Evolution and Self-teaching in Neural Networks

Another comparison when the agent is more primitively conscious

Nam Le
Natural Computing Research & Applications
University College Dublin
Dublin, Ireland
nam.lehai@ucdconnect.ie

ABSTRACT

Previous work presented a technique called evolving self-taught neural networks – neural networks that can teach themselves, intrinsically motivated, without external supervision or reward [3]. In an autonomous multi-agent setting in which the agent is *primitively* set to know little or nothing about its environment, self-teaching was shown to give rise to intelligence, whereas an evolutionary algorithm alone fails since it has no way to search without gradient information. In this paper, we conduct another comparative experiment in which the foraging agent is *built* more *conscious* of its environment beforehand. Experimental results show that the more conscious primitive design can let evolution alone be able to search. Yet the combination of evolution and self-teaching still outperforms the alternative. Indications for future work on evolving intelligence are also presented.

KEYWORDS

Neural Networks, Neuroevolution, Self-learning

ACM Reference Format:
Nam Le. 2019. Evolution and Self-teaching in Neural Networks: Another comparison when the agent is more primitively conscious. In *Proceedings of the Genetic and Evolutionary Computation Conference 2019 (GECCO '19).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3319619.3326904

1 INTRODUCTION

There are two ways in which the organism can adapt to its environment. The first is through biological evolution, or phylogenetic adaptation – this is a change at genetic level of a population. The second is ontogenetic adaptation, which includes learning – this is a change at a phenotypic level of an individual organism.

Taking inspiration from nature, evolution and learning from experience are can be used as two metaphors to create adaptive neural networks [12]. Interestingly, learning when combined with evolution can promote an evolving population better than evolution alone through the so-called **Baldwin Effect** [2, 4, 5, 7, 8]. This idea has also been employed in evolving neural networks. Exemplar studies include [10], [1], in which the combination of evolutionary search and backpropagation [11] is shown to create more adaptive neural networks. One shared feature of these studies is that each neural network agent stays in isolation, having no interaction.

Taking on this line of thought and with the perspective of artificial general intelligence (AGI), previous work proposed a technique called evolving self-taught neural networks, presenting an interplay between evolution and self-teaching to develop autonomous agents [3]. The proposed contribution is a technique that evolves intrinsically motivated self-supervised neural-networks, a step toward unsupervised learning or learning without (or with very few) labeled data and without (or very few) a priori knowledge. In [3], the foraging agent is designed *unconscious* of its environment and has to forage for food located very far from it initially. Unlike other studies in evolutionary robotics [9] or reinforcment learning [13], the agent (or robot) in [3] does not know anything about its relationship with the environment like the current position, the relative angle towards to food source, etc. It was shown that an evolutionary algorithm alone failed, whereas some degree of conscious and autonomous intelligence was demonstrated to emerge from the interaction between evolution and self-teaching [3].

The finding that evolution alone fails completely to create adaptive agents in [3] seems contradictory with most studies on evolutionary robotics and artificial life [9]. I hypothesise that the degree of *priori* knowledge in the primitive design of the agent relative to its environment matters here. In this paper, I extend previous study in [3] to develop another set of comparative experiments, in which the agent is designed with more primitive knowledge about its environment, namely the current angle and the relative position in the environment. Our simulations are described in the following section. We shall be seeing a more conscious primitive design allows evolution to create agents able to forage for food. Yet evolution and self-teaching still interact to create more adaptive autonomous agents than evolution alone.

2 SIMULATION SETUP

2.1 The Simulated World

Suppose that 20 agents are situated in a continuous 640x640 2D-world, called **MiniWorld**. Agents seek to find resources to feed themselves in order to survive. The world map used in our simulations as described in Figure 1a. It should be noted that the agent here is designed more environmentally conscious beforehand compared to previous study [3], as it knows more about its environment (its current position and angle). This is what I call *primitive consciousness* of its environment.

All agents in the population live in the same MiniWorld and their behaviours interact. As an agent finds and consumes food particles,

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
GECCO '19, July 13–17, 2019, Prague, Czech Republic
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326904

it changes the environment in which other agents live, forming a more complex dynamics. It can be the case that the more an agent eats, the less the chance for others to feed themselves. The default velocity (or speed) of each agent is 1.

Every agent has three basic movements: Turn left by 9 degrees and move; move forward by double speed; or turn right by 9 degrees and move. For simplicity, these rules are pre-defined by the system designer of MiniWorld. The motor action of an agent is guided by its neural network as described in each simulation below.

2.2 Simulation 1: Evolution alone (EVO)

In this simulation, we evolve a population of agents which do not have a lifetime learning capability. The neural controller is described in Figure 1c.

Selection chooses individuals based on the number of food particles consumed. The higher the number of particles eaten, the higher the agent's fitness value. For crossover, two individuals are selected to produce one offspring. We implement crossover as follows. The more successful a parent, the greater the likelihood that its weights are copied to the child. Each weight element in the matrix of the child network is copied from the fitter parent if the random probability is greater than 0.5, and vice versa. Once a child has been created, that child will be mutated based on a predefined *mutation rate* of 0.05. Mutation occurs at a specific weight as a random number is added to that weight. After that, the newly born individual is placed in the new population. This process is repeated until the new population is filled with 20 new individual agents. No elitism is employed in our evolutionary algorithm.

The population goes through a total of 100 generations, with 5000 time steps per generation. At each time step, an agent undertakes the following activities: Perceiving MiniWorld through its sensors, computing its motor outputs from its sensory outputs, moving in the environment which then updates its new heading and location. In evolution alone simulation, the agent cannot perform any kind of learning during its lifetime. After that, the population undergoes selection and reproduction processes.

2.3 Simulation 2: Evolving Self-taught agents

In this simulation, we allow lifetime learning, in addition to the evolutionary algorithm, to update the weights of neural network controllers when agents interact with the environment. We evolve a population of **Self-taught** agents – agents that can teach themselves. The self-taught agent has a self-taught neural network architecture as described in Figure 1d.

We use the same parameter setting for evolution as in EVO simulation above. At each time step, an agent does the following activities: Perceiving MiniWorld through its sensors, computing its motor outputs from its sensory outputs, moving in the environment which then updates its new heading and location, and updating the weights in action module by **self-teaching**. After one step, the agent updates its fitness by the number of food particles consumed. After that, the population undergoes selection and reproduction processes as in the Evolution alone simulation. [1]

[1] In these experiments, we implement learning and evolution in a Darwinian, not a Lamarckian framework. This means that the lifetime learning of an agent (the weights in its action module) is not passed down to its offspring.

2.4 Simulation 3: Self-taught agents alone

We conduct another simulation in which all agents are self-taught agents – having self-taught networks that can teach themselves during lifetime. What differs from simulation 2 is that at the beginning of every generation, all weights are randomly initialised, rather than updated by an evolutionary algorithm like in simulation 1. The learning agents here are initialised as *blank-slates*, or *tabula rasa*, having no predisposition to learn or some sort of *priori knowledge* about the world. The reason for this simulation is that we are curious whether evolution brings any benefit to learning in MiniWorld. In other words, we would like to see if there is a synergy between evolution and learning, or self-teaching here.

Experimental results are discussed in the following section.

3 RESULTS AND DISCUSSION

One notable point here is that evolved agents without learning here can still search for food, unlike in previous study [3]. This can be explained by the fact that the agent in the current contribution is designed more conscious of its environment by the human engineer. It knows its current position and angle in the environment, thus having more relevant sensory information to drive its movement. Conversely, the agent in [3] has no priori and relational knowledge about its environment at all, it has to be getting more conscious over time to survive.

First compare the performance between the baseline EVO alone with EVO+Self-taught. It can be seen in Figure 2 and 3 that the whole EVO+Self-taught population outperforms EVO alone in terms of the average fitness. This could be explained by the effect of individual learning on evolution, or the *Baldwin Effect* [6]. Yet there is a difference but not that significant in terms of the best fitness. In biology, even a little different still implies something interesting. What has been observed here implies that there are more agents in EVO+Self-taught having adaptive movements towards the food source than in EVO alone. Thus, the best agent in EVO alone has less competitive pressure and more free to eat, whereas the best evolved self-taught agent has more adaptive competitors. Therefore, even the difference is not that significant, evolution and self-teaching is supposed to create the better best agent.

Another curious question here is whether evolution facilitates learning? It can be observed in Figure 2 that EVO+Self-taught outperforms the Self-taught alone in both best and average fitness, and the box-plot in Figure 3 shows that the difference is significant. One interesting point here is that when agent is designed more conscious beforehand, the EVO alone outperforms the self-taught alone population in MiniWorld.

It is plausible here to conclude that even when the agent is designed more conscious of its environment by the human engineer, the evolved self-taught neural network still presents better adaptive behaviour than evolution and self-teaching alone. This, again, presents an effect resulted from the interaction between evolution and self-teaching.

4 CONCLUSION AND FUTURE WORK

We have analysed and extended previous study on evolving self-taught neural networks towards autonomous intelligence [3], by putting more human-engineered domain knowledge into the design

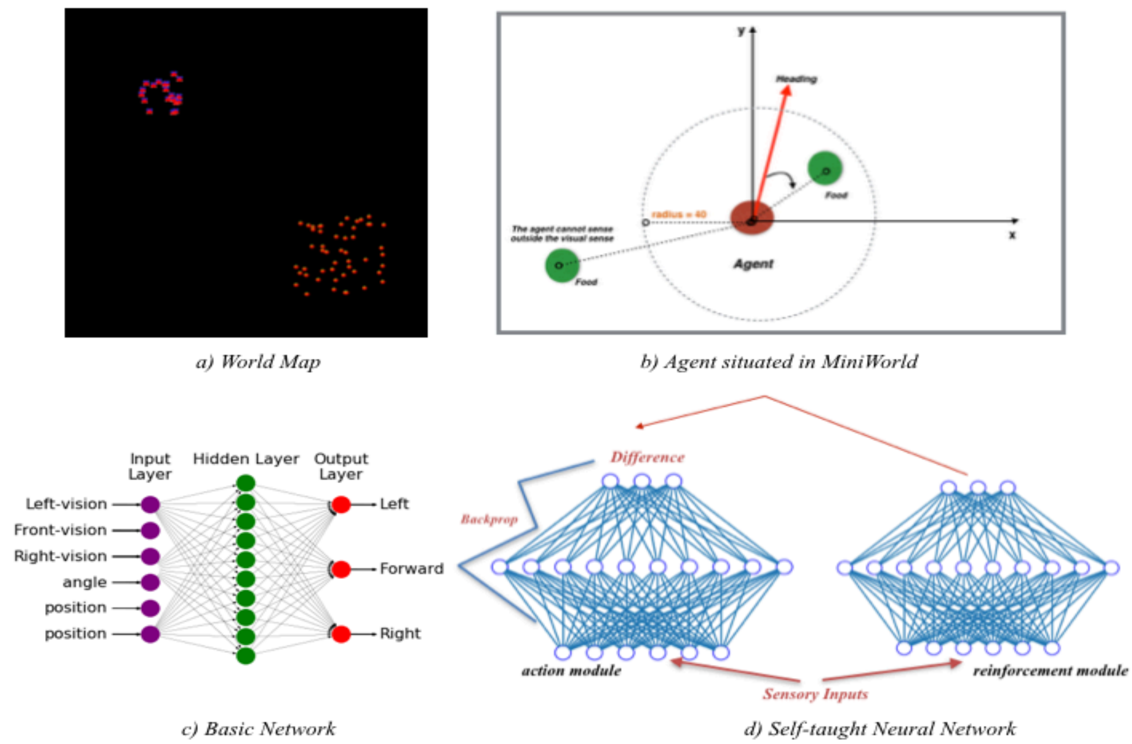

Figure 1: **MiniWorld – The environment of agents and food, 640x640. 50 food particles are randomly dispersed and each particle is represented by a square image with size 10x10. Each agent in MiniWorld also has a similar size.**
a) Denote w and h as the width and the height of MiniWorld. Initially all agents are located in a radius of 40 (4 times the size of an agent) around a central point: $(w/4, h/4)$. The food has its horizontal and vertical dimensions randomly chosen in the range $(5w/8, 7w/8)$ and $(5h/8, 7h/8)$, accordingly.
When an agent's body happens to collide with a food particle, the food particle is "eaten", the energy level of the agent increases by 1, and another food piece is randomly spawned in the same region but at a different location. The collision detection criterion is specified by the distance between the two bodies (of the agent and of the food particle). The environment changes as an agent eats a particle. Each agent has a heading (in principle) of movement in the environment. Rather than initialising all agents with random headings, all the agents are initialised with a horizontal heading (i.e,. with a heading of 0 degrees). This somewhat explains the purpose of the design the map. Agents are born with facing away from the food source resulting in a more difficult environment.
b) Assume that every agent has an a priori ability to sense the angle between its current heading and the food if this appears in its visual range. The visual range of each agent is a circle with radius 40. Each agent takes as inputs, 6 pieces of sensory information. The first three bits (left, front, right) are set to 0 or 1 depending on whether the substance appears (in the left, front, and right) or not. Let θ (in degree) be the angle between the agent and the substance in its visual sense. An agent determines whether a food appears in the left, front, or right side in its visual range using the following rule: Right if $15 < \theta < 45$; Front if $\theta < 15$ or $\theta > 345$; and Left if $315 < \theta < 345$. The other three sensory inputs are the current angle and the x and y dimensions in MiniWorld, all normalised in [0, 1].
c) Basic network without learning. Each neural network includes 3 layers with 6 input nodes, 10 hidden nodes, and 3 output nodes. The first layer takes as input what an agent senses from the environment in its visual range. The output layer produces 3 values in which the max value is chosen as a motor-guidance.The *genotype* of each agent is the weight matrix of its neural network, and the evolutionary process takes place as we evolve a population of weights.
d) Self-taught neural architecture. The difference between the output of the reinforcement module and the action module is used to update the weights in action modules through back-propagation. Through this self-teaching process, the action module approximates its output activation towards the output of the reinforcement module. The learning rate is 0.01. During the lifetime of an agent, the reinforcement modules produce outputs in order to guide the weight-updating process of the action module. Only the weights of action modules can be changed by learning, the weights of reinforcement module are genetically specified in the same evolutionary process as specified above in Evolution alone simulation.

of the agent to make it more conscious beforehand. Experimental results have shown that the conscious design has an effect on evolutionary fitness of the agent, yet the evolved self-taught agents still outperform both evolution and self-teaching in isolation. The interplay between learning and evolution has also been demonstrated in the sense that evolution learning guides the evolutionary search, and evolution facilitates future self-teaching, better than *blank-slates*.

There is several avenues for future research, in different directions. The idea of self-taught neural networks can be powerful when there is no external supervision (or *label* provided from external data. The algorithm and technique used in this paper can also be a potential technique to solve unsupervised learning, or learning with limited label data (weak supervision, especially in reinforcement learning and games. Indeed, the shallow network used in this paper does not restrict the application of the core philosophical idea into deep neural networks, as long as we can combine evolutionary search and the idea of self-taught neural architecture by employing variants of gradient-based learning.

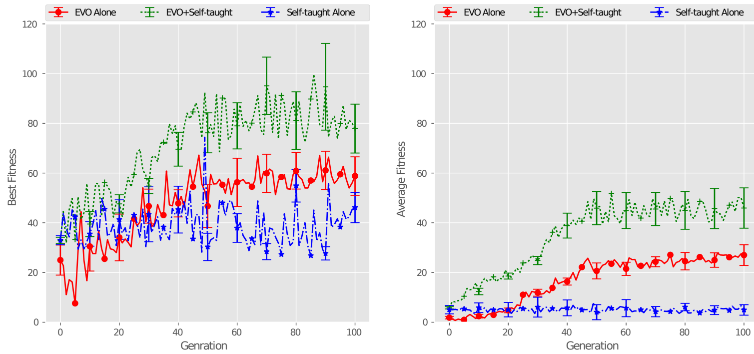

Figure 2: Fitness comparison. Left: Best, Right: Average

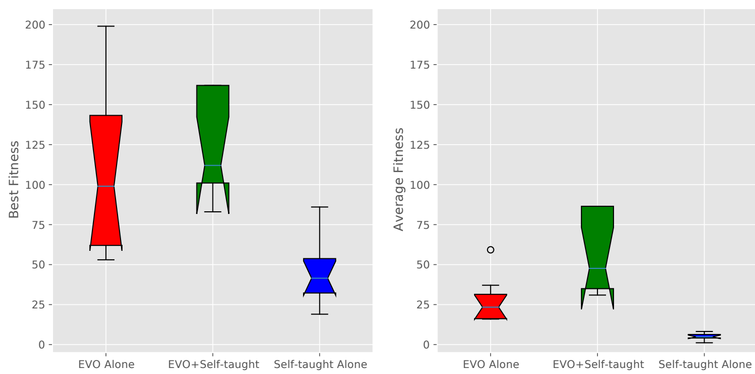

Figure 3: Fitness Boxplot. Left: Best, Right: Average

Towards the creation of autonomous and general intelligence, unlike traditional computational models in reinforcement learning [13] and evolutionary robotics [9], I propose that the agent should be equipped with little or no *priori* knowledge about its environment (like in [3]), and let that of the relational understanding between the agent and its environment emerge as part of conscious intelligence. The idea of evolving better self-taught agent can be an interesting attempt to create the emergence of autonomous (or conscious) intelligence, from the agent-perspective, in that sense, without human-designed rewards or knowledge.

ACKNOWLEDGMENT

This research is funded by the Science Foundation Ireland under Grant No. 13/IA/1850.

REFERENCES

[1] Chrisantha Thomas Fernando, Jakub Sygnowski, Simon Osindero, Jane Wang, Tom Schaul, Denis Teplyashin, Pablo Sprechmann, Alexander Pritzel, and Andrei A. Rusu. 2018. Meta-Learning by the Baldwin Effect. *CoRR* abs/1806.07917 (2018). arXiv:1806.07917 http://arxiv.org/abs/1806.07917

[2] Geoffrey E. Hinton and Steven J. Nowlan. 1987. How Learning Can Guide Evolution. *Complex Systems* 1 (1987), 495–502.

[3] Nam Le. 2019. Evolving Self-taught Neural Networks: The Baldwin Effect and the Emergence of Intelligence. In *10th Symposium on AI & Games, in AISB Convention 2019*. Falmouth, UK. http://aisb2019.falmouthgamesacademy.com/wp-content/uploads/2019/04/AISB-AI-AND-Games2019_proceedings.pdf

[4] Nam Le, Anthony Brabazon, and Michael O'Neill. 2018. How the "Baldwin Effect" Can Guide Evolution in Dynamic Environments. In *Theory and Practice of Natural Computing*. Springer International Publishing, 164–175. https://doi.org/10.1007/978-3-030-04070-3_13

[5] Nam Le, Michael O'Neill, and Anthony Brabazon. 2018. Adaptive Advantage of Learning Strategies: A Study Through Dynamic Landscape. In *Parallel Problem Solving from Nature – PPSN XV*, Anne Auger, Carlos M. Fonseca, Nuno Lourenço, Penousal Machado, Luís Paquete, and Darrell Whitley (Eds.). Springer International Publishing, Cham, 387–398.

[6] N. Le, M. O'Neill, and A. Brabazon. 2018. The Baldwin Effect Reconsidered Through the Prism of Social Learning. In *2018 IEEE Congress on Evolutionary Computation (CEC)*. 1–8. https://doi.org/10.1109/CEC.2018.8477654

[7] Nam Le, Michael O'Neill, and Anthony Brabazon. 2019 forthcoming. Evolutionary Consequences of Learning Strategies in a Dynamic Rugged Landscape. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '19)*. ACM, Prague, Czech Republic.

[8] Nam Le, Michael O'Neill, and Anthony Brabazon. 2019 forthcoming. How Learning Strategies Can Promote an Evolving Population in Dynamic Environments. In *IEEE Congress on Evolutionary Computation, CEC 2019*. IEEE Press, Wellington, New Zealand.

[9] S. Nolfi, D. Floreano, and D.D. Floreano. 2000. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-organizing Machines*. MIT Press.

[10] Stefano Nolfi, Domenico Parisi, and Jeffrey L. Elman. 1994. Learning and Evolution in Neural Networks. *Adaptive Behavior* 3, 1 (1994), 5–28.

[11] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323, 6088 (oct 1986), 533–536. https://doi.org/10.1038/323533a0

[12] Andrea Soltoggio, Kenneth O. Stanley, and Sebastian Risi. 2018. Born to learn: The inspiration, progress, and future of evolved plastic artificial neural networks. *Neural Networks* 108 (dec 2018), 48–67.

[13] R.S. Sutton, A.G. Barto, and F. Bach. 2018. *Reinforcement Learning: An Introduction*. MIT Press. https://books.google.ie/books?id=6DKPtQEACAAJ