The StarCraft Multi-Agent Challenge

Extended Abstract

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, Shimon Whiteson*

ABSTRACT

In the last few years, deep multi-agent reinforcement learning (RL) has become a highly active area of research. A particularly challenging class of problems in this area is partially observable, cooperative, multi-agent learning, in which teams of agents must learn to coordinate their behaviour while conditioning only on their private observations. This is an attractive research area since such problems are relevant to a large number of real-world systems and are also more amenable to evaluation than general-sum problems.

Standardised environments such as the ALE and MuJoCo have allowed single-agent RL to move beyond toy domains, such as grid worlds. However, there is no comparable benchmark for cooperative multi-agent RL. As a result, most papers in this field use one-off toy problems, making it difficult to measure real progress. In this paper, we propose the StarCraft Multi-Agent Challenge (SMAC) as a benchmark problem to fill this gap.[1] SMAC is based on the popular real-time strategy game StarCraft II and focuses on micromanagement challenges where each unit is controlled by an independent agent that must act based on local observations. We offer a diverse set of challenge maps and recommendations for best practices in benchmarking and evaluations. We also open-source a deep multi-agent RL learning framework including state-of-the-art algorithms.[2] We believe that SMAC can provide a standard benchmark environment for years to come.

Videos of our best agents for several SMAC scenarios are available at: https://youtu.be/VZ7zmQ_obZ0.

KEYWORDS

StarCraft; Reinforcement Learning; Multi-Agent Learning

ACM Reference Format:

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, Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

Deep reinforcement learning (RL) promises a scalable approach to solving arbitrary sequential decision making problems, demanding only that a user must specify a reward function that expresses the desired behaviour. However, many real-world problems that might be tackled by RL are inherently multi-agent in nature. For example, the coordination of self-driving cars, autonomous drones, and other multi-robot systems is becoming increasingly critical. Network traffic routing, distributed sensing, energy distribution, and other logistical problems are also inherently multi-agent. As such, it is essential to develop multi-agent RL (MARL) solutions that can handle decentralisation constraints and deal with the exponentially growing joint action space of many agents.

(a) 3 Stalkers vs 5 Zealots (b) 2 Colossi vs 64 Zerglings

Figure 1: Decentralised micromanagement in StarCraft II. Each unit is an independent learning agent that needs to coordinate with its teammates to defeat the enemy units. Shown are screenshots of two SMAC scenarios.

Partially observable, cooperative, multi-agent learning problems are of particular interest. Cooperative problems avoid difficulties in evaluation inherent with general-sum games (e.g., which opponents are evaluated against). Cooperative problems also map well to a large class of critical problems where a single user that manages a distributed system can specify the overall goal, e.g., minimising traffic or other inefficiencies. Most real-world problems depend on inputs from noisy or limited sensors, so partial observability must also be dealt with effectively. This often includes limitations on communication that result in a need for decentralised execution of learned policies. However, there commonly is access to additional information during training, which may be carried out in controlled conditions or in simulation. This gives rise to the paradigm of *centralised training with decentralised execution*, which has been well-studied in the planning community [6, 9].

A growing number of recent works [3, 8, 11, 14] have begun to address the problems in this space. However, there is a clear lack of standardised benchmarks for research and evaluation. Instead, researchers often propose one-off environments which can be overly simple or tuned to the proposed algorithms. In single-agent RL, standard environments such as the Arcade Learning Environment [1], or MuJoCo for continuous control [10], have enabled great progress. In this paper, we aim to follow this successful model by

*MS and TR contributed equally to this work. MS is with Russian-Armenian University, Armenia. The rest of the authors are with the University of Oxford, United Kingdom. Correspondence to mikayel@samvelyan.com and tabish.rashid@cs.ox.ac.uk

[1]Code is available at https://github.com/oxwhirl/smac

[2]Code is available at https://github.com/oxwhirl/pymarl

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

offering challenging standard benchmarks for deep MARL, and to facilitate more rigorous experimental methodology across the field.

Some testbeds have emerged for other multi-agent regimes, such as Poker [5], Pong [16], Keepaway Soccer[13], or simple gridworld-like environments [7, 8, 18, 19]. Nonetheless, we identify a clear gap in challenging and standardised testbeds for the important set of domains described above.

To fill this gap, we introduce the StarCraft Multi-Agent Challenge (SMAC). SMAC is built on the popular real-time strategy game StarCraft II[3] and makes use of the SC2LE environment [17]. Instead of tackling the full game of StarCraft with centralised control, we focus on decentralised micromanagement challenges (Figure 1). In these challenges, each of our units is controlled by an independent, learning agent that has to act based only on local observations, while the opponent's units are controlled by the hand-coded built-in StarCraft II AI. We offer a diverse set of scenarios that challenge algorithms to handle high-dimensional inputs and partial observability, and to learn coordinated behaviour even when restricted to fully decentralised execution.

The full games of StarCraft: BroodWar and StarCraft II have already been used as RL environments, due to the many interesting challenges inherent to the games [15, 17]. DeepMind's AlphaStar [2] has recently shown an impressive level of play on a StarCraft II matchup using a centralised controller. In contrast, SMAC is not intended as an environment to train agents for use in full StarCraft II gameplay. Instead, by introducing strict decentralisation and local partial observability, we use the StarCraft II game engine to build a new set of rich cooperative multi-agent problems that bring unique challenges, such as the nonstationarity of learning [4], multi-agent credit assignment [3], and the difficulty of representing the value of joint actions [11].

To further facilitate research in this field, we also open-source PyMARL, a learning framework that can serve as a starting point for other researchers and includes implementations of several key MARL algorithms. PyMARL is modular, extensible, built on PyTorch, and serves as a template for dealing with some of the unique challenges of deep MARL in practice. We include results on our full set of SMAC environments using QMIX [11] and several baseline algorithms, and challenge the community to make progress on difficult environments in which good performance has remained out of reach so far. We also offer a set of guidelines for best practices in evaluations using our benchmark, including the reporting of standardised performance metrics, sample efficiency, and computational requirements.

We hope SMAC will serve as a valuable standard benchmark, enabling systematic and robust progress in deep MARL for years to come.

2 STARCRAFT MULTI-AGENT CHALLENGE

Akin to most real-time strategies, StarCraft has two main gameplay components: macromanagement and micromanagement. *Macromanagement* refers to high-level strategic considerations, such as economy and resource management. *Micromanagement* (micro), on the other hand, refers to fine-grained control of individual units.

[3]StarCraft II is the sequel to the game StarCraft and its expansion set Brood War. StarCraft and StarCraft II are trademarks of Blizzard Entertainment$^{\text{TM}}$.

In order to build a rich multi-agent testbed, we focus solely on micromanagement. Micro is a vital aspect of StarCraft gameplay with a high skill ceiling, and is practiced in isolation by amateur and professional players. For SMAC, we leverage the natural multi-agent structure of micromanagement by proposing a modified version of the problem designed specifically for decentralised control. In particular, we require that each unit be controlled by an independent agent that conditions only on local observations restricted to a limited field of view centred on that unit. Groups of these agents must be trained to solve challenging combat scenarios, battling an opposing army under the centralised control of the game's built-in scripted AI.

Proper micro of units during battles will maximise the damage dealt to enemy units while minimising damage received, and requires a range of skills. For example, one important technique is *focus fire*, i.e., ordering units to jointly attack and kill enemy units one after another. When focusing fire, it is important to avoid *overkill*: inflicting more damage to units than is necessary to kill them. Other common micromanagement techniques include: assembling units into formations based on their armour types, making enemy units give chase while maintaining enough distance so that little or no damage is incurred (*kiting*), coordinating the positioning of units to attack from different directions or taking advantage of the terrain to defeat the enemy.

Learning these rich cooperative behaviours under partial observability is a challenging task, which can be used to evaluate the effectiveness of MARL algorithms.

Scenarios. SMAC consists of 22 StarCraft II micro scenarios which aim to evaluate how well independent agents are able to learn coordination to solve complex tasks. These scenarios are carefully designed to necessitate the learning of one or more micromanagement techniques to defeat the enemy. Each scenario is a confrontation between two armies of units. The initial position, number, and type of units in each army varies from scenario to scenario, as does the presence or absence of elevated or impassable terrain. Figure 1 includes screenshots of two SMAC micro scenarios.

The first army is controlled by the learned allied agents. The second army consists of enemy units controlled by the built-in game AI, which uses carefully handcrafted non-learned heuristics. An episode ends when all units of either army have died or when a pre-specified time limit is reached.

3 RESULTS AND CONCLUSIONS

For a more detailed description of the 22 scenarios available in SMAC, recommendations for reporting evaluations and a thorough report and discussion of several state-of-the-art MARL algorithms such as QMIX and COMA please see [12].

In the near future, we aim to extend SMAC with new challenging scenarios that feature a more diverse set of units and require a higher level of coordination amongst agents. Particularly, we plan to make use of the rich skill set of StarCraft II units, and host scenarios that require the agents to utilise the features of the terrain. With harder multi-agent coordination problems, we aim to explore the gaps in existing MARL approaches and motivate further research in this domain, particularly in areas such as multi-agent exploration and coordination.

REFERENCES

[1] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research* 47 (jun 2013), 253–279.

[2] DeepMind. 2019. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. (2019). https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/

[3] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual Multi-Agent Policy Gradients. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence.*

[4] Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Triantafyllos Afouras, Philip H. S. Torr, Pushmeet Kohli, and Shimon Whiteson. 2017. Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning.* 1146–1155.

[5] Johannes Heinrich and David Silver. 2016. Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. *CoRR* abs/1603.01121 (2016). arXiv:1603.01121 http://arxiv.org/abs/1603.01121

[6] Landon Kraemer and Bikramjit Banerjee. 2016. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing* 190 (2016), 82–94.

[7] Joel Z. Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. 2017. Multi-agent Reinforcement Learning in Sequential Social Dilemmas. (Feb. 2017). http://arxiv.org/abs/1702.03037 arXiv: 1702.03037.

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

[9] Frans A. Oliehoek, Matthijs T. J. Spaan, and Nikos Vlassis. 2008. Optimal and Approximate Q-value Functions for Decentralized POMDPs. *JAIR* 32 (2008), 289–353.

[10] Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. 2018. Multi-Goal Reinforcement Learning: Challenging Robotics Environments and Request for Research. (2018). arXiv:arXiv:1802.09464

[11] 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 *Proceedings of the 35th International Conference on Machine Learning.* 4295–4304.

[12] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philiph H. S. Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. *CoRR* abs/1902.04043 (2019).

[13] Peter Stone, Gregory Kuhlmann, Matthew E Taylor, and Yaxin Liu. 2005. Keepaway soccer: From machine learning testbed to benchmark. In *Robot Soccer World Cup*. Springer, 93–105.

[14] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. 2017. Value-Decomposition Networks For Cooperative Multi-Agent Learning. *arXiv:1706.05296 [cs]* (June 2017). http://arxiv.org/abs/1706.05296 arXiv: 1706.05296.

[15] Gabriel Synnaeve, Nantas Nardelli, Alex Auvolat, Soumith Chintala, TimothÃľe Lacroix, Zeming Lin, Florian Richoux, and Nicolas Usunier. 2016. TorchCraft: a Library for Machine Learning Research on Real-Time Strategy Games. *arXiv preprint arXiv:1611.00625* (2016).

[16] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. 2015. Multiagent Cooperation and Competition with Deep Reinforcement Learning. (Nov. 2015). http://arxiv.org/abs/1511.08779 arXiv: 1511.08779.

[17] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich KÃijttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. 2017. StarCraft II: A New Challenge for Reinforcement Learning. *arXiv:1708.04782 [cs]* (Aug. 2017). http://arxiv.org/abs/1708.04782 arXiv: 1708.04782.

[18] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean Field Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:1802.05438* (2018).

[19] Lianmin Zheng, Jiacheng Yang, Han Cai, Weinan Zhang, Jun Wang, and Yong Yu. 2017. MAgent: A Many-Agent Reinforcement Learning Platform for Artificial Collective Intelligence. *arXiv preprint arXiv:1712.00600* (2017).