LNCS 3316, pp.592-598, 2004; Proc. 11th Int. Conf. on Neural Information Processing (ICONIP2004), Calcutta, India.

Generalization in Learning Multiple Temporal Patterns Using RNNPB

Masato Ito[1] and Jun Tani[2]

[1] Sony Corporation, Tokyo, 141-0001, Japan
masato@pdp.crl.sony.co.jp
[2] Brain Science Institute, RIKEN, Saitama, 351-0198, Japan
tani@brain.riken.go.jp

Abstract. This paper examines the generalization capability in learning multiple temporal patterns by the recurrent neural network with parametric bias (RNNPB). Our simulation experiments indicated that the RNNPB can learn multiple patterns as generalized by extracting relational structures shared among the training patterns. It was, however, shown that such generalizations cannot be achieved when the relational structures are complex. Our analysis clarified that the qualitative differences appear in the self-organized internal structures of the network between generalized cases and not-generalized ones.

1 Introduction

Learning temporal patterns from examples are important problems in various domains including robot learning, adaptive process controls, auditory processing and etc. Recurrent neural network (RNN) [1] has been investigated for this purpose and it has been shown that an RNN is good at learning a single pattern by self-organizing a corresponding attractor [1].

Then a question arises that how multiple temporal patterns can be learned using RNNs. There are two distinct approach for the problem by using local representation and distributed representation. In the local representation scheme, each temporal pattern is learned to be stored in a local module network by utilizing winner-take-all dynamics among the modules. Such examples can be seen in [2, 3]. On the other hand in the distributed representation, multiple temporal patterns are learned in a single network by sharing its neural units and synaptic weights. One possible implementation is the scheme of recurrent neural network with parametric biases (RNNPB) [4] in which the parametric bias (PB) plays a role of modulation parameters of RNN dynamical structures. By modulating the PB, different temporal patterns are generated. The values of PB for each of target patterns are self-organized during learning processes.

An interesting characteristics of the RNNPB is that it can generate not only learned patterns but also varieties of unlearned patterns by modulating the PB values [4]. It is, however, not clear yet that how such unlearned patterns are generated and under what sort of constraints they do. One of the most essential characteristics in the distributed representation scheme is that each pattern is learned not independently but as embedded in sorts of relational structures among other learned patterns. Therefore, unlearned patterns can be generated as constrained by such relational structures organized in the network. It can be said that the learning of a set of patterns is generalized when such relational structures among patterns are successfully extracted in the network.

The current study examines this generalization characteristics of the RNNPB learning by conducting a set of simulation experiments. In the first experiment, we investigate how generalization is achieved in learning a set of training patterns those share simple constraints. In the second experiment, we investigate a representational case of not achieving the generalization with using training patterns that are posed with more complex constraints. Through these experiments we attempt to clarify the underlying mechanisms of achieving generalizations as well as not achieving of them in learning multiple patterns in the RNNPB.

2 Model

First, here are the basic ideas behind our model. Figure 1 shows the neural net architecture used in the current study.

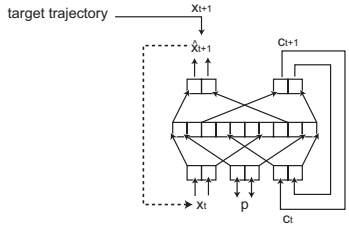

Fig. 1. The RNN associated with the PB inputs.

The architecture employs a Jordan-type [5] recurrent neural network (RNN) associated with the PB nodes. The RNN learns to generate sequence patterns by receiving x_t as inputs and generating predictions of the next step inputs $\hat{x}_{t+1}$ as outputs by using the context state c_t and the PB vector p. Here, x_t, c_t and p are vectors. Note that p is fixed while x_t and c_t change dynamically during generating a temporal pattern. The current context state c_t represented by context nodes in the input layer is mapped to that in the next time step, c_{t+1}, represented by context output nodes. The PB vector plays the role of the pattern modulator which is analogous to bifurcation parameters of nonlinear dynamical systems. Each specific temporal pattern is generated while the PB is clamped to its corresponding value. Specific values of PB for generating each training pattern is self-determined through their prior learning processes (as will be described later). The following subsections will describe the learning processes in detail as well as the pattern generation processes.

2.1 The learning process

The idea of learning in the model is to search simultaneously for optimal synaptic weights that are common for N of training sequence patterns and N of optimal PB vectors each of which is specific to one of the training sequence patterns.

p_n as the PB vector values at learning step n is updated iteratively for each training pattern with using back-propagation through time (BPTT) algorithm [5] while the synaptic weights in the network is updated. The BPTT utilizes a window of a working memory that stores computational results in the current learning step for each training sequence. The forward activation sequence by cascade, the error sequence and the delta error sequence are stored. The step length of the sequences stored in the working memory is denoted as L (that is

equal to step length of the training sequence pattern). For each learning iteration, L steps of the forward dynamics are computed by cascading the network for each sequence. Once the L steps of the output sequence are generated, the errors between the teaching targets x_{t+1} and its prediction outputs $\hat{x}_{t+1}$ are computed. The error at each step is back-propagated through time in the sequence by which the delta error at PB nodes at each step in the cascaded network is obtained. The summation of this delta error over L steps provides the update direction of the PB vector in order to minimize the total error for the training sequence. The update equations for the ith unit of the PB at learning step n are:

$$\delta\rho_n^i = k_{bp} \cdot \sum_{t=0}^{L} \delta_t^{bp^i} \tag{1}$$

$$\Delta\rho_n^i = \epsilon \cdot \delta\rho_n^i + \eta \cdot \Delta\rho_{n-1}^i \tag{2}$$

$$p_n^i = sigmoid(\rho_n^i) \tag{3}$$

In Eq. (1) the δ force for the update of the internal potential values of the PB ρ is obtained from the summation of the delta error at PB node $\delta_t^{bp^i}$ for L steps. Then, the ρ_n is updated by using the delta force by means of the steepest descent method. The current PB p_n are obtained by means of the sigmoidal outputs of the internal potential values ρ_n.

2.2 The pattern generation

Once the synaptic weights in the RNN are determined by the learning process, each of trained sequence pattern can be generated using the so-called closed-loop mode with clamping the PB vector with the values obtained in the learning. In the closed-loop mode, the RNN's forward dynamics proceeds autonomously without receiving the inputs x_t externally. Instead, the prediction outputs $\hat{x}_{t+1}$ is fed-back to the inputs. It is noted that the RNN can generate various dynamic patterns beyond learned ones by arbitrary setting the PB vector, of which characteristics is the main discussion topics in the current paper.

3 Learning Experiments

In the learning experiments, 2 channels sinusoidal patterns are employed for training patterns. In the experiments, two learning sets are used. The set 1 consists of 5 patterns each of which has different amplitude and frequency while other properties, such as phase differences between two channels and offset, are the same. The set 2 consists of 5 patterns each of which has different offset and phase difference between channel 1 and 2 with other properties set as the same. In the first simulations, only training set 1 is learned. In the subsequent simulation, we examine the learning case with these two training sets merged into one training set. The RNNPB used in these experiments has 2 input nodes and 2 prediction output nodes for learning the forward dynamics of patterns. It also had 2 PB nodes, 60 hidden nodes, and 60 context nodes. Parameter settings in Eq. (1), (2) were $k_{bp} = 0.5$, $k_{nb} = 0.4$, $\epsilon = 0.1$, $\eta = 0.9$. In the learning, the RNNPB learns all patterns in the training set simultaneously. The learning is iterated for 100000 steps, starting from randomly set initial synaptic weights. The final root-mean-square error of the output nodes was less than 0.0002 over all learning results.

3.1 Training set 1: amplitude and frequency

In the simulation 1, the RNNPB was trained using the set 1. After it was confirmed that the RNNPB can regenerate all patterns in the set 1, we examined how much the RNNPB can also generate unlearned patterns that are possibly generated by interpolating the training patterns in terms of amplitude and frequency. For this purpose, interpolated patterns are prepared by taking intermediate amplitude and frequency between selected pairs of the training patterns as the targets. Then, the PB values are searched for best mimicking those interpolated patterns.

In Figure 2, it is observed that the RNNPB can generate the interpolated patterns successfully. In Figure 2, the plot (b) shows the outputs generated by the RNNPB, N2-3 that is interpolated between the (a) pattern 2 and (b) 3 in the training set is generated with the PB of $(0.22, 0.20)$. The amplitude of N2-3 is intermediate between the pattern 2 and pattern 3 in the training set. In the same way, other 3 patterns, N2-4, N3-5 and N4-5 have intermediate characteristics among the training patterns. It is noted that those patterns are generated with modulating only for their amplitude and frequency, but not for other properties.

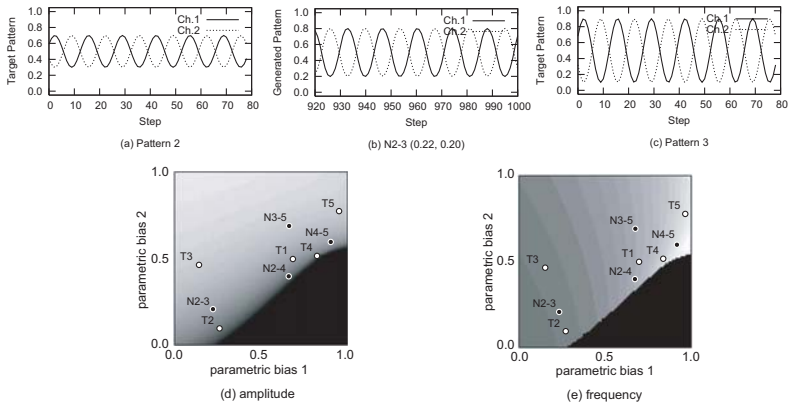

Fig. 2. The generated patterns correspond to intermediate patterns (a) N2-3 between the pattern (a) 2 and (c)3 in the learning sample. The phase plots for (d) the amplitude and (e) the periodicity for the outputs using the values of the parametric biases after learning the training set 1.

Next, in order to clarify the structure of the mapping between the PB and corresponding temporal pattern characteristics, a two-dimensional phase diagram for the PB space was plotted. More specifically, the amplitude and the periodicity of the generated patterns were plotted with two varying values of the PB nodes. The amplitude and the periodicity for generated patterns was computed while the two values of the PB were gradually changed at 0.01 intervals. The sequences of 1000 step lengths were generated by the forward dynamics in the closed-loop mode, and then the periodicity and the amplitude were calculated at each point in the PB space using the sequence from the 600th to the 1000th step in order to exclude the initial transient period.

Figure 2 (d) and (e) show the phase plots generated for (d) amplitude and (e) periodicity by using color grading from black to white. In Figure 2 (d), a white tile denotes that the amplitude is 1.0 while a black one indicates zero amplitude (meaning that the trajectory converges to a fixed point). In Figure 2 (e), a white tile denotes that the periodicity is more than 30 steps, while a black one indicates the convergence to a fixed point. In figure 2 (d) and (e), N labels indicate the PB values that generated the interpolated patterns. And T labels indicate the PB values that regenerated the training patterns.

We observed that the amplitude and periodicity changes smoothly over the PB space except for the region of fixed point dynamics. In the PB space, it is observed that the PB vectors T1, T2, T3, T4 and T5 which can regenerate the training patterns, are widely distributed in the limit cycling dynamics. And it is also observed that each PB vector for the interpolated patterns, N2-3, N2-4 and N3-5, is located in intermediate between T2 and T3, T2 and T4, and T3 and T5, respectively. These observations conclude that the learning of the training set 1 was well generalized by capturing the underlying regularity in the training set 1, that is – in generating sinusoidal patterns their amplitude and periodicity can be changed independently while other profiles are preserved.

Note that we obtained similar results for the set 2 in terms of offset and phase difference.

3.2 Training set 1 and 2

In the simulation 2, the RNNPB was trained using both of the set 1 and 2. We confirmed that the RNNPB can regenerate all patterns in the set 1 and 2.

Figure 3 shows the phase plots generated for (a) amplitude and (b) periodicity using color grading from black to white.

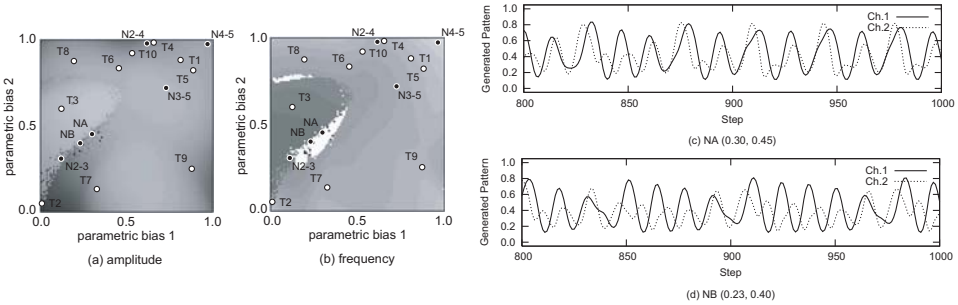

Fig. 3. The phase plots for (a) the amplitude and (b) the periodicity for the outputs using the values of the parametric biases after learning the training set 1 and 2. The diverse patterns (c) and (d) generated with the PB values in the region of complex structures in a relatively small PB space.

It is observed that the PB space is self-organized with much more complex structures compared to those in the simulation 1. The PB points those correspond to the regenerations of the training patterns in the set 2 and their interpolated patterns are placed in the orthogonal region from the top-left to the down-right. On the other hand, those points corresponding to the set 1 are placed as divided into two regions of the down-left and the top-right parts. At

each of these regions, some of the PB points for the interpolations are found to be way out from intermediate between two PB points that correspond to training patterns to be interpolated.

In examining whole possible patterns generated in the all space of the PB, non-periodic patterns were found frequently especially in the boundary between the regions of the set 1 and the set 2. Figure 3 (c) and (d) show such fluctuated patterns generated diversely, which cannot be explained by simple interpolations among the training patterns. (c) NA was generated with the PB vector $(0.30, 0.45)$ and (d) NB with $(0.23, 0.40)$. It is considered that these fluctuated patterns are generated because the PB mapping in the boundary is highly distorted in a nonlinear way where two distinct functional structures meet each other in a conflicting manner. These observations suggest that generalization in learning becomes much harder when embedding of different structures are attempted in a relatively small PB space.

4 Summary

Our experiments showed that the RNNPB can learn multiple temporal patterns by extracting certain common structures among them. In the successful learning case, it was observed that the self-organized mapping between the PB and the characteristics of generated patterns becomes smooth where patterns interpolated among the training patterns can be generated. This explains the generalization capability of the RNNPB when the training patterns share relatively simple constraints. It was, however, observed that the RNNPB learning cannot be generalized well in more complex situations where two distinct relationships exist among training patterns. It seems that the PB mapping is self-organized with substantial distortions by attempting to embed complex structures in a relatively small PB space. In such situations, diverse fluctuated patterns, which cannot be explained by simple interpolations among the trained patterns, tend to be generated.

References

1. Jordan, M.: Attractor dynamics and parallelism in a connectionist sequential machine. In: Proc. of Eighth Annual Conference of Cognitive Science Society, Hillsdale, NJ: Erlbaum (1986) 531–546
2. Wolpert, D., Kawato, M.: Multiple paired forward and inverse models for motor control. Neural Networks **11** (1998) 1317–1329
3. Tani, J., Nolfi, S.: Learning to perceive the world as articulated: an approach for hierarchical learning in sensory-motor systems. In Pfeifer, R., Blumberg, B., Meyer, J., Wilson, S., eds.: From animals to animats 5. Cambridge, MA: MIT Press. (1998) later published in Neural Networks, vol12, pp1131–1141, 1999.
4. Tani, J., Ito, M.: Self-organization of behavioral primitives as multiple attractor dynamics: a robot experiment. IEEE Trans. on Sys. Man and Cybern. Part A **33** (2003) 481–488
5. Rumelhart, D., Hinton, G., Williams, R.: Learning internal representations by error propagation. In Rumelhart, D., Mclelland, J., eds.: Parallel Distributed Processing. Cambridge, MA: MIT Press (1986) 318–362