t)$ with a scaling factor $\sigma(t)$.

$$p^k(t) = \epsilon e^{\frac{-E^k(t)}{\sigma(t)^2}} \quad (5)$$

The responsibility estimation module keeps a memory of the previous $p^k(t-\delta)$ values, where $\delta = 0,...,\Delta$. Responsibility factor $\lambda^k(t)$ is then calculated by normalizing the product of all the probabilities.

$$\lambda^k(t) = \frac{\prod_{\delta=0}^{\Delta} p^k(t-\delta)}{\sum_i \prod_{\delta=0}^{\Delta} p^i(t-\delta)} \quad (6)$$

The combination of using the exponential function and normalization amounts to using a softmax operation in the responsibility estimation. The use of past probability values ensures that modules are not switched around wildly.

2.4 Learning in the Sequence Modules

Hebbian learning in the sequence modules is gated by the responsibility factor. Thus at any time, the module that predicts the best learns the most. Although in the current formulation the forward models work as one-step-ahead predictors, they can be designed as more general predictors with longer sequential context.

$$w^k_{ji}(t+1) = w^k_{ji}(t) + \eta\lambda^k error^k_j(t) \cdot \hat{y}^k_j(t)(1-\hat{y}^k_j(t))y_i(t-1) \quad (7)$$

In the next section we present the details of the experiments of Tanji and Shima on monkeys. The specific equations used to reproduce the results of Tanji and Shima are given in the Appendix. The subsequent section presents the simulation results.

3 Studies on Monkeys

Tanji and colleagues devised a sequence learning paradigm and made single cell electrode recordings in behaving monkeys while they performed sequential arm movements. The recordings have been made in various areas in the anteriror part of the neocortex, including the supplementary motor area (SMA), the presupplementary motor area (preSMA), the premotor cortex (PM), and the motor cortex (MI). All these areas are hypothesized to be participating in the learning and/or execution of the external-input-driven and/or internally-generated sequential motor acts [Tanji, 1994]. After describing the experimental setup, a summary of results from various experiments will be given below.

3.1 Experimental Details

Monkeys were initially trained to respond with a push, pull or turn movement on a handle when presented with a red, yellow, or green color light, respectively, on a display panel. After establishing the color-to-action mapping, they were required to sequence these elementary actions by observing a succession of color lights and recalling the appropriate movement at every step. In this manuscript we will refer to the four sequences as I:123, II:132, III:321, and IV:312. The trials when color lights preceded movements were called, 'Visual Trials (VT)' and the other trials when there were no colored lights were called 'Memory Trials (MT)'. In a block of trials, monkeys experienced 5 VT followed by 6 MT. They were taught four such sequences till they achieved 95% success rate on each sequence. After this stage, single cell recordings were made while monkeys performed several blocks of trials. Each block used a different sequence and the order of appearance of different sequences varied semi-randomly. The end of a block is signaled by random flashing of lights which also indicated a change in the sequence.

3.2 Experimental Results

In the present study, we focus on the results of single unit recordings in SMA and PreSMA. SMA recordings [Tanji and Shima, 1994] revealed a significant activation exhibiting two main properties: i) *sequence-specific activity*: activity of some cells preceded specifically before the performance of a particular sequence (say, Turn-Pull-Push) and ii) *transition-specific activity*: some cells were tonically active after performing a specific movement and before initiating a particular next movement (say, Push followed by Pull). In contrast, the activity of majority of cells in the MI was selective to single movements but not sequence-specific. While SMA cells were mainly active in the MT phase, cells in PM were active only during the VT phase and cells in MI were active during all phases of the task. Some cells in preSMA were active only during the very first VT, possibly signifying the start of a new sequence and thus specifying the need for a change in motor plan [Shima *et al.*, 1996]. In this article learning of sequence and transition specificity is demonstrated using MFM architecture.

4 Simulation Results

Detailed equations used in the simulations are shown in the Appendix. Each sequence is presented to the network successively for 5 blocks. All the four sequences are shown to the network, the order of presentation being randomized. Each block consists of 5 VT followed by 6 MT as in the monkey experiments. During the VT, the network has access to the desired target and hence the learning progresses in a supervised fashion. In the current simulations, during the VT period we assume that the appropriate (colored) light-to-movement mapping has already been established and thus the network performs at 100% accuracy. However, during the MT the desired output is not available and is calculated based on the reward. For example, if the predicted response is incorrect, then the desired target is set as the opposite of the actual response (for example, 110 for 001). Thus the learning during the MT progresses in a semi-supervised fashion.

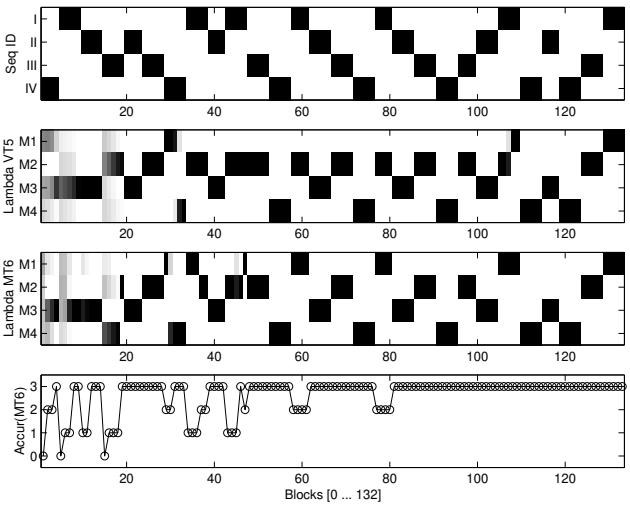

Figure 2: Results for all the blocks are shown. Legend: Modules – M1 : Module 1; M2 : Module 2; M3 : Module 3; M4 : Module 4; Sequence IDs – I : 123, II : 132, III : 321, IV : 312

4.1 Long Term Behavior of the Network

Simulation results for 132 blocks are shown in Figure 2. The top row shows the randomized presentation of all the four sequences during the simulations. The second and third rows show how the responsibility factor λ varies as sequences are learnt during the last visual and memory trials, denoted as λ-VT5 and λ-MT6 respectively. The bottom row shows the number of steps performed correctly (performance accuracy) in the last memory trail (MT6) as simulations progressed. It can be observed that as the switching became stable, the performance accuracy also improved. For example, although Module 3 initially captured both Sequence I (123) and Sequence II (132), specialization for Sequence II (132) developed by 20th block and stayed stable since then. More details are shown in later figures below.

For clarity, various snapshots during the intermediate blocks are shown in Figures 3, 4 and 5. Figure 3 shows that the performance in the early blocks is erratic and that Module 3 is active during the presentation of Sequences IV, I, and II in the 6th memory trial (see the graph on the third row – λ-MT6).

By the 30th block in Figure 4, we see that Sequences II and III are represented by Modules 3 and 2, respectively and that for Sequences IV and I switching is not yet stable. However, by the end of the simulation the

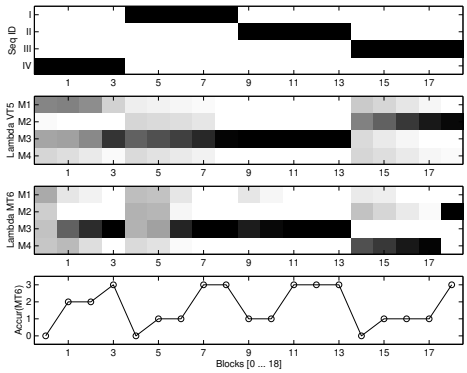

Figure 3: Results from blocks 0-18 are shown. Legends as in Figure 2.

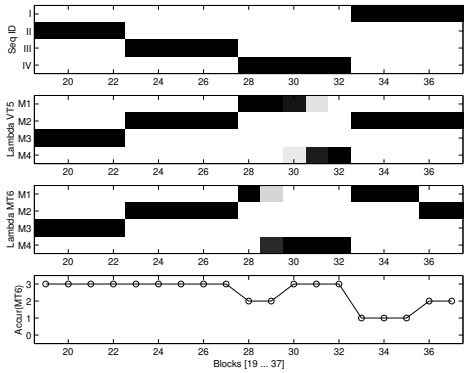

Figure 4: Results of Blocks 19-37 are shown. Legends as in Figure 2.

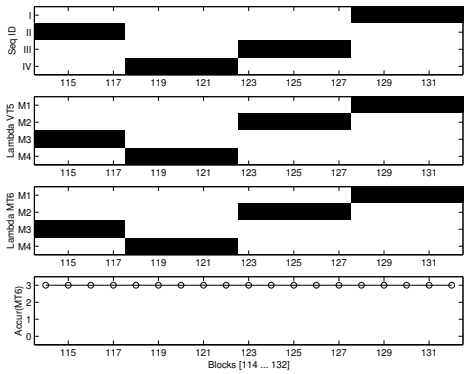

Figure 5: Results of Blocks 114-132 are shown. Legends as in Figure 2.

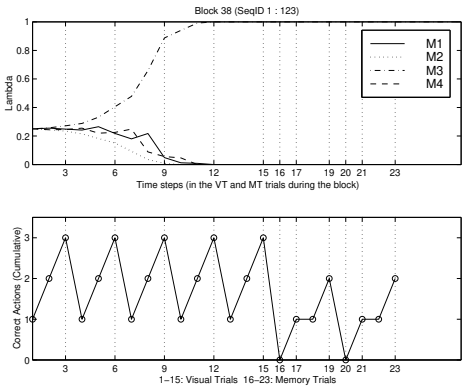

Figure 6: Detailed Results of 38th block are shown.

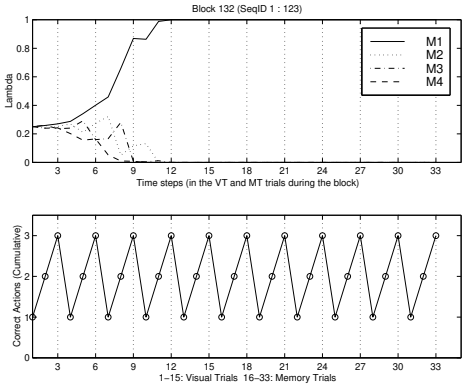

Figure 7: Detailed Results of 132nd block are shown.

switching became stable and specialized (SeqI:M1, SeqII:M3, SeqIII:M2, SeqIV:M4) as shown in Figure 5.

4.2 Short Term Behavior of the Network

Short term behavior of the network during single blocks (Block 38 and 132) is shown in Figures 6 and 7. Sequence I is presented to the network during both these blocks. During the early stages the network did not arrive at a stable switching solution for Sequence I. As shown in Figure 6, the network locked into an incorrect module (Module number 3 which represents Sequence II) by the end of the 5th VT and as a consequence there are significant number of errors of recall during the subsequent 6 MTs. In comparison, after extensive training (by block 132 as shown in Figure 7), Module 1 specialized in Sequence I and the recall was accurate during all of the MTs.

4.3 Discussion of Results

Thus the learning of sequence specificity and stable switching among learned sequences is achieved by this architecture. Each module learns the sequence transitions appropriate to the sequence that the module specializes in. For example, Module 1 learns the transitions

1-to-2 and 2-to-3. In the current formulation, sequence specificity in the modules arises as a result of learning the appropriate transitions.

Although sequence and transition specificity are demonstrated, detailed comparison between the results of monkey experiments and the simulations is left as future work.

5 General Discussion

Here we put forward some preliminary hypotheses about the brain areas whose functionality is captured in the MFM architecture (also indicated in Figure 1): *SequenceModules* mimic the function of the SMA, the *ResponsibilityEstimation* may take place in the preSMA. One of the roles attributed to the basal ganglia is context dependent response selection [Houk *et al.*, 1995] and this aspect is attributed to the *ResponseSelection* module in the proposed architecture.

In the current simulations, complex sequence learning, where the individual elements of a sequence may be repeating, is not demonstrated. A more general approach would be to use recurrent networks for sequence modules in place of purely feed-forward networks. Simulations of the general version are in progress.

Although the current architecture consists only of one level, the concept can be extended to multiple levels of hierarchy. In such a hierarchical system, module switching at any level will be affected not only by the the responsibility esitmations at the current level but also by the values at the other levels. In one such formulation, a network can be set up such that the sequential context encoded at each level increases as one moves up the hierarchy. Hierarchical networks are of immense use in the domains of natural language processing and sequential planning.

References

[Bapi and Doya, 1998] R. S. Bapi and K. Doya. A sequence learning architecture based on cortico-basal ganglionic loops and reinforcement learning. In *Proceedings of the 5th International Conference on Neural Information Processing*, 1:260-263, 1998.

[Dominey, 1995] P. F. Dominey. Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning. *Biological Cybernetics*, 73:265–274, 1995.

[Elman, 1990] J. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

[Haruno *et al.*, 1998] M. Haruno, D. M. Wolpert, and M. Kawato. Multiple paired forward-inverse models for human motor learning and control. *NIPS*98*, vol 11 (in press).

[Houk *et al.*, 1995] J. C. Houk, J. L. Davis, and D. G. Beiser, editors. *Models of information processing in the basal ganglia*. MIT Press, Cambridge, MA, 1995.

[Rosenbaum, 1991] D. A. Rosenbaum. *Human Motor Control*. Academic Press, San Diego, CA, 1991.

[Shima *et al.*, 1996] K. Shima, H. Mushiake, N. Saito, and J. Tanji. Role for cells in the presupplementary motor area in updating motor plans. *Proceedings of the National Academy of Science USA*, 93:8694–8698, 1996.

[Tanji, 1994] J. Tanji. The supplementary motor area in the cerebral cortex. *Neuroscience Research*, 19:251–268, 1994.

[Tanji and Shima, 1994] J. Tanji and K. Shima. Role for supplementary motor area cells in planning several movements ahead. *Nature*, 371(29):413–416, 1994.

[Wolpert and Kawato, 1998] D. M. Wolpert and M. Kawato. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11:1317–1329, 1998.

Appendix A – Equations

A.1 k^{th} Forward Sequence Model Output

$$\hat{y}_j^k(t+1) = f\left(\sum_{i=0}^{4} w_{ji}^k(t) y_i(t)\right);$$

where, $j = 1,2,3$ (Action Ouputs)
$k = 1,2,3,4$ (Modules / Sequences)
$i = 0,1,2,3,4$ (Inputs)

$$y_i(0) = \begin{cases} 1 & i = 0 \text{ (Bias node)} \\ 0 & i = 1,2,3 \\ 1 & i = 4 \text{ (Ready node)} \end{cases}$$

$$y_i(t) = \begin{cases} 1 & i = 0 \\ \{0,1\} & i = 1,2,3 \text{ (Previous Response)} \\ 0 & i = 4 \end{cases}$$

$$\hat{y}_j^k(t+1) \in (0,1); \text{ (Module Output)}$$

$$f(x) = \frac{1}{1+e^{-4x}}; \text{ (Sigmoid Function)}$$

A.2 Response Selection

$$y_j(t+1) = \begin{cases} 1 & j = m \\ 0 & j \neq m \end{cases}$$

where,

$$m = max_j\left(y_j^\star(t) + \sum_{k=1}^{4} \lambda^k(t)\hat{y}_j^k(t)\right);$$

$$y_j^\star(t) = \begin{cases} \text{Desired Target} & \text{in VT} \\ 0 & \text{in MT} \end{cases}$$

$$j = 1,2,3$$

A.3 Responsibility Estimation

A.3.1 Prediction Error

$$error_j^k(t) = (y_j^{\star\star}(t) - \hat{y}_j^k(t))$$

$$y_j^{\star\star}(t) = \begin{cases} y_j(t) & \text{if Correct Trial} \\ 1 - y_j(t) & \text{if Error Trial} \end{cases}$$

$$E^k(t) = \frac{1}{2}\sum_j \left(error_j^k(t)\right)^2$$

A.3.2 Responsibility

$$p^k(t) = e^{\frac{-E^k(t)}{\sigma(t)^2}}$$

$$\lambda^k(t) = \frac{\prod_{\delta=0}^{3} p^k(t-\delta)}{\sum_{i=1}^{4}\prod_{\delta=0}^{3} p^i(t-\delta)}$$

$$\sigma(t+1) = \alpha\sigma(t) + (1-\alpha)\frac{1}{\Delta}\sum_{\delta=0}^{\Delta} min\left[E^k(\delta)\right]$$

$$\text{where, } p^k(t) = \frac{1}{4}; t \leq 0, k = 1,2,3,4$$

$$\sigma(0) = 3$$

Note 1: The scaling factor keeps a running average of the best prediction error and thus ensures that the responsibility is not locked onto one module prematurely.

Note 2: p and σ values are reset at the beginning of every block of 11 trials (6 visual and 5 memory). Whereas, all other variables are reset at the beginning of every trial.

A.4 Weight Update in the k^{th} Forward Sequence Model

$$w_{ji}^k(t+1) = w_{ji}^k(t) + \eta\lambda^k error_j^k(t)\hat{y}_j^k(t)(1-\hat{y}_j^k(t))y_i(t-1)$$

$$w_{ji}^k(0) \in (0,1)\text{(Uniformly distributed random values)}$$