Distributed Value Decomposition Networks with Networked Agents

Extended Abstract

Guilherme S. Varela
Instituto Superior Técnico, INESC-ID
Lisbon, Portugal
guilherme.varela@tecnico.ulisboa.pt

Alberto Sardinha
PUC-Rio
Rio de Janeiro, Brazil
sardinha@inf.puc-rio.br

Francisco S. Melo
Instituto Superior Técnico, INESC-ID
Lisbon, Portugal
fmelo@inesc-id.pt

ABSTRACT

We investigate the problem of distributed training under partial observability, whereby cooperative multi-agent reinforcement learning agents (MARL) maximize the cumulative joint reward. We propose distributed value decomposition networks (DVDN) that generate a joint Q-function that factorizes into agent-wise Q-functions. Whereas the original value decomposition networks rely on centralized training, our approach is suitable for domains where centralized training is either unavailable or unreliable and agents must resort to learning by interacting with the physical environment in a decentralized manner while communicating with their peers.

KEYWORDS

Artificial Intelligence; Multi-Agent Systems; Reinforcement Learning; Deep Learning; Partial Observability

ACM Reference Format:
Guilherme S. Varela, Alberto Sardinha, and Francisco S. Melo. 2025. Distributed Value Decomposition Networks with Networked Agents: Extended Abstract. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 3 pages.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning focuses on designing utility-maximizing agents that interact with a shared environment, where the system dynamics depend on their joint action. Utility function representation for decision-making is challenging because of the large combined observation and action spaces. Value decomposition networks (VDN) [5] avoid this combinatorial trap by considering a factorized family of Q-functions, yielding a joint Q-function that linearly decomposes between agents. This approach offers a viable solution to the scalability of MARL systems at the cost of constraining the set of representable joint Q-functions . Another limitation is that VDN is relies *centralized training with decentralized execution.*

However, in many real-world scenarios, the premise of centralized training is too restrictive. For instance, in reinforcement learning based distributed[1] load balancing [10] intelligent switches act as agents to distribute various types of requests to a data center's server fleet. The agents assign incoming load to the servers, resolving requests at low latencies under quality-of-service constraints. In this domain there is no simulator, agents learn online, observing queue lengths at links and selecting links to route requests. The reward is the change in queue length within the overall system.

Therefore, we propose *decentralized training* RL agents, which do not rely on a central node for computing decentralized policies and the joint Q-function. Unlike independent learners, DVDN agents produce Q-functions that combine linearly to form a joint Q-function. Thus, DVDN agents *implicitly* encode information about their teammates' actions through local *peer-to-peer* communication. In homogeneous settings, gradient tracking [4] can emulate parameter sharing in the decentralized setting, further enhancing the learning process.

2 PRELIMINARIES

Value decomposition networks [5] are deep RL agents that implicitly learn *additive value decomposition* over individual agents captured by the relation:

$$Q^{\text{VDN}}(o, a; \omega) = \sum_{i=1}^{N} Q_i(o_i, a_i; \omega_i). \tag{1}$$

where ω is the concatenation of the individual network parameters ω_i, o and a represent the concatenation over the agents' observations and actions respectively. The *local temporal difference* (TD) δ_i is given by a deep Q-network [2]:

$$\delta_i = \bar{R} + \gamma \max_{u_i} Q_i(o_i', u_i; \omega_i^-) - Q_i(o_i, a_i; \omega_i).$$

where γ is the discount factor, $\bar{R}$ is the team reward at the next time step, ω_i^- are the parameters of a target network, and the max operator over the target network Q-function at the next step follows the Q-leaning update [8]. In centralized training, the temporal differences are summed. As a result, agents receive temporal difference feedback from the network. The error increment guiding the weight updates at each agent is the *joint temporal difference* (JTD) $\delta = \Sigma_{i=1}^{N} \delta_i$ [7].

3 DISTRIBUTED VALUE DECOMPOSITION NETWORKS

In decentralized training, there is no overseer capable of performing addition over the local temporal differences. Thus, there is no direct way to obtain the joint temporal difference. We propose the use of the *consensus mechanism* [9] whereby each agent updates its TD with a weighted average of its previous TD and TDs of its neighbors:

[1] We use distributed and decentralized interchangeably.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

$$\delta_i^{(k+1)} = \sum_j^N \alpha_{i,j}^{(k)} \delta_j^{(k)} \quad k = 1, 2, \ldots, \tag{2}$$

for a system consisting of N agents. The coefficients $\left[\alpha_{i,j}\right]_{N\times N}$ are the consensus weights, which are non-zero for agent i and its neighbors (Appendix [7]). These coefficients change following a network topology dynamics. By repeatedly applying this procedure, agents asymptotically agree (reach consensus) on an average JTD ($^{\delta}/_{N}$) under mild assumptions. Since it is impossible to guarantee consensus with a finite number of steps, so we limit the number of consensus iterations during training. From these local approximations, agents recover *network estimated JTD*: $\hat{\delta}_{-i} = N(\sum_j^N \alpha_{i,j}^{(k)} \delta_j^{(k)}) - \delta_i^{(k)}$. The index $-i$ represents an estimator at agent i of the network average excluding its own temporal difference. DVDN agents minimize the mean square error criteria:

$$\ell(\omega_i; \tau_i, \hat{\delta}_{-i}) = \frac{1}{T} \sum_{\tau_i} \left(\delta_i + \hat{\delta}_{-i}\right)^2 \tag{3}$$

where τ_i is the batch of trajectories for agent i and T denotes the episode length, the network-estimated JTD term allows agents to adjust their weights based on the error increments at their peers.

In settings with homogeneous agents, where agents have the same observation space and action spaces, *gradient tracking* (GT) [4] can align their parameters and gradient updates. GT allows agents to aggregate their knowledge, emulating parameter sharing [1] in the decentralized setting. Distributed value decomposition agents with gradient tracking DVDN (GT) perform the update in (2) while producing localized estimators for the average network parameter and the average gradient. For an arbitrary batch update k:

$$\hat{\delta}_{-i}^{(k)} = N\left(\sum_{j=1}^N \alpha_{i,j}^{(k)} \delta_j^{(k)}\right) - \delta_i^{(k)} \tag{4a}$$

$$g_i^{(k)} = \nabla \ell(\omega_i^{(k-1)}; \tau_i^{(k)}, \hat{\delta}_{-i}^{(k)}) \tag{4b}$$

$$z_i^{(k)} = \sum_{j=1}^N \alpha_{i,j}^{(k)} z_i^{(k-1)} + g_i^{(k)} - g_i^{(k-1)} \tag{4c}$$

$$\omega_i^{(k)} = \sum_{j=1}^N \alpha_{i,j}^{(k)} \omega_i^{(k-1)} - \eta z_i^{(k)} \tag{4d}$$

where the variable $g_i^{(k)}$ tracks the local gradient (3), $z_i^{(k)}$ tracks the team gradient, and the parameter $\omega_i^{(k)}$ is updated using the weighted average parameter from neighbors and the team gradient. We open source the codebase with the implementation of both algorithms[2].

4 RESULTS

We showcase our approach in two standard MARL benchmark environments with partial observability.[3] At each training step, agents execute consensus iteration ((2), (4a), (4c), (4d)) per consensus equation, over a connected graph. Table 1 provides results for DVDN in the MARBLER [6] environment, where heterogeneous robotic agents are faced with four different navigation tasks, and DVDN (GT) in three level-based foraging tasks, where homogeneous agents must coordinate to collect fruits in a sparse reward environment. We evaluate DVDN's performance against independent deep-Q learners (IQL) [3], VDN [5], and VDN with parameter sharing [3] (VDN (PS)). IQL serves as a lower performance threshold while VDN acts as a performance ceiling.

Table 1: Maximum average episodic returns over ten independent seeds, their respective 95% bootstrapped confidence interval for all algorithms and tasks. Highlighted results are those with the higher maximum average episodic returns. The asterisk denotes results that match the performance of the best result for the task. The double asterisk denotes results that are second in the rank.

Env.	Scenarios	Heterogenous IQL	Heterogenous DVDN	Heterogenous VDN
MARBLER	Arctic	−43.51 (−1.64, 1.65)	−37.56** (−1.01, 0.84)	**−30.93** (−0.70, 0.76)
	Material	12.81 (−0.49, 0.51)	18.07** (−1.14, 1.30)	**21.82** (−0.36, 0.36)
	PCP	130.72** (−0.81, 0.76)	**133.02** (−0.67, 0.78)	125.10 (−2.57, 3.09)
	Warehouse	21.99 (−0.42, 0.38)	**28.74** (−0.45, 0.45)	23.65** (−0.90, 0.93)
		IQL	**DVDN (GT)**	**VDN (PS)**
LBF	Easy	0.81 (−0.02, 0.02)	0.89** (−0.02, 0.02)	**0.94** (−0.01, 0.01)
	Medium	0.61 (−0.02, 0.02)	0.72** (−0.01, 0.02)	**0.79** (−0.02, 0.02)
	Hard	0.43 (−0.01, 0.02)	0.52** (−0.01, 0.02)	**0.56** (−0.02, 0.02)

In the homogeneous agents setting, results show that DVDN (GT), which has information loss due to the communication network dropping out links, approximates the performance of VDN (PS). In the heterogeneous agents setting, results show that DVDN not only approximates but can even exceed the performance of VDN. We hypothesize that DVDN weight updates generate policies that are more effective in the exploration of some tasks.

ACKNOWLEDGMENTS

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) through the projects with references UIDB/50021/2020 (DOI: 10.54499/UIDB/50021/2020), PTDC/CCI-COM/5060/2021, and the Center for Responsible AI (ref. n. C628696807-00454142). Guilherme S. Varela is supported by FCT scholarship 2021.05435.BD.

[2] https://github.com/GAIPS/DVDN

[3] For extra experimental configurations and analysis of the results, refer to Sections 4 and 5 [7].

REFERENCES

[1] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative Multi-agent Control Using Deep Reinforcement Learning. In *Autonomous Agents and Multiagent Systems*, Gita Sukthankar and Juan A. Rodriguez-Aguilar (Eds.). Springer International Publishing, Cham, 66–83.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2015. Playing Atari with Deep Reinforcement Learning. *Nature* 518, 7540 (2015), 529–533.

[3] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano Albrecht. 2021. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, J. Vanschoren and S. Yeung (Eds.), Vol. 1.

[4] Guannan Qu and Na Li. 2018. Harnessing Smoothness to Accelerate Distributed Optimization. *IEEE Transactions on Control of Network Systems* 5, 3 (2018), 1245–1260.

[5] 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. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems* (Stockholm, Sweden) *(AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2085–2087.

[6] Reza Joseph Torbati, Shubham Lohiya, Shivika Singh, Meher Shashwat Nigam, and Harish Ravichandar. 2023. MARBLER: An Open Platform for Standardized Evaluation of Multi-Robot Reinforcement Learning Algorithms. In *International Symposium on Multi-Robot and Multi-Agent Systems, MRS 2023, Boston, MA, USA, December 4-5, 2023*. IEEE, 57–63.

[7] Guilherme S. Varela, Alberto Sardinha, and Francisco S. Melo. 2025. Distributed Value Decomposition Networks. arXiv:2502.07635 [cs.LG] https://arxiv.org/abs/2502.07635

[8] Christopher Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning* 8, 3 (1992), 279–292. https://doi.org/10.1007/BF00992698

[9] Lin Xiao, Stephen Boyd, and Seung-Jean Kim. 2007. Distributed average consensus with least-mean-square deviation. *J. Parallel and Distrib. Comput.* 67, 1 (2007), 33–46.

[10] Changgang Zheng, Benjamin Rienecker, and Noa Zilberman. 2023. QCMP: Load Balancing via In-Network Reinforcement Learning. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Future of Internet Routing & Addressing* (New York, NY, USA) *(FIRA '23)*. Association for Computing Machinery, 35–40.