The Twenty-Fifth International Symposium on Artificial Life and Robotics 2020 (AROB 25th 2020),
The Fifth International Symposium on BioComplexity 2020 (ISBC 5th 2020),
B-Con Plaza, Beppu, Japan, January 22-24, 2020

Evolution of metamemory based on self-reference in artificial neural network with neuromodulation

Yusuke Yamato†[1], Reiji Suzuki[1] and Takaya Arita[1]

[1]Graduate School of Informatics, Nagoya University
†yamato@alife.cs.is.nagoya-u.ac.jp

Abstract: The ability that is people's self-monitoring and controlling of their memory processes is called metamemory. It has been regarded as truly unique characteristics of human memory, and has been studied widely as a component of metacognition in cognitive psychology. We aim to evolve artificial neural networks with neuromodulation, that has a metamemory function. Our constructive approach is based on the repetition of evolutionary experiments, analysis of the evolved networks and refinement of the measure, to reduce the gap between the functional properties of behavior and subjective reports of phenomenal experience. In this paper, we show the evolution of a neural network that has metamemory function based on the self-reference of memory, and analysis of the mechanism of metamemory.

Keywords: Metamemory, Cognitive science, Neural networks, Evolutionary computations

1 INTRODUCTION

Metamemory refers to people's self-monitoring and self-control of their own memory processes. It has been regarded as truly unique characteristics of human memory [1], and has been studied widely as a component of metacognition in cognitive psychology. Testing metamemory in animals has been a challenging topic as comparative studies. Memory awareness in humans is inferred primarily based on verbal reports from human subjects, which is difficult in animals. Hampton devised a delayed matching-to-sample (DMTS) paradigm [2]: Animals able to distinguish between the presence and absence of their own memory should improve accuracy if allowed to decline memory tests when they have forgotten. They also should decline tests by selecting the escape option more frequently if memory is attenuated experimentally. Hampton showed that one of two macaque monkeys examined met these criteria. Since then, metamemory of other animals has been investigated based occasionally on other methodologies (information-seeking and postdecision wagering paradigms). Macaque monkeys, apes, and dolphins are the only animals besides humans that were demonstrated to exhibit metamemory skills [3].

Using a constructive computational method as an alternative approach, we have been investigating whether agents controlled by artificial neural networks could have metamemory capacity. We evolved neural networks using the neuromodulation technique by which modulatory neurons [4] can dynamically alter the plasticity of the connections of the neurons they project to. We successfully found some of the evolved agents clearly showed the metamemory capability from the perspective of the DMTS paradigm [5].

Using computational methods allows us to investigate the mechanisms that drive the behaviors, while we have to rely heavily on behavioristic paradigms (e.g., DMTS paradigm) when dealing with living animals. Therefore, we then focused on the mechanisms of the evolved neural networks that meets the Hampton's criteria [6, 7]. We found that the evolved neural networks just associate particular input configurations that are more difficult than the rest, with the escape option in order to obtain greater rewards. In the discussion on interpretation of animal experiments, this type of solution was indeed predicted, and for this reason, the paradigm was criticized [8]. This finding lead us to define the following two criteria to exclude such solutions.

- Criterion 1 (C1)

  The agent satisfies one of the behavioristic paradigms for metamemory (e.g., DMTS paradigm), but not by changing the behavior according mainly to particular stimuli configurations.

- Criterion 2 (C2)

  C1 is met, but is based on the self-reference on some part of the stored information regarding the stimuli input.

We performed evolutionary experiments and analyzed the evolved neural networks to find that detection of forgetting is achieved not by self-reference on the stored information but by some kind of spurious relationship between stored information and escape selection both controlled by modulatory neurons. In other words, the evolved neural networks satisfies C1 but not C2 [6, 7].

This paper explores neural networks with neuromodulation that satisfy the criterion C2 by performing computational evolutionary experiments. For this purpose, we modify the previous experimental settings, including the addition of Gaussian noise to all of the neurons for obstructing the evolution of neural circuits that are not robust.

2 TASK

Fig. 1 shows an overview of the task based on DMTS paradigm [5]. Firstly, an agent receives a target pattern composed of 5 binary digits, which is randomly selected from 00001, 00010, 00100, 01000 and 10000 in the study phase. The delay phase follows, in which the agent receives 00000

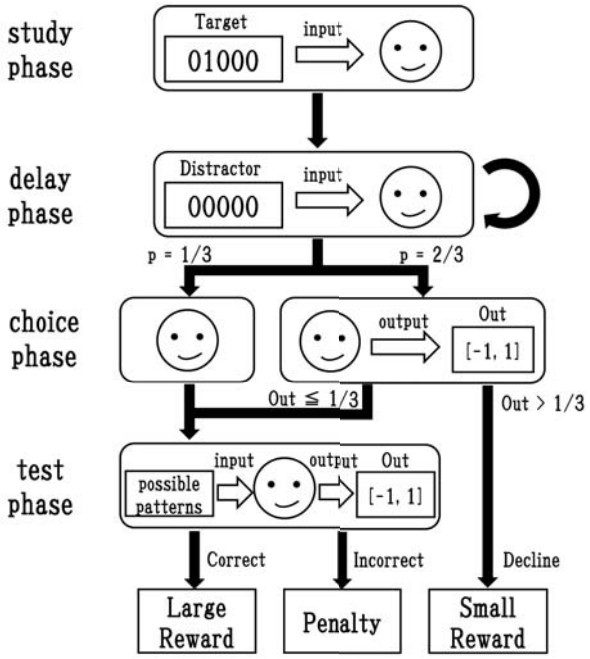


Fig. 1: The delayed match-to-sample task which is introduced the escape option.

as a distractor pattern a predefined number of times defined by Eq. 1, which will affect the degree of uncertainty in its memory.

$$N_{delay} = \lfloor \frac{-1}{\lambda \times \ln(R)} \rfloor + 1, \quad (1)$$

where λ is a parameter related to the shape of the distribution, and R is a uniform random number from [0, 1].

Then, with a probability of 2/3, the choice phase starts. During the phase, the agent receives a signal meaning that it is in that phase and a constant value (CV) which is given to each input neuron as an input pattern. The constant value is expected to be used for self-reference of a memory. One output from the agent in the range [-1, 1] will be interpreted as an intention to decline the trial when it is more than 1/3. In this case, the agent receives a small reward (0.3), Otherwise if the value equals to or is more than 1/3, it will be interpreted as an intention to take the test. On the other hand, with a probability of 1/3, the choice phase is skipped as a compulsory condition.

In the test phase, the agent receives all patterns one by one in random order, and an output ranged in [-1, 1] is interpreted as a response for each pattern. Specifically, when the response goes over 1/3 for the first time, the corresponding input pattern will be interpreted as the selection of the agent and the task ends. If it matches the target pattern presented in the study phase, the agent is rewarded with a large reward (1.0). Otherwise, if it does not match the target or all responses do not go over the value of 1/3, it is rewarded with nothing.

In addition, we use an unsolvable condition, in which agents do not receive one of the 5 patterns but receive a distractor pattern 00000 in the study phase. In this condition, they receive the large reward (1.0) only if it selects the decline option. We expect this addition accelerates the evolution of the ability to select the option.

3 MODEL

3.1 Neural network

The neural network of each agent is composed of several standard neurons including 7 input and 2 output neurons, and modulatory neurons (described in the next subsection) as shown in Fig. 2. Among 7 inputs, one input neuron gets a signal indicating whether it is in the choice phase (1) or not (0). Another input neuron gets a bias that is always 1. Each input of the other 5 neurons gets one of 5 digits of an input pattern, respectively.

The topology of the network evolves while keeping the number of neurons not more than 16, including standard and modulatory neurons, but excluding input neurons. Each value of all neurons in all phases is slightly modified by adding a suitable amount of Gaussian noise (input neurons: $\mu = 0.0$ and $\sigma = 0.1$, other neurons: $\mu = 0.0$ and $\sigma = 0.0001$). The reason for adding the noise is to prevent evolution of agents that are not robust. We regard the lack of the robustness as the cause of the failure of the previous study in which the evolved agents met C1 but not C2 [7]. The network is inputted the same patterns sequentially for 4 times in the choice phase and 3 times in the other phases. Repetition of input is essential when the evolved neural networks have recurrent connections.

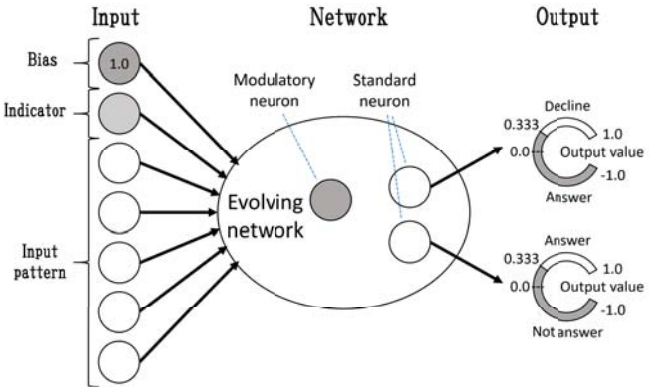


Fig. 2: Inputs and outputs of a neural network.

3.2 Neuromodulation

The neural network contains modulatory neurons in addition to standard neurons. Modulatory neurons affect the learning rate of the connection weights of target neurons and dynamically change it as shown in Fig. 3. In particular, the output of the modulatory neuron m_i modulates the learning rate of the update rule of the connection weight by using it as a modulatory signal instead of directly affecting an activation signal a_i. They are computed by Eqs. 2 and 3.

$$a_i = \sum_{j \in Std} w_{ji} \cdot o_j, \tag{2}$$

$$m_i = \sum_{j \in Mod} w_{ji} \cdot o_j, \tag{3}$$

where w_{ji} is a connection weight from a presynaptic neuron j to a postsynaptic neuron i. Std and Mod are the sets of standard and modulatory neurons connected to the neuron i, respectively. o_j is an output of the neuron j and is computed as $o_j = \tanh(a_j)$. The connection weight from a neuron j to a neuron i is updated by Eq. 4, which is based on an extension of the Hebb's rule called Extended Hebbian rule [9].

$$\Delta w_{ji} = \tanh(m_i) \cdot \eta \cdot (Ao_j o_i + Bo_j + Co_i + D), \tag{4}$$

where o_j and o_i are the outputs of a presynaptic neuron j and a postsynaptic neuron i, respectively. η, A, B, C and D are also genetic parameters. Thus, the update rule can represent various types of synaptic updating via the evolution of these parameters.

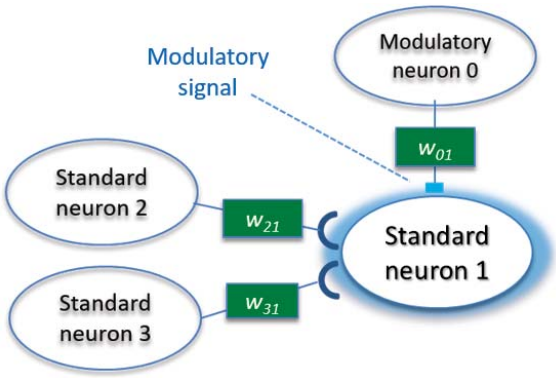


Fig. 3: The overview of the neuromodulation.

3.3 Evolutionary algorithm

We use a variant of a genetic algorithm. The setting of the algorithm is basically the same as the one proposed in [5]. Each agent has a matrix of real-valued connection weights (one axis corresponds to presynaptic neurons while the other to postsynaptic neurons) and the type of each neuron (standard or modulatory) for the decision of the structure, and five parameters (η, A, B, C, D) for the update rule Eq. 4, as a genome. Each connection weight w_{ij} and η are ranged in [-1, 1] while $A, \ldots, B$ in [-100, 100]. These parameters in genotype are converted to each weight w_{ji} in the phenotype by Eq. 5 and $A, \ldots, B$ in the phenotype by Eq. 6 except for η.

$$w_{ji} = \begin{cases} 0 & (|w_{ji}^3| < 0.1) \\ 10 \cdot w_{ji}^3 & (otherwise), \end{cases} \tag{5}$$

$$p = \begin{cases} 0 & (|p^3| < 0.1) \\ p^3 & (otherwise), \end{cases} \tag{6}$$

where $p \in \{A, B, C, D\}$. The total score of each agent obtained by performing the task of each is defined as its fitness. The genetic operators are conducted as follows. Individuals are stored in an array, and are divided into consecutive segments of size 5 (with random segmentation offset at each generation). The best individual of each segment becomes a parent, and generates 5 children for its segment by repeating crossover with a probability of 0.1 or copying itself otherwise. When crossover happens, a partner individual is randomly selected from the all individuals in the population. Random integers r and c are selected from [1, N], and two matrices are generated by exchanging the sub matrices of the parents that are composed by i, j elements with i and j less than or equal to r and c, respectively. A uniform crossover is performed on the parameters for the plasticity rule. Then, as a mutation operator, with a probability of 0.1, Gaussian noise (μ = 0.0, σ = 0.3) is added to each of connection weights and the parameters for the plasticity rule except η, while Gaussian noise (μ = 0.0, σ = 3.0) is added to η. Finally, insertion, deletion and duplication of each neuron are independently performed with probabilities of 0.04, 0.06 and 0.02, respectively. When insertion happens, the weight of the added neuron is randomly set in the range [-1, 1] and the type of the neuron (standard or regulatory) is randomly set. These processes constitute a generation and are repeated G times.

4 EXPERIMENTAL AND ANALYSES

We performed 10 evolutionary trials using the parameters: N = 300, G = 1000, λ = 0.7 and CV = 0.25. During the first 100 generations, the agents have to jump to the test phase by skipping the choice phase every time. This setting is meant to guide the evolution in the initial stage. In subsequent generations, the tasks with the unsolvable condition were performed U = 50 times randomly among T = 300 times of task execution. The neural network of each agent was initialized by setting random values within corresponding possible ranges every when a task started.

We successfully found neural networks by analysis that satisfy C2 in 1 trial among 10 trials. Fig. 4 shows the behavior of the evolved agent at each delay time in the delay phase. This individual has the accuracy of the selection condition which is higher than that in forcing condition. We can also observe the increase in the avoidance rate with the increase in the delay. This result was reasonable if the agent has a metamemory function, and was also observed in the behavior of the monkey in the Hampton's experiment.

Table 1 shows the average values of accuracy with a time delay of 40 in forced tests, chosen tests, and declined tests (assuming that they had taken the test despite declining). The latter two are shown with the probabilities of taking and declining in parentheses, respectively. There is a certain difference in accuracy between forced and chosen tests, and between chosen and declined tests, independent of input patterns. This reveals that the agent tended to avoid the trials

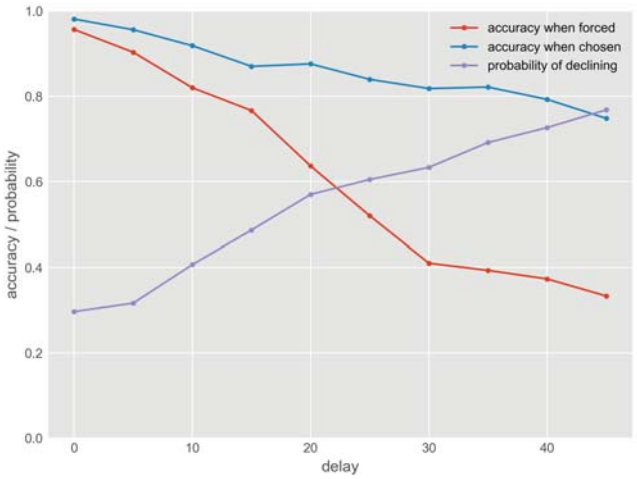


Fig. 4: The behavior of the evolved agent for each delay time.

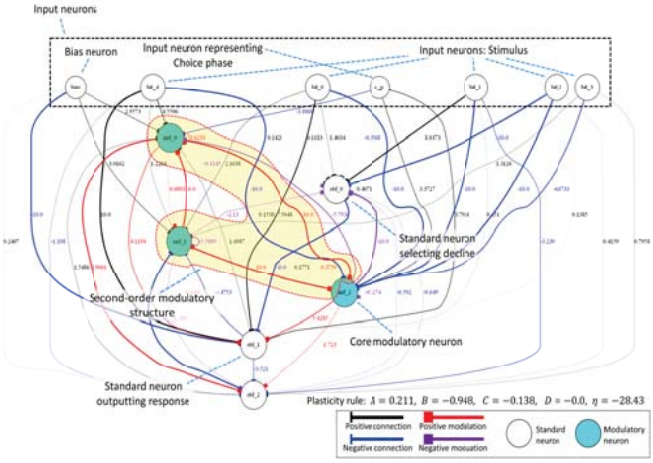


Fig. 5: The structure of the evolved neural network. The width of each line represents the weight of a connection.

which it could not answer correctly. The behavior of this agent satisfies C1 because the behavior was not simply based on the response for particular input stimuli configurations.

Table 1: Accuracy for each input pattern in each case ($delay = 40$).

Input patterns	Accuracy in forced tests	Accuracy in chosen tests (probability of taking)	Accuracy in declined tests (probability of declining)
00001	0.361	0.835 (0.270)	0.175 (0.730)
00010	0.354	0.803 (0.266)	0.216 (0.734)
00100	0.340	0.764 (0.240)	0.205 (0.760)
01000	0.361	0.819 (0.258)	0.183 (0.742)
10000	0.347	0.816 (0.272)	0.206 (0.728)

We analyzed how the network had solved the metamemory task by observing temporal changes in neuronal activity and connection weights in the network. Fig. 5 shows the network which was analyzed above. As shown in the yellow area of Fig. 5, we found that the networks had a second-order modulator structure in which two modulatory neurons modulated some connections from the standard neurons to another modulatory neuron. This structure played an important role in memorizing a received pattern, declining the test and answering the test.

Fig. 6a shows the mechanism of memorizing a received pattern. The network reflects and keeps a structure of a target pattern which it received in the study phase. This ability is made by two modulatory neurons maximally strengthening/weakening each connection weight between the corresponding input neuron and a modulatory neuron (hereinafter referred to as *core modulatory neuron)* by outputting modulatory signals. During the delay phase, the structure of the memorized pattern is probably protected because the learning of the connection weights is impeded by the absolute value of the sum of the modulatory signals becomes a small value. However, the absolute value becomes larger than the small value if the other two modulatory neurons are disturbed by much noise. In this case, the learning is promoted, which can make the agent forget the memorized pattern.

Fig. 6b shows the mechanism of declining the test. The core modulatory neuron can manage the behaviors of declining and answering by its activity state. It judges the presence of the memory of the presented pattern by the sign of the sum of the inputs coming through each connection (each weight of which should reflect the presented pattern) between the input neurons and the core modulatory neuron. If and only if the agent forgets the memorized pattern, the core modulatory neuron receives the negative sum of the inputs and activates negatively in the choice phase. As a result, the core modulatory neuron changes some connection weights and the declining neuron activates positively. Thus, the network successfully declines the test. We also found that the mechanism of pattern selection in the test phase was also based on the monitoring of memory state by the core modulatory neuron.

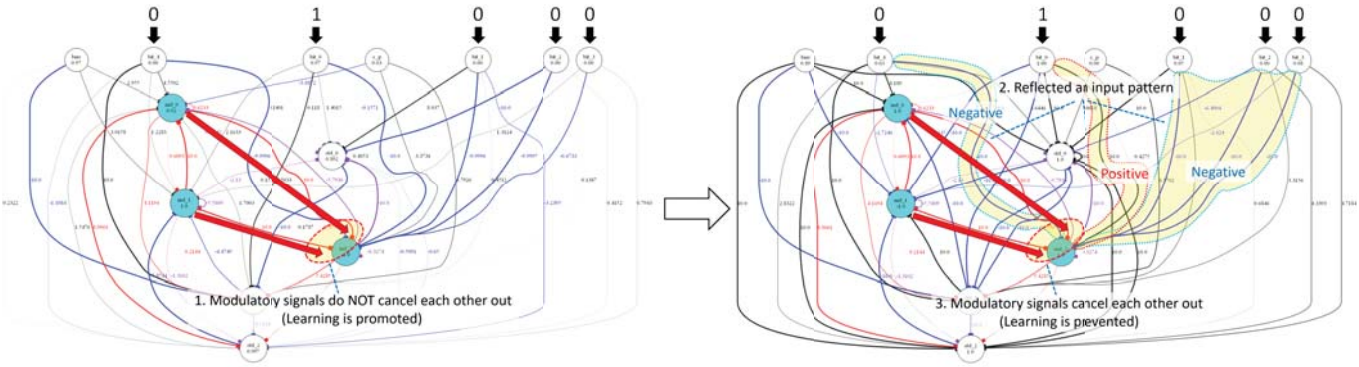
Thus, our experiments indicate that, in conclusion, that these networks can work correctly based on self-reference, in other words, they satisfy C2.

5 CONCLUSION

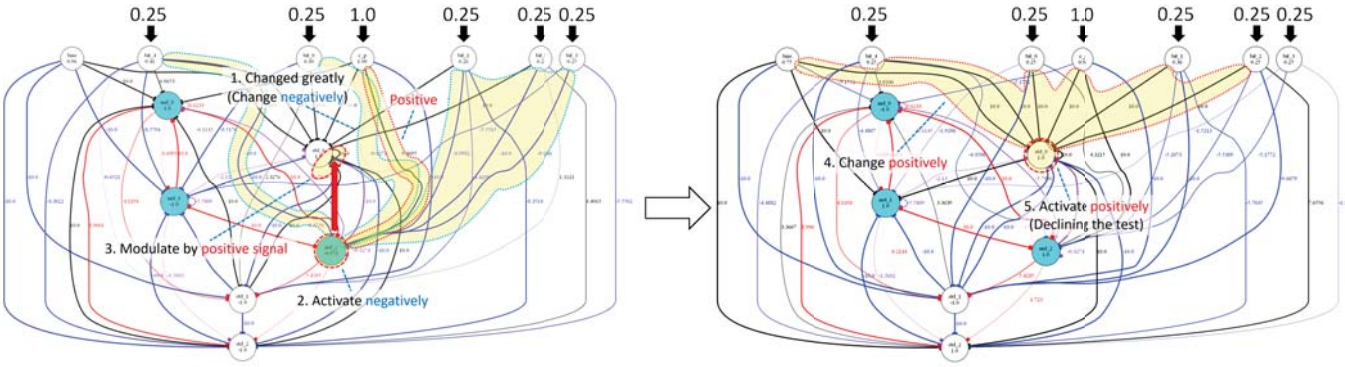
This paper reported on the results of the evolution of metamemory based on the self-reference, and the analysis of its mechanism. We used the criteria (C1 and C2) which were previously defined for clarifying how to exclude the solutions that had been criticized in the discussion on the experiments based on the escape response paradigm dealing with metacognition in non-human animals.

We performed evolutionary experiments, and found an agent which met C2. We investigated the evolved neural network of the agent by analyzing its behavior, structure and neural dynamics.

We found that the agent showed the behavior like a monkey which was claimed to have the ability of metamemory in Hampton's experiment, and that the behavior is not a reflex



(a) The mechanism of memorizing and keeping a target pattern in the study phase. 1) Learning of connection weights is promoted because the modulatory signal from mod_0 to mod_2 and that from mod_1 to mod_2 do not cancel each other out. 2) As a result, some connections reflect a structure of an input pattern. 3) Thereafter, the modulatory signals cancel each other out and learning is prevented. In other words, the reflected information is protected.



(b) The mechanism of declining the test in the choice phase. 1) If the connections in which an input pattern is memorized change greatly, mod_2 (core modulatory neuron) receives the inputs in which the sum is negative from bit_0, ..., bit_4 and c_p (where c_p acts as a bias in order to judge whether the agent forgets or not). 2) As a result, mod_2 activates negatively, and 3) the std_0 (the declining neuron) is modulated by the positive signal from mod_2. 4) Therefore, the connections which connect to std_0 change positively, and 5) std_0 activates positively and the network declines the test.

Fig. 6: The mechanism by which the network makes memorizing and keeping a target pattern (00001), and declining the test. A red arrow represents a modulation. Each the event occurs in numerical order.

behavior for particular input stimuli configurations.

Furthermore, we observed temporal changes in the state of the network. We found that the network had a structure in which two modulatory neurons modulated some connections from the standard neurons to another modulatory neuron. This structure played an important role in memory and metamemory ability, which is as follows: In the study phase, the network reflected and kept the structure of the received pattern in each connection weight between the corresponding input neuron and the modulatory neuron by the effect of a pair of modulatory neurons. Moreover, the modulatory neuron modulated the network circuit based on a monitoring result of memory state in the choice phase. The modulation of the network allowed the network to appropriately decline or answer the test. In other words, the mechanism was based on self-reference.

We believe that our study contributes to the understanding of human metamemory and realization of artificial consciousness.

ACKNOWLEDGEMENTS

This work was supported in part by JSPS/MEXT KAKENHI: JP17H06383 in #4903 (Evolinguistics).

REFERENCES

[1] Endel Tulving and Stephen A Madigan. Memory and verbal learning. *Annual review of psychology*, 21(1): 437–484, 1970.

[2] Robert R Hampton. Rhesus monkeys know when they remember. *Proc. of the National Academy of Sciences*, 98(9): 5359–5362, 2001.

[3] Kentaro Miyamoto, Takahiro Osada, Rieko Setsuie, Masaki Takeda, Keita Tamura, Yusuke Adachi, and Yasushi Miyashita. Causal neural network of metamemory for retrospection in primates. *Science*, 355(6321): 188–193, 2017.

[4] Andrea Soltoggio, John A Bullinaria, Claudio Mattiussi, Peter Dürr, and Dario Floreano. Evolutionary advantages

of neuromodulated plasticity in dynamic, reward-based scenarios. *Proc. of the 11th international conference on artificial life (Alife XI)*, pages 569–576, 2008.

[5] Masaru Sudo, Reiji Suzuki, and Takaya Arita. Can agents with neuromodulation know when they remember? *Proc. of the 19th Int'l Symp. on Artificial Life and Robotics*, pages 330–334, 2014.

[6] Yusuke Yamato, Reiji Suzuki, and Takaya Arita. Creating metamemory by evolving neural network with neuromodulation. *Proc. of the 24th Int'l Symp. on Artificial Life and Robotics*, pages 85–90, 2019.

[7] Yusuke Yamato, Reiji Suzuki, and Takaya Arita. Evolution of metamemory ability by artificial neural networks with neuromodulation. *Proc. of the 2019 Conference on Artificial Life*, pages 461–462, 2019.

[8] Josep Call. Do apes know that they could be wrong? *Animal cognition*, 13(5): 689–700, 2010.

[9] Jean-Baptiste Mouret and Paul Tonelli. Artificial evolution of plastic neural networks: a few key concepts. In *Growing adaptive machines*, pages 251–261. Springer, 2014.