Discretization of Series of Communication Signals in Noisy Environment by Reinforcement Learning

Katsunari Shibata
Department of Electrical and Electronic Engineering, Oita University, Japan
E-mail: shibata@cc.oita-u.ac.jp

Abstract

Thinking about the "Symbol Grounding Problem" and the brain structure of living things, the author believes that it is the best solution for generating communication in robot-like systems to use a neural network that is trained based on reinforcement learning. As the first step of the research of symbol emergence using neural network, it was examined that parallel analog communication signals are binarized in some degree by noise addition in reinforcement learning-based communication acquisition. In this paper, it is shown that two consecutive analog communication signals are binarized by noise addition using recurrent neural networks. Furthermore, when the noise ratio becomes larger, the degree of the binarization becomes larger.

1 Introduction

We humans can communicate complicated information skillfully using symbols. It has been thought that artificial neural networks(ANNs) are good at continuous nonlinear approximation, but are not good at symbol handling or logical processing. In living things, the functional difference has been pointed out between the left-brain and the right-brain[1]. Based on these, the idea of the specialization that the ANN corresponding to the right-brain is used for pattern processing, and a digital computer corresponding to the left-brain is used for logical processing has been accepted generally. However, there is no general idea about what signals should be transferred between the ANN and the computer, and that causes the "symbol grounding problem". Furthermore, the left-brain and the right-brain looks almost the same in the real brain compared with the difference between the ANN and the digital computer.

The author believes that in order to solve the serious "symbol grounding problem", the pattern processing and the logical processing should not be distinguished. Accordingly it is expected for the ANN to perform the both without any discriminations. For this reason, it is very significant to show that symbols emerge in the ANN only by applying reinforcement learning inspired by the learning of living things. Then, there appears a question "is it true that ANN is not good at symbol processing?"

Here, for simplicity, symbols are considered as discretized signals. Considering from the point of necessity, the reason why the communication signals are discretized can be either "logical thinking" or "elimination of noise effect". Considering from the point of structure, associative memory, in other words, fixed-point dynamics can be a solution to realize the discretization. As the first step of the research of symbol emergence, the necessity of "elimination of noise effect" has been focused on, and it was examined whether parallel analog communication signals are discretized by noise addition in reinforcement learning-based communication acquisition[2][3].

In this paper, like a "word", a series of signals are communicated on behalf of the parallel signals. Same as the previous work, binarization of the signals by reinforcement learning in noisy environment is examined. It seems more difficult than the case of parallel signals because memory is necessary to generate and recognize the communication signals.

2 Learning and Task

A simple communication environment in which only two agents exist is assumed here. Referring to [4], one of them can transmit a communication signal to the other. They are put on an one-dimensional space as shown in Fig. 1. When the both agents touch together, they get a reward. The transmitting agent (transmitter) cannot

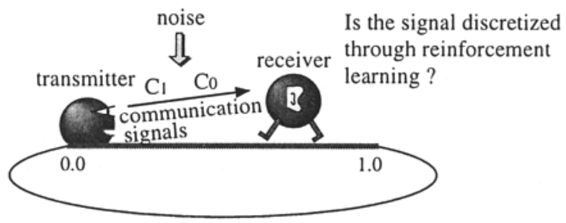

Fig. 1. Communication task employed in this paper.

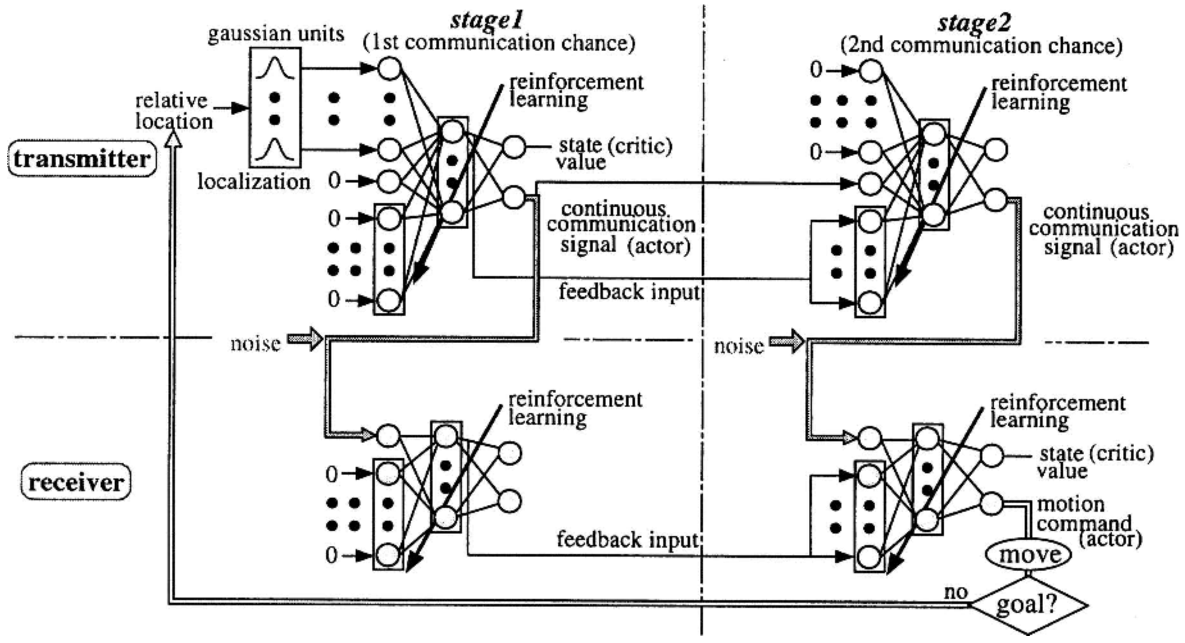

Fig. 2. Architecture of each agent and signal flow in one time step.

move, but can observe the relative location of the opponent, and generates a series of communication signals by its own recurrent neural network (RNN). The receiving agent (receiver) interprets the communication signals and generates its motion command also by its own RNN. It can move according to the motion command. It cannot observe anything except for the communication signal, and cannot transmit anything. Both agents are trained based on reinforcement learning independently.

The transmitter is fixed at the left edge on a one-dimensional space where the left edge is linked to the right edge. The length of the space is 1.0. The receiver is located randomly at every trial. When the motion command is positive, it moves to the right, and when negative, it moves to the left. The absolute value decides the moving distance. When the distance between the transmitter and receiver is less than 0.11, they can touch each other and get a reward. However, if the motion command is too large for the distance, it goes past the transmitter, and they cannot get the reward. Accordingly, the receiver's motion should be in a range, and the range is gradually sifted according to the receiver's location.

Fig. 2 shows the architecture of each agent and the signal flow in one time step. Each agent has an Elman-type RNN that enables memorization of some necessary information. There are two stages in one time step. In the first stage, the transmitter observes the receiver's location, and generates the first communication signal by computing its RNN. The receiver receives the signal, and computes its RNN. The outputs are not used at this stage, but the hidden outputs are used as the feedback inputs at the stage 2. In the stage 2, the transmitter receives the first communication signal as input, and generates the second communication signal. The receiver receives the signal, and generates the motion command.

The information of receiver's location is localized by N Gaussian units. This helps the neural network to learn a strong nonlinear transformation. The center of each Gaussian is arranged between 0.0 and 1.0 with the constant interval of $1.0/(N-1)$, where N is the number of Gaussian units. The size of each Gaussian σ is the same as the interval. The output is described as

$$GS_i(loc) = exp\left(-\frac{1}{2}\left((N-1)\cdot loc - i\right)^2\right), \quad (1)$$

where i is the index of each Gaussian unit ($i = 0, 1, 2, .., N-1$), loc is the receiver's location. Here, $N = 30$.

As a reinforcement learning architecture, actor-critic is employed for each agent. Here, the transmitter deals with the communication signal as its action, while the receiver deals with the communication signal as its state. One of the outputs of each network is used as critic, and the other is used as actor. The hidden neurons are used in common by the both types of outputs. The training signals are generated based on reinforcement learning, and the network is trained by BPTT (Back Propagation Through Time). TD (Temporal Difference) error $\hat{r}$ is calculated as

$$\hat{r}_{t-1} = r_t + \gamma P_t - P_{t-1} \tag{2}$$

where r_t is the reward, P_t is the critic output at t time step, and γ is a discount factor. The critic output is trained by the training signal as

$$P_{s,t-1} = P_{t-1} + \hat{r}_{t-1} = r_t + \gamma P_t. \tag{3}$$

As the critic output P_t, the output at the stage 1 is used for the transmitter, while the output at the stage 2 is used for the receiver. The training signal is also given to the output at the stage 1 for the transmitter, while to the output at the stage 2 for the receiver. The actual motion M_t is calculated as

$$M_t = \alpha(2.5A_t + rnd_t + n_t) \tag{4}$$

where A_t is the actor output, rnd is the random number for trial and error, and n is the noise factor that is not added in the case of the receiver's motion, but is added in the case of the communication signal. α is a constant. The actor output is trained by the training signal as

$$A_{s,t-1} = A_{t-1} + \beta\hat{r}_{t-1}rnd_{t-1} \tag{5}$$

where β is a constant, and it is 0.5 here. The training signal is given to the output at each stage for the transmitter, while to the output at the stage 2 for the receiver. When the training signal is given to the output at the stage 2, the learning traces back also to the stage 1 based on BPTT.

The output function of each hidden or output neuron is a sigmoid function that ranges from -0.5 to 0.5. All the training signals are limited from -0.4 to 0.4 to avoid the saturation area of the sigmoid function. In Eq. (4), by multiplying 2.5 to each actor output, the range becomes from -1.0 to 1.0, and after that, the trial and error factor and noise are added. When the value becomes larger than 1.0 or less than -1.0, it is returned to 1.0 or -1.0 respectively. Here, the trial and error factor is cubed uniform random number whose level, in other words, whose amplitude is ± 0.4. The noise factor is a uniform random number whose level is varied from ± 0.0 to ± 1.6 with the interval of 0.2 in the following simulations. Even in the case that the noise factor is always zero, the random number for the trial and error factor of the transmitter is received as a noise for the receiver.

For the critic computation based on TD learning in Eq. (2) and (3), 0.5 is added to the critic output actually. The reward that is given to each agent is 0.9. To generate the communication signal, α in Eq. 4 is 1.0 in the transmitter. For the motion command, α is 0.4 in the receiver so as that the receiver can touch the transmitter in one step from any locations by an appropriate motion. If the receiver's motion is discretized completely, no less than 4 levels of output is required. The number of layers is 3, and the number of neurons in the hidden layer is 10 for both agents. All the initial connection weights from the hidden layer to the output layer are 0.0, and those from the input layer including feedback inputs to the hidden layer are decided randomly in the range from -1.0 to 1.0.

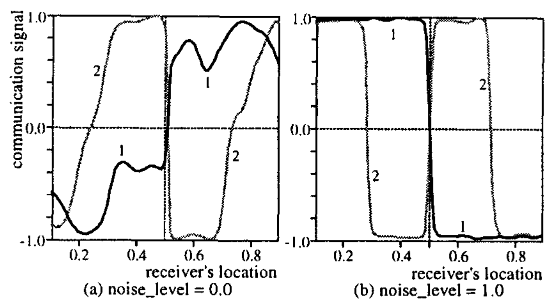

Fig. 3. The communication signals as a function of the receiver's location. The random number level is 0.4.

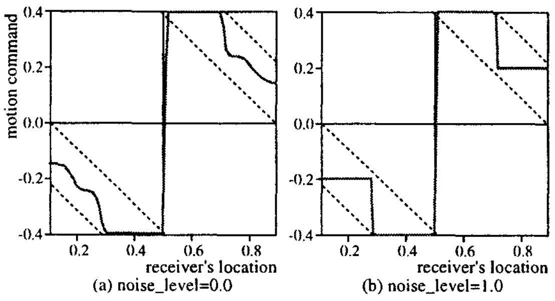

Fig. 4. The receiver's motion command as a function of the receiver's location. The random number level is 0.4 and the noise level is 1.0.

3 Result

It was observed whether the transmitted signals became discrete when the noises were added to the communication signals during the learning. The consecutive two communication signals and the motion command after learning with no noise are shown in Fig. 3(a) and Fig. 4(a), and those with some noise (level=1.0) are shown in Fig. 3(b) and Fig. 4(b). The initial connection weights are the same between the two cases. The sloping lines in Fig. 4 indicate the maximum and minimum limit values of the motion for the receiver to touch the transmitter in one time step by the motion as a function of the receiver's location.

Roughly, the results are similar to the case of the parallel communication signals[2][3]. In the both cases, the transmitter generated the first communication signal from the receiver's location and generated the second one from the first communication signal and the feedback inputs. Then the receiver received the first communication signal, kept the information through the feedback inputs, and could acquire appropriate motions from the stored information and the second signal at the stage 2. After learning with noise, each communication signal was almost binarized, and only around the boundary of the binary values, the signal took a medium value. However, it is clear that the degree of binarization is larger than in the case of no noise. The receiver's motion is discretized into four levels in the optimal range by the combination of the two consecutive binary communication signals. The motion is more clearly discretized than the communication signals. The reason might be that the receiver also learned to binarize the received signal utilizing non-linear transformation of the neural network. When the number of communication chances was increased to three times, the assignment of information to the signal of each chance could not be done well, and the optimal motion command could not be acquired. That is different from the case of parallel signals.

The degree of the binarization according to the noise level was also observed. The degree of binarization that means how the signal is close to the maximum value 1.0 or the minimum value -1.0 is defined as

$$bin = \sum_{i}^{Nc}\sum_{j}^{Nd}|com_{i,j}|/(Nc \cdot Nd) \qquad (6)$$

where Nc is the number of communication chances, Nd is the number of sampled receiver's locations, and com is the communication signal without the random number and noise. If the communication signal is always -1.0 or 1.0, the degree becomes the maximum of 1.0, while if the signal is always 0.0, it becomes the minimum of 0.0. The degree of binarization according to the noise level is shown in Fig. 5(a). Each small circle shows the average degree over 50 simulations, and the vertical line shows the standard deviation. It can be seen that when the noise level becomes larger, the degree becomes larger and the deviation becomes smaller. However, when the noise level becomes larger than 1.0, the degree decreases slightly according to the noise level. When the noise level is 1.6, the both signals encode the same information as the signal 1 in Fig. 3(b) in most cases. That results in the high degree of binarization.

The noise tolerance was also examined. Fig. 5(b) shows the average time steps to the goal as a function of the noise level in the learning phase for each noise level in the test phase after learning. It can be seen that when some noise is added in the test phase, the performance is the best when the noise level in the learning phase is 1.0. It is interesting that under the same condition, the degree of binarization is the maximum. However, when

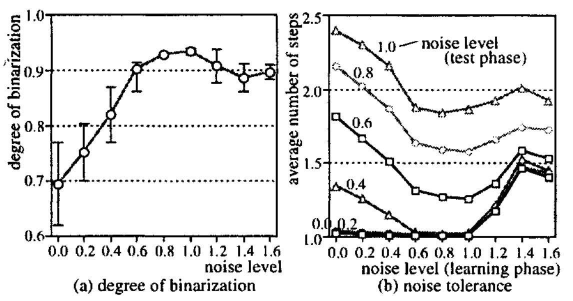

Fig. 5. Noise Effect. (a)Degree of binarization according to the noise level in the learning phase. (b)Noise tolerance according to the noise level in the learning phase.

the noise level in the learning phase is larger than 1.0, the performance becomes worse even if the noise level in the test phase is 0.0. This means that the large noise in the learning phase disturbed the proper learning even for the case of no noise.

4 Conclusion

It was shown that as well as the case of the parallel communication signals, noise addition has the effect to promote the binarization of the signal only by reinforcement learning in the case of a series of signals. The reason why appropriate signals cannot be obtained in the case of more than two signals should be examined.

Acknowledgment

This research was supported by the Japan Society for the Promotion of Science, Grants-in-Aid for Scientific Research, #14350227 and #15300064. The author also would like to say thanks to Dr. J. Yoshimoto at Nara Inst. of Science and Technology, Japan who advised the interest of series of communication signals.

References

[1] Sperry, R.W. (1968) Hemisphere Deconnection and Unity in Conscious Awareness, *American Psychologist*, **23**, pp. 723–733

[2] Shibata, K. and Sugisaka, M. (2004) Discretization of Analog Communication Signals by Noise Addition in Communication Learning, *Proc. of AROB 9th*, **2**, pp. 351-354

[3] Shibata, K (2004) Discretization of Analog Communication Signals by Noise Addition in Reinforcement Learning of Communication, *Technical Report of IEICE*, **103** (734), NC2003-203, pp. 55-60 (in Japanese)

[4] Ono, N, Ohira, T. and Rahmani, A.T. (1995) Emergent Organization of Interspecies Communication in Q-Learning Artificial Organs, *Advances in Artificial Life*, pp.396–405