Evolution of Recollection and Prediction in Neural Networks

Ji Ryang Chung, Jaerock Kwon, and Yoonsuck Choe

Abstract— **A large number of neural network models are based on a feedforward topology (perceptrons, backpropagation networks, radial basis functions, support vector machines, etc.), thus lacking dynamics. In such networks, the order of input presentation is meaningless (i.e., it does not affect the behavior) since the behavior is largely reactive. That is, such neural networks can only operate in the present, having no access to the past or the future. However, biological neural networks are mostly constructed with a recurrent topology, and recurrent (artificial) neural network models are able to exhibit rich temporal dynamics, thus time becomes an essential factor in their operation. In this paper, we will investigate the emergence of recollection and prediction in evolving neural networks. First, we will show how reactive, feedforward networks can evolve a memory-like function (recollection) through utilizing external markers dropped and detected in the environment. Second, we will investigate how recurrent networks with more predictable internal state trajectory can emerge as an eventual winner in evolutionary struggle when competing networks with less predictable trajectory show the same level of behavioral performance. We expect our results to help us better understand the evolutionary origin of recollection and prediction in neuronal networks, and better appreciate the role of time in brain function.**

I. Introduction

Many neural network models are based on a feedforward topology (perceptrons, backpropagation networks, radial basis functions, support vector machines, etc.), thus lacking dynamics (see [1], and selective chapters in [2]). In such networks, the order of input presentation is meaningless (i.e., it does not affect the behavior) since the behavior is largely reactive. That is, such neural networks can only operate in the present, having no access to the past or the future. However, biological neural networks are mostly constructed with a recurrent topology (e.g., the visual areas in the brain are not strictly hierarchical [3]). Furthermore, recurrent (artificial) neural network models are able to exhibit rich temporal dynamics [4], [5], [6]. Thus, time becomes an essential factor in neural network operation, whether it is natural or artificial (also see [7], [8], [9], [10]).

Our main approach in here is to investigate the emergence of recollection and prediction in evolving neural networks. Recollection allows organism to connect with its past, and prediction with its future. If time was not relevant to the organism, it would always live in the eternal present.

First, we will investigate the evolution of recollection. We will see how reactive, feedforward networks can evolve a memory-like function (recollection), through utilizing external markers dropped and detected in the environment. In this part, we trained a feedforward network using neuroevolution, where the network is allowed to drop and detect markers in the external environment. Our hypothesis is that this kind of agents could have been an evolutionary bridge between purely reactive agents and fully memory-capable agents. The network is tested in a falling-ball catching task inspired by [6], [11], where an agent with a set of range sensors is supposed to catch multiple falling balls. The trick is that while trying to catch one ball, the other ball can go out of view of the range sensors, thus requiring some sort of memory to be successful. Our results show that even feedforward networks can exhibit memory-like behavior if they are allowed to conduct some form of material interaction, thus closing the loop through the environment (cf. [12]). This experiment will allow us to understand how recollection (memory) could have evolved.

Second, we will examine the evolution of prediction. Once the recurrent topology is established, how can predictive function evolve, based on the recurrent network's recollective (memory-like) property? For this, we trained a recurrent neural network in a 2D pole-balancing task [13], again using neuroevolution (cf. [14], [15], [16]). Here, the agent is supposed to balance an upright pole while moving in an enclosed arena. This task, due to its more dynamic nature, requires more predictive power to be successful than the simple ball-catching task. Our main question here was whether individuals with a more predictable internal state trajectory have a competitive edge over those with less predictable trajectory. We partitioned high-performing individuals into two groups (i.e., they have the same behavioral performance), those with high internal state predictability and those with low internal state predictability. It turns out that individuals with highly predictable internal state have a competitive edge over their counterpart when the environment poses a tougher problem [17].

In sum, our results suggest how recollection and prediction may have evolved. We expect our results to help us better understand the evolutionary origin of recollection and prediction in neuronal networks, and better appreciate the role of time in neural network models. The rest of the paper is organized as follows. Sec. II presents the method and results from the recollection experiment, and Sec. III, those from the prediction experiment. We will discuss interesting points arising from this research (Sec. IV), and conclude our paper in Sec. V.

II. Part I: Evolution of Recollection

In this section, we will investigate how memory-like behavior can evolve in a reactive, feedforward network. Below, we will describe the ball catching task, and the explain in

Ji Ryang Chung, Jaerock Kwon, and Yoonsuck Choe are with the Department of Computer Science and Engineering, Texas A&M University, 3112 TAMU College Station, TX 77843-3112, USA email: jrkwon@tamu.edu,jchung@cs.tamu.edu,choe@tamu.edu

detail our neuroevolution methods for the learning component. Next, we will present the details of our experiments and the outcomes.

A. Task: Catching Falling Balls

The main task for this part was the falling ball catching task, inspired by [6], [11]. The task is illustrated in Fig. 1. See the figure caption for details. The task is simple enough, yet includes interesting dynamic components and temporal dependency. The horizontal locations of the balls are on the two different sides (left or right) of the agent's initial position. Between the left and right balls, one is randomly chosen to have faster falling speed (2 times faster than the other). The exact locations are randomly set with the constraint that they must be separated far enough to guarantee that the slower one must go out of the sensor range as the agent moves to catch the faster one. For example, as shown in Fig. 1C, when there are multiple balls to catch and when the balls are falling at different speeds, catching one ball (usually the faster one) results in the other ball (the slower one) going out of view of the range sensors. Note that both the left-left or right-right ball settings cannot preserve the memory requirement of the task. The vertical location, ball speed, and agent speed are experimentally chosen to guarantee that the trained agent can successfully catch both balls. In order to tackle this kind of situation, the controller agent needs some kind of memory.

The learning of connection weights of the agents is achieved by genetic search where the fitness for an agent is set inversely proportional to the sum of horizontal separations between itself and each ball when the ball hits the ground. 10 percent of the best-performing agents in a population are selected for 1-point crossover with probability 0.9 and a mutation with the rate 0.04.

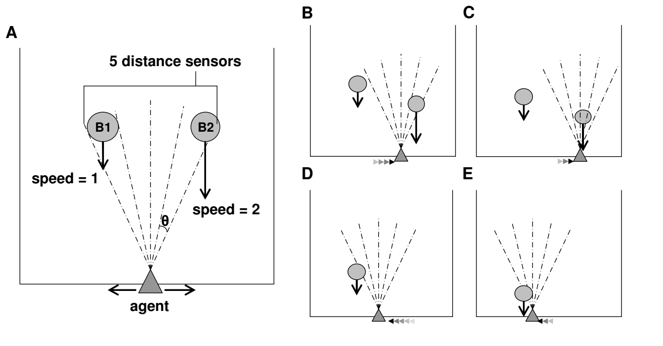

Fig. 1. **Ball Catching Task.** An illustration of the ball catching task is shown. The agent, equipped with a fixed number of range sensors (radiating lines), is allowed to move left or right at the bottom of the screen while trying to catch balls falling from the top. The goal is to catch both balls. The balls fall at different speeds, so a good strategy is to catch the fast-falling ball first (B and C) and then the go back and catch the slow one (D and E). Note that in C the ball on the left is outside of the range sensors' view. Thus, a memory-less agent would stop at this point and fail to catch the second ball.

B. Methods

In order to control the ball catcher agents, we used feedforward networks equipped with external marker droppers and detectors (Fig. 2, we will call this the "dropper network"). The agent had five range sensors that signal the distance to the ball when the ball comes into contact within the direct line-of-sight of the sensors. We used standard feedforward networks with sigmoidal activation units as a controller (see e.g., [2]):

$$H_j = \sigma\left(\sum_{i=1}^{N_{\text{in}}} v_{ji} I_i\right) \qquad j = 1, ..., N_{\text{hid}}$$
$$O_k = \sigma\left(\sum_{j=1}^{N_{\text{hid}}} w_{kj} H_j\right) \qquad k = 1, ..., N_{\text{out}} \quad (1)$$

where I_i, H_j and O_k are the activations of the i-th input, j-th hidden, and k-th output neurons; v_{ji} the input-to-hidden weights and w_{kj} the hidden-to-output weights; $\sigma(\cdot)$ the sigmoid activation function; and N_{in}, N_{hid}, and N_{out} are the number of input, hidden, and output neurons whose values are 7, 3, and 3 respectively.

The network parameters were tuned using genetic algorithms, thus the training did not involve any gradient-based adaptation. Two of the output units were used to determine the movement of the agent. If the agent was moved one step to the left when $O_1 > O_2$, one step to the right when $O_1 < O_2$, and remained in the current spot when $O_1 = O_2$.

If these were the only constructs in the controller, the controller will fail to catch multiple balls as in the case depicted in Fig. 1C. In order to solve this kind of problem, a fully recurrent network is needed, but from an evolutionary point of view, going from a feedforward neural circuit to a recurrent neural circuit could be nontrivial, thus our question was what could have been an easier route to memory-like behavior, without incurring much evolutionary overhead.

Our answer to this question is illustrated in Fig. 2. The architecture is inspired by primitive reactive animals that utilize self-generated chemical droppings (excretions, pheromones, etc.) and chemical sensors [18], [19], [20]. The idea is to maintain the reactive, feedforward network architecture, while adding a simple external mechanism that would incur only a small overhead in terms of implementation. As shown in Fig. 2, the feedforward network has two additional inputs for the detection of the external markers dropped in the environment, to the left or to the right (they work in a similar manner as the range sensors, signaling the distance to the markers). The network also has one additional output for making a decision whether to drop an external marker or not.

As a comparison, we also implemented a fully recurrent network, with multiple levels of delayed feedback into the hidden layer. (See [4], [5] for details.) This network was used to see how well our dropper network does in comparison to a fully memory-equipped network.

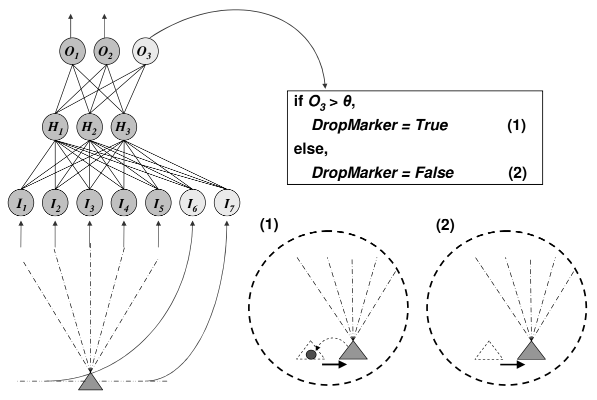

Fig. 2. **Feedforward Network with Dropper/Detector ("Dropper Network").** A feedforward network with a slight modification (dropper and detector) is shown. The basic internal architecture of the network is identical to any other feedforward network, with five range sensor (I_1 to I_5), and two output units that determine the movement (O_1 and O_2). The two added input units (I_6 and I_7) signal the presence of a dropped marker on the bottom plane, and the one additional output unit (O_3) makes the decision of whether to drop a marker at the current location or not. Note that there are no recurrent connections in the controller network itself.

C. Experiments and results

The network was trained using genetic algorithms (neuroevolution), where the connection weights and the dropper threshold θ were encoded in the chromosome. The fitness was inversely proportional to the sum of the distance between the agent and the ball(s) when the ball(s) contact the ground. Each individual was tested 12 times with different initial ball position (which was varied randomly) and speed (1 or 2 steps/time unit), and mixed scenarios with fast left ball vs. fast right ball. We used one-point crossover with probability 0.9, with a mutation rate of 0.04.

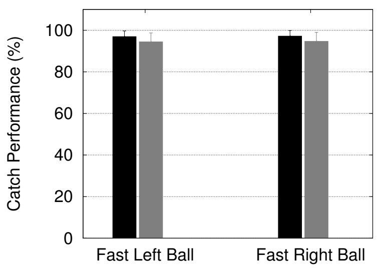

Fig. 3. **Ball Catching Performance.** The average ball catching performance of the dropper network is presented (gray bar), along with that of the recurrent network (black bar). The error bars indicate the standard deviation. The results are reported in two separate categories: fast left ball and fast right ball. This was to show that the network does not have any bias in performance. Both networks perform at the same high level (above 90% of all balls caught). This is quite remarkable for a feedforward network, although it had the added dropper/detector mechanism. We also tested a purely feedforward networks, but they were only able to catch 50% of the balls (catch one, miss one).

It is quite remarkable that feedforward networks can show an equal level of performance as that of the recurrent network, although the feedforward networks were equipped with the dropper/detector. For example, compared to the recurrent networks, the number of tunable parameters are meager for the dropper network since they do not have layers of fully connected feedback. Six additional weights for input-to-hidden, and three for hidden-to-output, plus a single threshold parameter (10 in all) is all that is needed.

One question arises from the results above. What kind of strategy is the dropper network using to achieve such a memory-like performance? We analyzed the trajectory and the dropping pattern, and found an interesting strategy that evolved. Fig. 4 shows some example trajectories. Here, we can see a curious *overshooting* behavior.

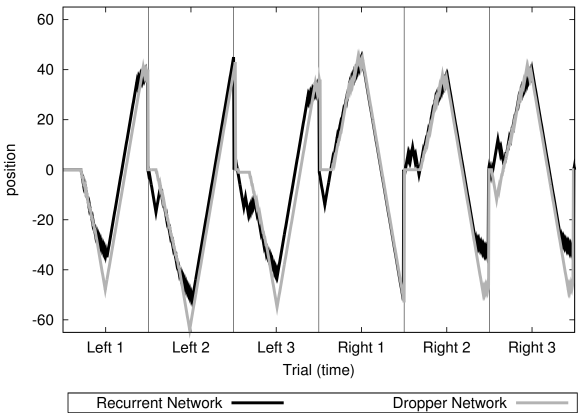

Fig. 4. **Agent Trajectory.** Agent trajectory during six ball catching trials are shown (gray: dropper network; black: recurrent network). The x axis represents time, and the y axis the agent position (0 marks the initial location of the agent). Within each trial, 200 time steps are shown. As the left and the right ball positions were randomized, the peak of the trajectories differ in their y values. The first three trials were the "fast left ball" condition and the last three were the "fast right ball" condition. Both networks are successful at catching both balls within each trial, but the dropper network shows a curious *overshooting* behavior (for example, near the half way point in each trial). See Fig. 5 for details.

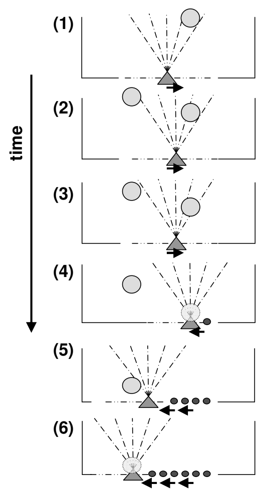

Fig. 5. **Dropper Network Strategy.** A strategy that evolved in the dropper network is shown. (1) Fast ball enters the view. (2&3) Agent moves toward the fast ball. (4) Agent catches fast ball, lose view of the slow ball, overshoots, and start dropping markers (black dots). (5) Seemingly repelled by the markers, the agent moves back to the slow ball, continuously dropping the markers, and (6) eventually catches it.

Fig. 5 shows how this overshooting behavior is relevant to the task, when combined with the dropping events. The strategy can be summarized as below: (1) The right ball fall fast, which is detected first. (2&3) The agent moves toward the right ball, eventually catching it (4). At this point, the left ball is outside of the range sensors' view, it overshoots the right ball, drops a marker there, and immediately returns back, seemingly repelled by the marker that has just been dropped. (5) The agents keeps on dropping the marker which pushing back to the left, until the left ball comes within the view of the range sensor. (6) The agent successfully catches the second ball. This kind of aversive behavior is quite the opposite of what we expected, but for this given task it seem to make pretty good sense, since in some way the agent is "remembering" which direction to avoid, rather than remembering where the slow ball was (compare to the "avoiding the past" strategy proposed in [21]).

III. Part II: Evolution of Prediction

In this second part, we will now examine how predictive capabilities could have emerged through evolution. Here, we use a recurrent neural network controller in a 2D pole-balancing task. Usually recurrent neural networks are associated with some kind of memory, i.e., an instrument to look back into the past. However, here we argue that it can also be seen as holding a predictive capacity, i.e., looking into the future. Below, we first describe the 2D pole-balancing task, and explain our methods, followed by experiments and results. The methods and results reported in this part are largely based on our earlier work [17].

A. Task: 2D Pole Balancing

Fig. 6 illustrates the standard 2D pole balancing task. The cart with a pole on top of it is supposed to be moved around while the pole is balanced upright. The whole event occurs within a limited 2D bound. A successful controller for the cart can balance the pole without making it fall, and without going out of the fixed bound. Thus, the pole angle, cart position, and their respective velocities become an important information in determining the cart's motion in the immediate next time step.

B. Methods

For this part, we evolved recurrent neural network controllers, as shown in Fig. 7A. The activation equation is the same as Eq. 1, and again, we used the same neuroevolution approach to tune the weights and other parameters in the model. One difference in this model was the inclusion of a facilitating dynamics in the neuronal activation level of the hidden units. Instead of using the H_j value directly, we used the facilitated value

$$A_j(t) = H_j(t) + r\left(H_j(t) - A_j(t-1)\right), \qquad (2)$$

where $H_j(t)$ is the hidden unit j's activation value at time t, $A_j(t)$ the facilitated hidden unit j's activation value, and r an evolvable facilitation rate parameter (see [22] for details). This formulation turned out to have a smoother characteristic,

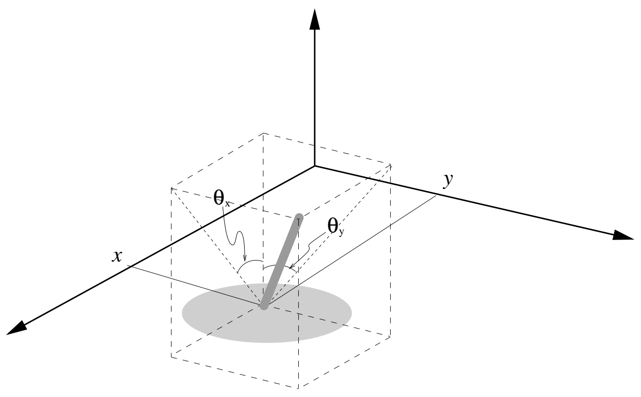

Fig. 6. **2D Pole-Balancing Task.** The 2D pole-balancing task is illustrated. The cart (gray disk) with an upright pole attached to it must move around on a 2D plane while keeping the pole balanced upright. The cart controller receives the location (x, y) of the cart, the pole angle (θ_x, θ_y), and their respective velocities as the input, and generates the force in the x and the y direction.

compared to our earlier facilitation dynamics in [23], [24], [25].

One key step in this part is to *measure* the predictability in the internal state dynamics. That is, given m past values of a hidden unit H_j (i.e., $\langle H_j(t-1), H_j(t-2), ..., H_j(t-m)\rangle$), how well can we predict $H_j(t)$. The reason for measuring this is to categorize individuals (evolved controller networks) that have a predictive potential and those that do not, and observe how they evolve. Our expectation is that individuals with more predictable internal state trajectory will have an evolutionary edge, thus opening the road for predictive functions to emerge. In order to have an objective measure, we trained a standard backpropagation network, with the past input vector $\langle H_j(t-1), H_j(t-2), ..., H_j(t-m)\rangle$ as the input and the current activation value $H_j(t)$ as the target value. Fig. 8 shows a sketch of this approach. With this, internal state trajectories that are smoother and easier to predict (Fig. 9A) will be easier to train, i.e., faster and more accurate, than those that are harder to predict (Fig. 9B). Note that the measured predictability is *not used as a fitness measure*. Predictability is only used as a post-hoc analysis. Again, the reason for measuring the predictability is to see how predictive capability can spontaneously emerge throughout evolution.

C. Experiments and results

Fig. 10 shows an overview of our experiment.

The pole balancing problem was set up within a 3 m $\times$ 3 m arena, and the output of the controller exerted force ranging from -10 N to 10 N. The pole was 0.5 m long, and the initial tilt of the pole was set randomly within 0.57° . We used neuroevolution (cf. [14]). Fitness was determined by the number of time steps the controller was able to balance the pole within $\pm 15^\circ$ from the vertical. Crossover was done with probability 0.7 and mutation added perturbation with a rate of ± 0.3. The force was applied at a 10 ms interval. The agent was deemed successful if it was able to balance the pole for 5,000 steps.

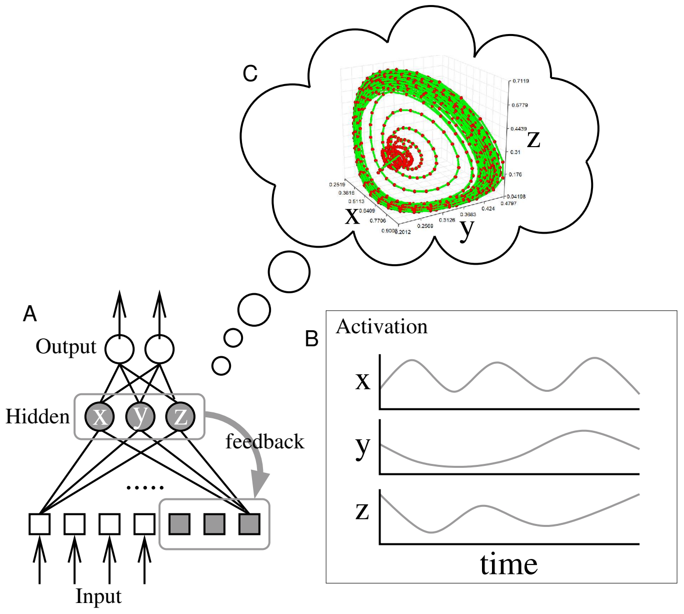

Fig. 7. **Cart Controller and Its Internal State.** A sketch of the cart controller network is shown. A. The network had 3 hidden units, which was fed back as the context input with a 1-step delay, to implement a recurrent architecture. The network had 8 inputs, each corresponding to the measures listed in Fig. 6. The two output units represents the force to be applied in the x and the y direction, respectively. B. The activity level of the hidden units can be seen as the agent's internal state, which in this case, can be plotted as a trajectory in 3D (see C).

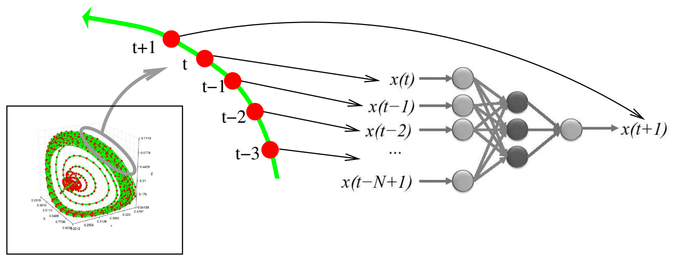

Fig. 8. **Measuring Predictability in the Internal State Trajectory.** A simple backpropagation network was used to measure the predictability of the internal state trajectory. A sliding window on the trajectory generated a series of input vectors (N past data points) and the target values (the current data point) to construct the training set. Those with a smoother trajectory would be easier to train, with higher accuracy.

For the backpropagation predictors, we took internal state trajectories from successful controllers, and generated a training set for supervised learning, using 3,000 data points in the trajectory data. We generated an additional 1,000 inputs for validation. Standard backpropagation was used, with a learning rate of 0.2. For each data point, if the error was within 10% of the actual value, we counted that as correct, and otherwise incorrect. With this, for each trajectory we were able to calculate the predictive accuracy.

We evolved a total of 130 successful individuals, and measured their internal state predictability. Fig. 11 shows the predictability in the 130 top individuals, which exhibits a smooth gradient. Among these, we selected the top 10 and the bottom 10, and further compared their performance. Note

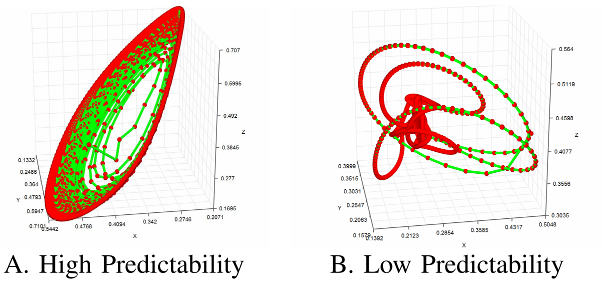

Fig. 9. **Internal State Trajectories.** Typical internal state trajectories from the hidden units of the controller networks are shown for A. the high predictability group and B. the low predictability group.

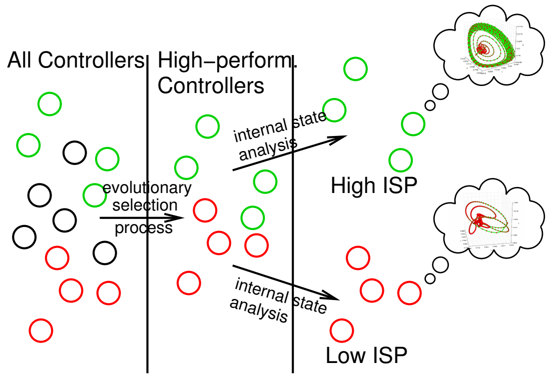

Fig. 10. **Overview of the Experiment.** An overview of the experiment is shown. First, high-performing individuals (capable of balancing the pole for over 5,000 steps) are collected throughout the generations. Next, the internal state predictability of the selected ones are measured to separate the group into high internal state predictability (High ISP) and low ISP groups. The High and Low ISP groups are subsequently tested in a tougher task.

that since all 130 had excellent performance, the 20 that are selected in this way by definition have the same level of performance. The trick here is to put those 20 controllers in a harsher environment, by making the pole balancing task harder. We increased the initial pole angle slightly to achieve this. The results are shown in Fig. 12. The results show that the high internal state predictability (high ISP) group outperforms the low internal state predictability (low ISP) group by a large margin. This is a surprising outcome, considering that the two types of networks (high ISP vs. low ISP) had the same level of performance in the task they were initially evolved in. This suggests that certain internal properties, although only internally scrutinizable at one time, can come out as an advantage as the environment changes. One interesting observation we made in our earlier paper [17] is that the high performance in the high ISP group is not due to the simpler, smoother internal state trajectory linearly carrying over into simpler, smoother behavior, thus giving it an edge in pole balancing. On the contrary, we found that in many cases, high ISP individuals had complex behavioral trajectories and vice versa (see [17] for details). In sum, these results show how *predictive* capabilities could have evolved in evolving neural networks.

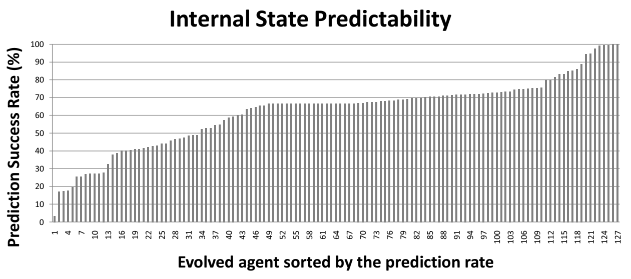

Fig. 11. **Internal State Predictability.** Internal state predictability of 130 successful controllers are shown, sorted in increasing order. Adapted from our earlier work [17].

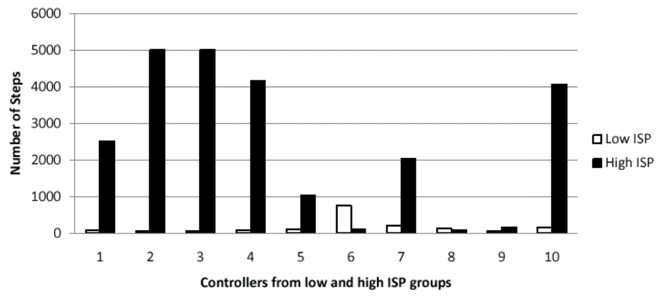

Fig. 12. **Pole Balancing Performance.** The performance (number of pole balancing steps) of the controller network is shown for the high ISP group (black bars) and the low ISP group (white bars). For this task the initial pole angle was increased to within $(\theta_x, \theta_y) = (0.14°, 0.08°)$. In all cases, the high ISP group does better, in many cases reaching the 5,000 performance mark, while those in the low ISP group show near zero performance. Note that these are new results, albeit being similar to our earlier results reported in [17].

IV. Discussion

The main contribution of this paper is as follows. We showed how recollection and prediction can evolve in neural circuits, thus linking the organism to its past and its future.

Our results in Part I suggest an interesting linkage between external memory and internalized memory (cf. [26], [27]). For example, humans and many other animals use external objects or certain substances excreted into the environment as a means for spatial memory (see [12] for theoretical insights on the benefit of the use of inert matter for cognition). In this case, olfaction (or other forms of chemical sense) serves an important role as the "detector". (Olfaction is one of the oldest sensory modalities, shared by most living organisms [28], [29], [30].) This form of spatial memory resides in the environment, thus it can be seen as external memory. On the other hand, in higher animals, spatial memory is also internalized, for example in the hippocampus. Interestingly there are several different clues that suggest an intimate relationship between the olfactory system and the hippocampus. They are located nearby in the brain, and genetically they seem to be closely related ([31], [32] showed that the Sonic Hedgehog gene controls the development of both the hippocampus and the olfactory bulb). Furthermore, neurogenesis is most often observed in the hippocampus and in the olfactory bulb, alluding to a close functional demand [33]. Finally, it is interesting to think of neuromodulators [34] as a form of internal marker dropping, in the fashion explored in this paper.

Prediction (or anticipation) is receiving much attention lately, being perceived as a primary function of the brain [35], [36] (also see [37] for an earlier discussion on anticipation). Part II of this paper raises interesting points of discussion regarding the origin and role of prediction in brain function. One interesting perspective we bring into this rich on-going discussion about prediction is the possible evolutionary origin of prediction. If there are agents that show the same level of behavioral performance but have different internal properties, why would evolution favor one over the other? That is, certain properties internal to the brain (like high ISP or low ISP) may not be visible to the external processes that drive evolution, and thus may not persist (cf. "philosophical zombies" [38]). However, our results show that certain properties can be latent, only to be discovered later on when the changing environment helps bring out the fitness value of those properties. Among these properties we found prediction.

There are several promising future directions. For Part I, recollection, it would be interesting to extend the task domain. One idea is to allow the agent to move in a 2D map, rather than on a straight line. We expect results comparable to those reported here, and also to those in [21]. Furthermore, actually modeling how the external memory became internalized would be an intriguing topic (a hint from the neuromodulation research such as [34] could provide the necessary insights). Insights gained from evolving an arbitrary neural network topology may also be helpful [39], [40]. As for Part II, prediction, it would be helpful if a separate subnetwork can actually be made to evolve to predict the internal state trajectory (as some kind of a monitoring process) and explicitly utilize the information.

V. Conclusion

In this paper we have shown how recollection and prediction could have evolved in neural network controllers embedded in a dynamic environment. Our main results are that recollection could have evolved when primitive feed-forward nervous systems were allowed to drop and detect external markers (such as chemicals), and that prediction could have evolved naturally as the environment changed and thus conferred a competitive edge to those better able to predict. We expect our results to provide unique insights into the emergence of time in neural networks and in the brain: recollection and prediction, past and future.

Acknowledgments

Sec. III was partly based on our earlier report in [17]. We would like to thank the anonymous reviewers who provided constructive criticism.

References

[1] C. M. Bishop, *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[2] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999.

[3] D. J. Felleman and D. C. V. Essen, “Distributed hierarchical processing in primate cerebral cortex,” *Cerebral Cortex*, vol. 1, pp. 1–47, 1991.

[4] J. L. Elman, “Finding structure in time,” *Cognitive Science*, vol. 14, pp. 179–211, 1990.

[5] ——, “Distributed representations, simple recurrent networks, and grammatical structure,” *Machine Learning*, vol. 7, pp. 195–225, 1991.

[6] R. D. Beer, “Dynamical approaches to cognitive science,” *Trends in Cognitive Sciences*, vol. 4, pp. 91–99, 2000.

[7] C. von der Malsburg and J. Buhmann, “Sensory segmentation with coupled neural oscillators,” *Biological Cybernetics*, vol. 67, pp. 233–242, 1992.

[8] Y. Choe and R. Miikkulainen, “Contour integration and segmentation in a self-organizing map of spiking neurons,” *Biological Cybernetics*, 2004, in press. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/choe.bc03.pdf

[9] R. Miikkulainen, J. A. Bednar, Y. Choe, and J. Sirosh, *Computational Maps in the Visual Cortex*. Berlin: Springer, 2005, uRL: http://www.computationalmaps.org.

[10] C. Peck, J. Kozloski, G. Cecchi, S. Hill, F. Schürmann, H. Markram, and R. Rao, “Network-related challenges and insights from neuroscience,” *Lecture Notes on Computer Science*, vol. 5151, pp. 67–78, 2008.

[11] R. Ward and R. Ward, “2006 special issue: Cognitive conflict without explicit conflict monitoring in a dynamical agent,” *Neural Networks*, vol. 19, no. 9, pp. 1430–1436, 2006.

[12] L. M. Rocha, “Eigenbehavior and symbols,” *Systems Research*, vol. 13, pp. 371–384, 1996.

[13] C. W. Anderson, “Learning to control an inverted pendulum using neural networks,” *IEEE Control Systems Magazine*, vol. 9, pp. 31–37, 1989.

[14] F. Gomez and R. Miikkulainen, “2-D pole-balancing with recurrent evolutionary networks,” in *Proceedings of the International Conference on Artificial Neural Networks*. Berlin; New York: Springer-Verlag, 1998, pp. 425–430.

[15] H. Lim and Y. Choe, “Facilitating neural dynamics for delay compensation and prediction in evolutionary neural networks,” in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO-2006*, M. Keijzer, Ed., 2006, pp. 167–174. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/lim.gecco06-reprint.pdf

[16] ——, “Compensating for neural transmission delay using extrapolatory neural activation in evolutionary neural networks,” *Neural Information Processing–Letters and Reviews*, vol. 10, pp. 147–161, 2006. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/lim.niplr06-reprint.pdf

[17] J. Kwon and Y. Choe, “Internal state predictability as an evolutionary precursor of self-awareness and agency,” in *Proceedings of the Seventh International Conference on Development and Learning*. IEEE, 2008, pp. 109–114. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/kwon.icdl08.pdf

[18] D. L. Wood, “The role of pheromones, kairomones, and allomones in the host selection and colonization behavior of bark beetles,” *Annual Review of Entomology*, vol. 27, pp. 411–446, 1982.

[19] J. A. Tillman, S. J. Seybold, R. A. Jurenka, and G. J. Blomquist, “Insect pheromones - an overview of biosynthesis and endocrine regulation,” *Insect Biochemistry and Molecular Biology*, vol. 29, pp. 481–514, 1999.

[20] M. R. Conover, *Predator-Prey Dynamics: The Role of Olfaction*. CRC Press, 2007.

[21] T. Balch, “Avoiding the past: a simple but effective strategy for reactive navigation,” in *Proceedings of the 1993 IEEE Ineternational Conference on Robotics and Automation*. IEEE, 1993, pp. 678–685.

[22] J. Kwon and Y. Choe, “Enhanced facilitatory neuronal dynamics for delay compensation,” in *Proceedings of the International Joint Conference on Neural Networks*. Piscataway, NJ: IEEE Press, 2007, pp. 2040–2045. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/kwon.ijcnn07-preprint.pdf

[23] H. Lim and Y. Choe, “Facilitatory neural activity compensating for neural delays as a potential cause of the flash-lag effect,” in *Proceedings of the International Joint Conference on Neural Networks*. Piscataway, NJ: IEEE Press, 2005, pp. 268–273. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/lim.ijcnn05-reprint.pdf

[24] ——, “Delay compensation through facilitating synapses and STDP: A neural basis for orientation flash-lag effect,” in *Proceedings of the International Joint Conference on Neural Networks*. Piscataway, NJ: IEEE Press, 2006, pp. 8385–8392. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/lim.ijcnn06.pdf

[25] ——, “Delay compensation through facilitating synapses and its relation to the flash-lag effect,” *IEEE Transactions on Neural Networks*, vol. 19, pp. 1678–1688, 2008. [Online]. Available: http://faculty.cs.tamu.edu/choe/ftp/publications/lim.tnn08-preprint.pdf

[26] A. Clark, *Supersizing the Mind: Embodiement, Action, and Cognition*, 2008.

[27] M. T. Turvey and R. Shaw, “The primacy of perceiving: An ecological reformulation of perception for understanding memory,” in *Perspectives on Memory Research: Essays in Honor of Uppsala University’s 500th Anniversary*, L.-G. Nilsson, Ed. Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers, 1979, ch. 9, pp. 167–222.

[28] J. G. Hildebrand, “Analysis of chemical signals by nervous systems,” *Proceedings of National Academy of Sciences, USA*, vol. 92, pp. 67–74, 1995.

[29] P. Vanderhaeghen, S. Schurmans, G. Vassart, and M. Parmentier, “Specific repertoire of olfactory receptor genes in the male germ cells of several mammalian species,” *Genomics*, vol. 39, pp. 239–246, 1997.

[30] G. O. Mackie, “Central circuitry in the jellyfish aglantha digitale iv. pathways coordinating feeding behaviour,” *The Journal of Experimental Biology*, vol. 206, pp. 2487–2505, 2003.

[31] R. Machold, S. Hayashi, M. Rutlin, M. D. Muzumdar, S. Nery, J. G. Corbin, A. Gritli-Linde, T. Dellovade, J. A. Porter, S. L. Rubin, H. Dudek, A. P. McMahon, and G. Fishell, “Sonic hedgehog is required for progenitor cell maintenance in telencephalic stem cell niches,” *Neuron*, vol. 39, pp. 937–950, 2003.

[32] V. Palma, D. A. Lim, N. Dahmane, P. Sánchez, T. C. Brionne, C. D. Herzberg, Y. Gitton, A. Carleton, A. Álvarez Buylla, and A. R. Altaba, “Sonic hedgehog controls stem cell behavior in the postnatal and adult brain,” *Development*, vol. 132, pp. 335–344, 2004.

[33] J. Frisén, C. B. Johansson, C. Lothian, and U. Lendahl, “Central nervous system stem cells in the embryo and adult,” *CMLS, Cellular and Molecular Life Science*, vol. 54, pp. 935–945, 1998.

[34] J. L. Krichmar, “The neuromodulatory system: A framework for survival and adaptive behavior in a challenging world,” *Adaptive Behavior*, vol. 16, pp. 385–399, 2008.

[35] R. R. Llinás, *I of the Vortex*. Cambridge, MA: MIT Press, 2001.

[36] J. Hawkins and S. Blakeslee, *On Intelligence*, 1st ed. New York: Henry Holt and Company, 2004.

[37] R. Rosen, *Anticipatory Systems: Philosophical, Mathematical and Methodological Foundations*. New York: Pergamon Press, 1985.

[38] D. J. Chalmers, *The Conscious Mind: In Search of a Fundamental Theory*. New York and Oxford: Oxford University Press, 1996.

[39] K. O. Stanley and R. Miikkulainen, “Evolving neural networks through augmenting topologies,” *Evolutionary Computation*, vol. 10, pp. 99–127, 2002.

[40] ——, “Efficient evolution of neural network topologies,” in *Proceedings of the 2002 Congress on Evolutionary Computation (CEC’02)*. Piscataway, NJ: IEEE, 2002, in press.