Collaborative Training of Multiple Autonomous Agents

Doctoral Consortium

Filippos Christianos
University of Edinburgh, Edinburgh, United Kingdom, f.christianos@ed.ac.uk

ABSTRACT

Exploration in multi-agent reinforcement learning is a challenging problem, especially with a large number of agents. Parameter sharing between agents is often used since it significantly decreases the number of trainable parameters, shortening training times to tractable levels and improving exploration efficiency. We present two algorithms that aim to be a middle ground between not sharing parameters and fully sharing parameters. These proposed algorithms show advantages of the baselines at the two ends of the spectrum and minimise their drawbacks. First, Shared Experience Actor-Critic [3], applies the basic idea of off-policy correction via importance weighting and combines the experiences generated by different agents into more informative and effective learning gradients. Then, Selective Parameter Sharing [2], based on rigorous empirical analysis of the impact of parameter sharing proposes a novel parameter sharing method that can be coupled with existing multi-agent reinforcement learning algorithms.

KEYWORDS

Reinforcement Learning, Multi-Agent Systems, Parameter Sharing

ACM Reference Format:
Filippos Christianos. 2022. Collaborative Training of Multiple Autonomous Agents: Doctoral Consortium. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 3 pages.

1 INTRODUCTION

Multi-agent reinforcement learning (MARL) necessitates exploration of the environment dynamics and of the joint action space between agents. This is a difficult problem due to non-stationarity caused by concurrently learning agents [7] and the fact that the joint action space typically grows exponentially in the number of agents. The problem is exacerbated in environments with sparse rewards in which most transitions will not yield informative rewards.

One common implementation technique to facilitate training with a larger number of agents is parameter sharing (e.g. Gupta et al. [5]) whereby agents share some or all parameters in their policy networks. In the literature, parameter sharing is typically applied indiscriminately across all agents, which we call naive. Naive parameter sharing (FuPS) has been effective primarily due to the similar (if not identical) observation and reward functions between agents found in many multi-agent environments. This similarity allows agents to share representations in intermediate neural network layers. Despite the occasional effectiveness of naive parameter sharing [4, 5, 8, 10], it is not supported by theoretical work and has not received much attention beyond being mentioned as an implementation detail. Indeed, naive parameter sharing can decrease training time, but we show that it can be detrimental to final convergence in many environments, even when paired with implementation details that generally accompany it.

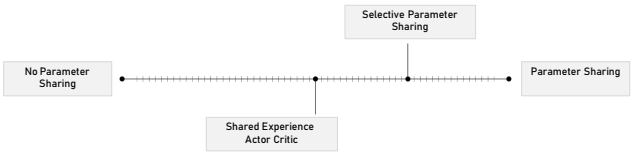


Figure 1: The two baselines (No Parameter Sharing and Parameter Sharing) are in the two ends of the spectrum. Shared Experience Actor Critic [3] is between the two and Selective Parameter Sharing [2] is closer to the Parameter Sharing baseline.

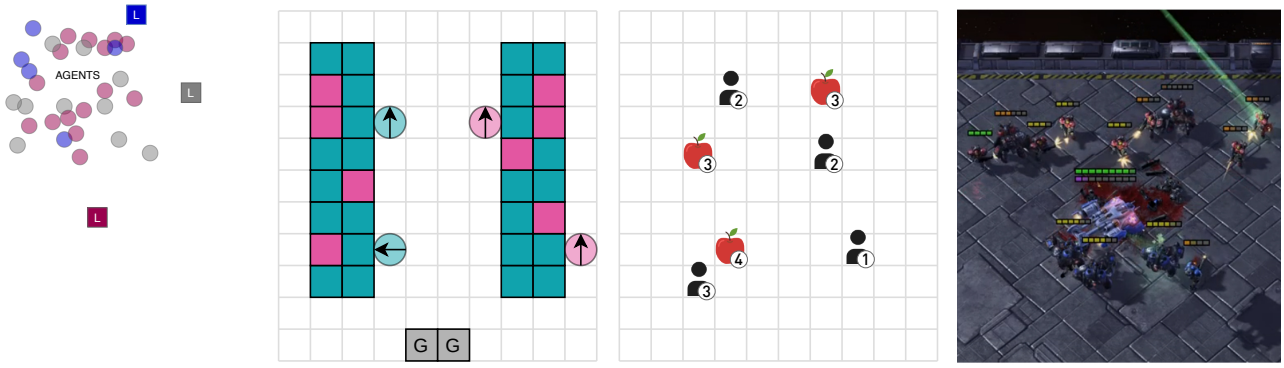
In many multi-agent problems, however, distinct behaviours are expected from the agents. In those problems, parameter sharing poses significant issues in the training procedure since overlapping gradients often destabilise neural network training. Hence, not sharing parameters (NoPS) is also used when such distinct behaviours need to be learned.

In our work, we consider the two baselines above as two ends of a spectrum (Figure 1). Our first work, Shared Experience Actor (SEAC) [3] does not share parameters between agents, but applies the basic idea of off-policy correction via importance weighting and combines the experiences generated by different agents into more informative learning gradients. Therefore, sample efficiency increases significantly by having agents make use of the experience collected by other agents. With SEAC all agents have their own set of parameters and manage to learn distinct behaviours which greatly increases the returns after the policies converge.

Selective Parameter Sharing (SePS) [2] takes a slightly different approach. It makes use of parameter sharing and generates clusters of agents that should share parameters. This is achieved with the use of an encoder-decoder model. The architecture of that model has been designed to encode the agent identities into an embedding space. Depending on the reward and observation functions of the individual agents, the model separates the embeddings into clusters that can then be defined with an unsupervised clustering method.

SEAC and SePS each have their own advantages and should be used in different settings. We have experimental evidence that SEAC achieves state-of-the-art performance in many sparse reward environments. SEAC learns considerably better policies than the other algorithms but does so at the cost of larger batch sizes and the number of parameters (equal to the NoPS baseline). Also, while SEAC learns distinct behaviours, it struggles if the policies should differ significantly. On the other hand, SePS is very computationally

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online.



(a) Particle Environments (MPE)

(b) Multi-Robot Warehouse (RWARE)

(c) Level-based Foraging (LBF)

(d) StarCraft Multi-Agent Challenge (SMAC/MMM2)

Figure 2: Examples of environments used in our work.

cheap and has been shown to scale to hundreds of agents (we have tested up to 200), even if the agents are not homogeneous. As such, SePS combines the benefits of FuPS with the ability to learn distinct policies as in NoPS.

2 EXPERIMENTAL EVALUATION

Throughout our work, we use a collection of multi-agent environments that includes Multi-Robot Warehouse (RWARE) [3], Level-based Foraging (LBF) [1], the Multi-agent Particle Environments (MPE) [6] and StarCraft Multi-agent Challenge (SMAC) [9] (example visualisations in figure 2).

SEAC and SePS have slightly different assumptions and work better in different settings (e.g. SePS scales to a large number of agents), therefore the environment tasks differ, but the overall domains remain the same.

Our SEAC results show similar patterns for the different algorithms across all tested environments. It is not surprising that NoPS requires considerably more environment samples to converge, given that the algorithm is less efficient in using them; NoPS agents only train on their own experience. Also, it is not surprising that FuPS does not achieve as high returns after convergence: sharing a single policy across all agents impedes their ability to coordinate or develop distinct behaviours that lead to higher returns.

We conjecture that SEAC converges to higher final returns due to agents improving at similar rates when sharing experiences, combined with the flexibility to develop differences in policies to improve coordination. We observe that SEAC is able to learn similarly quickly to SNAC because the combined local gradients provide a very strong learning direction. However, while SNAC levels off at some point due to the use of identical policies, which limit the agents' ability to coordinate, SEAC can continue to explore and improve because agents are able to develop different policies to further improve coordination.

SePS shows significant improvements in converged returns across our environments. Specifically, SePS works on environments with a large number of agents when there are agent groups that behave similarly (e.g. in the SMAC task pictured in figure 2d there are types of agents such as seven marines). We showed that SePS can make use of the homogeneity between some agents and correctly separate agents that should have distinct policies. In our results, SePS outperforms all baselines in terms of converged returns and is computationally cheaper than the NoPS baseline.

3 FUTURE WORK

In our future work, we are looking into extending the experimental results and analysis of SePS. In SePS, partitioning the agents using samples collected before agents are allowed to learn a policy does come with a disadvantage. In situations where agents share dynamics and reward functions early in the policies' training but diverge later (e.g. agents are required to do the same task and then a different task in the same episode), learning the encoder-decoder with the initially collected samples may fail to properly partition agents. While in that case SePS will operate similarly to the full parameter sharing baselines, it could be further improved by regularly retraining the encoder-decoder model with newer experience and redistributing agents to clusters if they have diverged.

We also aim to extend the experimental results to include SePS combined with current MARL paradigms such as state-based critics [6], value-decomposition (e.g. [8]) and independent learning. Such experiments will further note the usefulness of SePS and how it can be applied to a variety of different settings.

REFERENCES

[1] Stefano V. Albrecht and Subramanian Ramamoorthy. 2013. A Game-Theoretic Model and Best-Response Learning Method for Ad Hoc Coordination in Multiagent Systems. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS '13)*. Richland, SC, 1155–1156.

[2] Filippos Christianos, Georgios Papoudakis, Arrasy Rahman, and Stefano Albrecht. 2021. Scaling Multi-Agent Reinforcement Learning with Selective Parameter Sharing. In *Proceedings of the 38th International Conference on Machine Learning*.

[3] Filippos Christianos, Lukas Schäfer, and Stefano Albrecht. 2020. Shared Experience Actor-Critic for Multi-Agent Reinforcement Learning. In *Advances in Neural*

Information Processing Systems, Vol. 33. Curran Associates, Inc., 10707–10717.

[4] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[5] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative Multi-agent Control Using Deep Reinforcement Learning. In *Autonomous Agents and Multiagent Systems (Lecture Notes in Computer Science)*. Springer International Publishing, Cham, 66–83. https://doi.org/10.1007/978-3-319-71682-4_5

[6] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Neural Information Processing Systems (NIPS)* (2017).

[7] Georgios Papoudakis, Filippos Christianos, Arrasy Rahman, and Stefano V Albrecht. 2019. Dealing with Non-Stationarity in Multi-Agent Deep Reinforcement Learning. *arXiv preprint arXiv:1906.04737* (2019).

[8] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. PMLR, 4295–4304. http://proceedings.mlr.press/v80/rashid18a.html

[9] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. In *International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Vol. 4. 2186–2188. https://youtu.be/VZ7zmQ_obZ0.

[10] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean Field Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. PMLR, 5571–5580. http://proceedings.mlr.press/v80/yang18d.html