Evolving Recurrent Neural Networks for Emergent Communication

Joshua Sirota
sirota@ualberta.ca
Department of Computing Science, University of Alberta, Edmonton, Alberta, T6G 2E8, CANADA

Vadim Bulitko
bulitko@ualberta.ca
Department of Computing Science, University of Alberta, Edmonton, Alberta, T6G 2E8, CANADA

Matthew R G Brown
mbrown2@ualberta.ca
Department of Psychiatry, University of Alberta, Edmonton, Alberta, T6G 2E8, CANADA

Sergio Poo Hernandez
pooherna@ualberta.ca
Department of Computing Science, University of Alberta, Edmonton, Alberta, T6G 2E8, CANADA

ABSTRACT

Recent research showed that deep neural networks can be trained to create shared languages to communicate and cooperate with each other. These approaches used fixed, handcrafted network architectures which were trained with reinforcement learning. We extend this approach by using neuroevolution to automate network design and find network weights of communicating agents. We show that neuroevolution is a viable approach for training agents to develop novel languages so as to communicate amongst themselves.

ACM Reference Format:
Joshua Sirota, Vadim Bulitko, Matthew R G Brown, and Sergio Poo Hernandez. 2019. Evolving Recurrent Neural Networks for Emergent Communication. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3321957

1 INTRODUCTION

Emergent communication (EC) is an active research area of Artificial Intelligence with exciting results in facilitating cooperation between artificial agents [2, 3, 5].

Recent work [2] showed that it is possible to train two agents to develop a language from scratch so as to successfully play a game requiring communication between agents. Additionally, EC has been used to facilitate cooperation between agents in multi-agent environments [3, 5]. The communicating agents in these papers are fixed, hand-crafted neural networks which are trained with reinforcement learning (RL) [2]. Recent work in evolutionary computation, [6] though, has demonstrated that evolving the architecture of neural networks can improve their performance. As such, current approaches could be limiting the success of communicating agents by restricting their architecture to one chosen by human developers.

We propose to evolve the architecture of deep neural networks and train these networks to develop their own shared languages. To update these network's weights we will be using evolutionary search (ES) [8]. Our reasoning for using ES is twofold: ES allows us (i) to quickly evaluate many agents which facilitates a wide exploration through architecture space, and (ii) to explore whether techniques other than RL are viable for training deep neural networks to communicate.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3321957

2 PROBLEM FORMULATION

In this paper we address the problem of training agents to develop their own languages so as to cooperate with each other. In line with previous work [2], we formalize the problem as a repeated referential game played by agents. In these games a speaker agent A_S is shown some target object. A listener agent A_L is shown the target and some distracting objects. The speaker then sends messages to the listener which the listener uses to attempt to choose the target object rather than a distracting object.

Our objective in this paper is to maximize the referential game success rate of agent pairs when playing the referential game where the objects being referred to are previously unseen. For a repeated referential game over T rounds, we define the success rate of an agent pair (A_S, A_L) as the proportion of rounds in which A_L correctly chooses the target object.

3 PROPOSED APPROACH

Speakers A_S and listeners A_L are both recurrent neural networks (RNNs). While we periodically mutate the architecture of A_S and A_L, an initial architecture for the first generation is needed. We test two initial RNN architectures: a gated recurrent unit (GRU) [1], and a small RNN with randomized architecture. We selected these initial architectures to test, respectively, whether neuroevolution could improve the performance of a well known RNN architecture, and whether an RNN with no specific engineering could be evolved to communicate effectively.

Agents A_S and A_L are each equipped with their own feedforward neural networks F_S and F_L, respectively. These feedforward networks are used to preprocess target and distractor objects before they are inputted to their respective RNN. The networks F_S and F_L both have a single hidden layer and use sigmoid activation.

In this paper we use evolutionary search [4, 8] to generate the architecture of A_S and A_L as well as update the weights of A_S, A_L, F_S, and F_L. For this evolutionary search, generations of speaker-listener pairs (A_S, A_L) are evaluated on their performance at a repeated referential game. Each generation, the best performing

pairs of that generation reproduce to create the next generation. For a given agent pair (A_S, A_L), reproduction consists of two parts: (i) the weights of A_S, A_L, F_S, and F_L are mutated with Gaussian noise, (ii) the architecture of A_S and A_L is mutated. Possible architectural mutations are (i) operator swapping (e.g., changing one of the RNN's addition gates to a multiplication gate), (ii) operator insertion (e.g., inserting a tanh gate or a multiplication gate), (iii) operator removal (e.g., removing an addition gate and replacing it with one of its arguments, or removing a sigmoid gate).

4 EMPIRICAL EVALUATION

We used the Visual Attributes for Concepts Dataset [7]. This dataset contains 500 concepts (e.g., dog) in 16 categories (e.g., animals). For each concept, there are human-generated annotations describing its characteristics (e.g., has_fur).

We compared the random RNN and GRU initial architectures, each with and without architectural mutation. This resulted in four experimental conditions. We call the two conditions involving a random RNN **RT+E** and **RT**, denoting respectively the condition for which architectures are evolved and the condition for which they are not. Similarly, we call the GRU conditions **GRU+E** and **GRU**.

To evaluate our experimental conditions we created 13 random partitions of the dataset into three disjoint subsets: a training set, validation set, and test set. For each of the four experimental conditions we executed one evolution run using each of the 13 random partitionings for a total of 52 runs. For a given evolution run, every generation, all agent pairs played the referential game with that run's training set to choose reproducing pairs. The best performing pairs of each generation, in addition to reproducing, played the referential game again with that run's validation set. Over a given run we stored the agent pair which performed best on that run's validation set and, after the run had finished, tested that best-performing pair's success at the referential game with that run's test set. As 13 runs were performed for each condition, this resulted in 13 agent pairs tested on the test set for each condition. The performance measure by which we compared the four experimental conditions was the mean test set performance of the 13 top-performing agent pairs chosen from that condition's evolution runs.

Table 1: Mean and maximum success rates over 13 trials in the referential game when referring to objects drawn from a test set for each experimental condition.

	Experimental condition			
Success rate	**RT+E**	**RT**	**GRU+E**	**GRU**
Mean	64.9%	63.8%	**66.4%**	**66.4%**
Maximum	71.0%	68.9%	70.9%	**71.1%**

Results

The first and second rows of Table 1 show, respectively, the mean and maximum success rate of each condition's 13 agent pairs which were tested on their run's test sets. Each of these agent pairs are the pair that performed best on their run's validation set. We used one distractor, so if agents guessed randomly then their expected success rate would be 50% (the baseline). All experimental conditions achieved accuracy significantly higher than the baseline (one-tailed z-test, $p = 10^{-16}$ for all conditions). The highest mean success rate was achieved by both **GRU+E** and **GRU**. The highest maximum success rate was achieved by condition **GRU**. Both GRU conditions significantly outperformed both random tree conditions (two-tailed z-test, $p \leq 5.9 \times 10^{-5}$). Condition **RT+E** significantly outperformed condition **RT** (two-tailed z-test, $p \leq 5.4 \times 10^{-3}$).

5 CONCLUSION

We showed that evolutionary search is a viable technique for emergent communication. This can be seen since experimental condition **RT+E** significantly outperformed condition **RT**, providing evidence that evolutionary search for agent architectures can benefit communication. As all four experimental conditions performed well above baseline, we also see that evolutionary search over the parameters of two neural networks can be used to train those networks to develop a shared emergent language. The performance of our agents was significantly worse than that achieved with deep reinforcement learning [2], with our agents achieving a success rate of only 71.1% on the referential game, while prior work achieved success rates over 90% on more difficult versions of the same game. The work in this paper was preliminary, though, and motivates continued exploration of the value of evolutionary computation for emergent communication. In particular, we believe that this work would benefit most from more advanced neuroevolution techniques (e.g., speciation) as well as formal analysis of evolved speaker and listener architectures.

6 ACKNOWLEDGEMENTS

We appreciate support from the NSERC, Nvidia, and Compute Canada.

REFERENCES

[1] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. [n. d.]. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR* abs/1406.1078 ([n. d.]). arXiv:1406.1078 http://arxiv.org/abs/1406.1078

[2] Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. 2018. Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input. *CoRR* abs/1804.03984 (2018). arXiv:1804.03984

[3] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *CoRR* abs/1706.02275 (2017). arXiv:1706.02275

[4] Risto Miikkulainen, Jason Zhi Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. 2017. Evolving Deep Neural Networks. *CoRR* abs/1703.00548 (2017). arXiv:1703.00548

[5] Igor Mordatch and Pieter Abbeel. 2017. Emergence of Grounded Compositional Language in Multi-Agent Populations. *CoRR* abs/1703.04908 (2017). arXiv:1703.04908

[6] Aditya Rawal and Risto Miikkulainen. 2018. From Nodes to Networks: Evolving Recurrent Neural Networks. *CoRR* abs/1803.04439 (2018). arXiv:1803.04439

[7] Carina Silberer, Vittorio Ferrari, and Mirella Lapata. 2013. Models of semantic representation with visual attributes. In *ACL (Volume 1: Long Papers)*, Vol. 1. 572–582.

[8] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. 2017. Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning. *CoRR* abs/1712.06567 (2017). arXiv:1712.06567