18662 PROJECT REPORT - MULTI-AGENT REINFORCEMENT LEARNING WITH CONTINUOUS ACTIONS

Ipek Ilayda Onur
Eletrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
ionur@andrew.cmu.edu

Runqi (Tony) Huang
Eletrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
runqih@andrew.cmu.edu

March 17, 2023

1 Introduction and Motivation

Reinforcment leanring (RL) [1] is a powerful framework by which an agent can learn to solve complex tasks. In recent years, RL gained prominence because of its superior ability to play games such as Atari [2] and its reputation was cemented when AlphaGo [3] beat the world champion in Go by 4-to-1.

In the case of AlphaGo, the algorithm was controlling only one agent; however, many important problem settings require multiple agents to cooperate with each other and make simultaneously decisions. For example, in the real world, unmanned aerial vehicles need to navigate as a fleet without crashing into each other, while in the virtual world, StarCraft II requires players' cooperation to achieve the final prize. In practice, there can be constrains in physical communication channels, and as a result, the agents can only partially observe the environment and they need to make the best decision, given the limited available information.

In this project, we are motivated by the topic of multi-agent reinforcement learning, and by designing an appropriate network architectures for the agents, we seek to improve upon the state-of-the-art results for multi-agent reinforcement learning in the continuous state space.

2 Literature Review

In this section, we have outlined a few key areas that are important to understanding our project and are related to the state-of-art techniques. We included a brief description of what we learned and reference resources.

2.1 Continuous Control with Deep Reinforcement Learning [4]

Deep Q Networks (DQN) are powerful but they only work with "discrete and low dimensional action spaces." Therefore, there is a need to adapt DQNs to high dimensional continuous action spaces. This paper presents "a model-free, off-policy actor-critic algorithm" called Deep DPG (DDPG) that can handle high dimensional continuous action spaces and is an extension of the deterministic policy gradient (DPG) algorithm to the continuous action domain.

The parameterized policy $\mu\left(s \mid \theta^{\mu}\right) : \mathcal{S} \mapsto \mathcal{A}$ is updated using DPG by:

$$\nabla_{\theta^{\mu}} J \approx \mathbb{E}_{s \sim \rho^{\beta}}\left[\nabla_{\theta^{\mu}} Q\left(s, \mu(s \mid \theta^{\mu}) \mid \theta^{Q}\right)\right] = \mathbb{E}_{s \sim \rho^{\beta}}\left[\nabla_{\theta^{\mu}} \mu(s \mid \theta^{\mu}) \nabla_{a} Q(s, a \mid \theta^{Q})\big|_{a=\mu(s|\theta^{\mu})}\right], \tag{1}$$

where the expectation is computed over a different policy β. The parameterized action-value function $Q\left(s, a \mid \theta^{Q}\right)$ is updated by:

$$\mathcal{L}(\theta^Q) = \mathbb{E}_{s,a,r,s'}\left[\left(Q(s,a \mid \theta^Q) - y\right)^2\right], \quad \text{where} \quad y = r + \gamma \max_{a'} \bar{Q}\left(s', a' \mid \theta^{\bar{Q}}\right), \tag{2}$$

where $\bar{Q}\left(s, a \mid \theta^{\bar{Q}}\right)$ is the target network that stabilizes training.

2.2 Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments [5]

This paper extends the ideas of DDPG to multi-agent setting and presents a network called Multi-Agent Deep DPG (MADDPG). In most cooperative multi-agent settings, agents can partially observe their environment and they are constrained to act independently. Therefore, these scenarios require decentralized policies. However, a naive adaptation of DDPG [6] inevitably leads to poor local extrema, since the evolving policies of other agents cause the environment to be non-stationary, which consequently prevents the emergence of cooperative behavior. In order to avoid this, there is a need for centralized training with decentralized execution and MADDPG algorithm achieves this in continuous action domain by assigning a centralized critic for each agent and allowing the agents to have access to global state information during training time to facilitate learning cooperative behavior.

MADDPG augments each agent's critic $Q_i\left(o_1, \ldots, o_N, a_1, \ldots, a_N \mid \theta^{Q_i}\right) \coloneqq Q_i\left(\mathbf{x}, \mathbf{a} \mid \theta^{Q_i}\right)$ by inputting the actions of all agents and the global state information and outputting the Q-value for each agent i. Denoting $\boldsymbol{\mu} = \{\mu_1\left(o_1 \mid \theta^{\mu_1}\right), \ldots, \mu_N\left(o_N \mid \theta^{\mu_N}\right)\}$ as the set of policies, the actors and critics are updated according to Eq. 3 and Eq. 4, respectively:

$$\nabla_{\theta^{\mu_i}} J = \mathbb{E}_{\mathbf{x},\mathbf{a}\sim\mathcal{D}}\left[\nabla_{\theta^{\mu_i}} \mu\left(o_i \mid \theta^{\mu_i}\right) \nabla_{a_i} Q\left(\mathbf{x}, \mathbf{a} \mid \theta^{Q_i}\right)\Big|_{a_i=\mu_i(o_i)}\right], \tag{3}$$

$$\mathcal{L}\left(\theta^{Q_i}\right) = \mathbb{E}_{\mathbf{x},\mathbf{a},r,\mathbf{x}'}\left[\left(Q\left(\mathbf{x}, \mathbf{a} \mid \theta^{Q_i}\right) - y_i\right)^2\right], \quad \text{where} \quad y_i = r_i + \gamma Q\left(\mathbf{x}', \mathbf{a}' \mid \theta^{Q_i'}\right)\Big|_{a_j'=\mu_j'(o_j)} \tag{4}$$

2.3 QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning [7]

QMIX is "a novel value-based method that can train decentralized policies in a centralized" way. Environments with multiple agents typically generate global rewards based on agents' joint actions; as a result, the contribution from individual agent is difficult to evaluate. QMIX paper presents centralized training of decentralized policies through a value decomposition network (VDN) which decomposes the centralized Q_{tot} value as a linear combination of per-agent Q_i values. However, QMIX imposes a monotonicity constraint between the join-action value and per-agent action value. One weakness of the model is that the monotonicity constraints are not theoretically guaranteed in cooperative setting, which also limits the classes of representable functions.

3 Project Formulation

Based on our literature survey, the scope of this project will focus on multi-agent reinforcement learning, where the action space for each agent is continuous. In this section, we will outline our assessment of the state-of-the-art results in this arena, evaluation of the "gap," and proposal for a potential area of improvement.

3.1 State-of-the-art Assessment

Both MADDPG and QMIX are considered to be state of the art models that leverage the "centralized training with decentralized execution" approach to handle multi-agent settings. In other words, both approaches incorporate additional state information in training as a way to extract an optimal decentralized policy. In the original MADDPG formulation, the algorithm is applied to continuous action space, while QMIX's setting is applied to discrete action space.

3.2 What is the "Gap"?

Although both MADDPG and QMIX, some of their assumptions and constraints do not guarantee convergence of the policy and require further justification. Through our literature review, we have identified the following two assumptions that can be challenged. First, a fundamental challenge in multi-agent RL is the credit assignment problem. In an environment where the ultimate outcome and reward depend on the joint actions of the agents, the contribution of individual agent may be difficult to evaluate. The problem is left unanswered with MADDPG, and it remains unclear

whether the centralized critics under MADDPG encourages the optimal team behavior. Second, the QMIX architecture enforces a monotonicity constraint such that the joint action value is non-decreasing with respect to the individual action value. However, there are circumstances where individual sacrifices could result in short-term cost but long-term benefit. In such case, the monotonicity constraint would not theoretically guarantee the optimal cooperative policy and convergence in training.

3.3 Our Approach

To address the credit-assignment and non-stationarity problem, we propose Deterministic QMIX, a multi-agent actor-critic algorithm that operates in the continuous action domain. To overcome non-stationarity, we will adopt the framework of centralized training with decentralized execution, assigning for each agent a centralized critic that is used only during training time. On the other hand, we approach the credit-assignment problem by imposing a mixing network that non-linearly combines each agent's Q_i value into Q_{tot}, with no additional constraints such as monotonicity. We will also design a new training objective based on the mixing network. Overall, our main proposals can be summarized as follow:

1. Introduce a new neural network architecture for multi-agent actor-critics that can generate a joint-value based on the per-agent value, without imposing any assumed constraints.
2. In order to ensure consistent and cooperative behavior, we will introduce a new training scheme, whereby agents are trained to simultaneously maximize the joint-value.

4 Project Hypothesis and Expected Outcome

Based on the previous sections, we believe that the current challenges center on the credit-assignment problem and the non-stationarity problem. To address the problem of non-stationarity, the framework of centralized training with decentralized execution has demonstrated promising results. By allowing each agent's critic to incorporate additional information during training time, the centralized critic helps reduce the variance of the gradients, effectively removing a source of uncertainty [5]. Furthermore, to address the credit-assignment problem, a mixing network that non-linearly combines each agent's Q_i value into Q_{tot} can be used. Based on the above assumptions and proposal, we hypothesize that a training scheme that leverages the power of both centralized critic and mixing network is able to improve the performance and collaborative behavior of the agents.

5 Mid-Term Progress Report

At the current phase of the project, we have 1) identified the experiment environment, 2) implemented the baseline MADDPG algorithm, 3) designed the algorithm for our proposed deterministic QMIX (yet to be executed).

First, we will adopt the experiment environment used in [5], and evaluate our algorithm on a set of competitive and cooperative scenarios in the continuous action domain from the Multi-agent Particle Environment (MPE) [8] [1]. This environment offers a set of classical 2D games, and we will particularly focus on predator-and-prey, which is illustrated in Figure 1. In this case, the prey (green) is faster and receive a negative reward for being hit by the adversaries (the red predators). The predators are the slower-moving adversarial agents, and they receive a positive reward for hitting the prey. The obstacles block the way, applicable to both the predators and the prey.

Second, we have implemented our baseline MADDPG algorithm from [5]. The overall network architecture is shown in Figure 2. After training the agents, we are observing that the predators are able to collaborate with each to catch the prey. We report the scores of the predators and prey in Table 5, but note that the results will be more meaningful when the rewards from Deterministic QMIX training become available.

Algorithm	Predator Score	Prey Score
MADDPG	6.4	-10.8
Deterministic QMIX	TBD	TBD

Third, we have written down our proposed extension for Deterministic QMIX. The network architecture is shown in Figure 3. The network architecture is inspired by that used in [7], but we do not impose any additional monotonicity constraint. The full algorithm for our proposed Deterministic QMIX is shown in 1. The next phase of our project will focus on implementing this algorithm.

[1] The original code base for MPE is no longer maintained, a maintained version of the environment is available in PettingZoo [9].

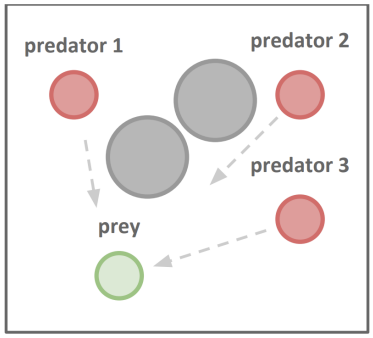

Figure 1: Illustrations of the experimental environment on predator-prey. The red circles represent predators, the green circle the prey, and the grey circles the obstacles. The goal is for the predator to maximize its rewards in a given game duration.

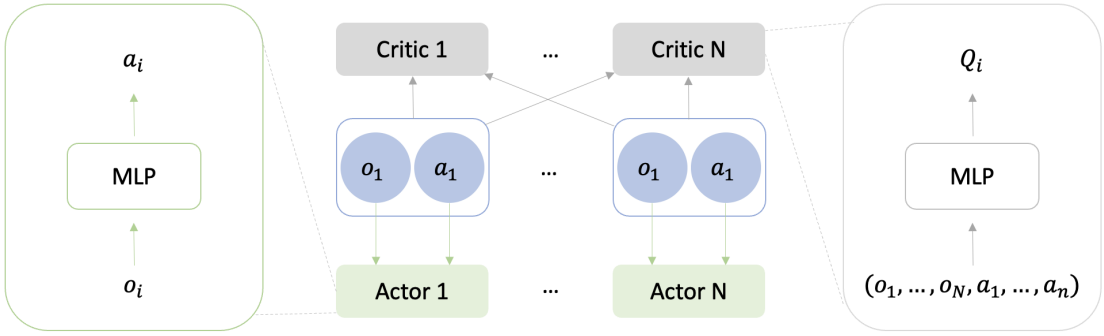

Figure 2: Illustrations of MADDPG algorithm's network architecture. Each actor generates an action a_i based on the partial observation o_i. The critic receives the actions and observations of all the agents, and outputs a Q_i value.

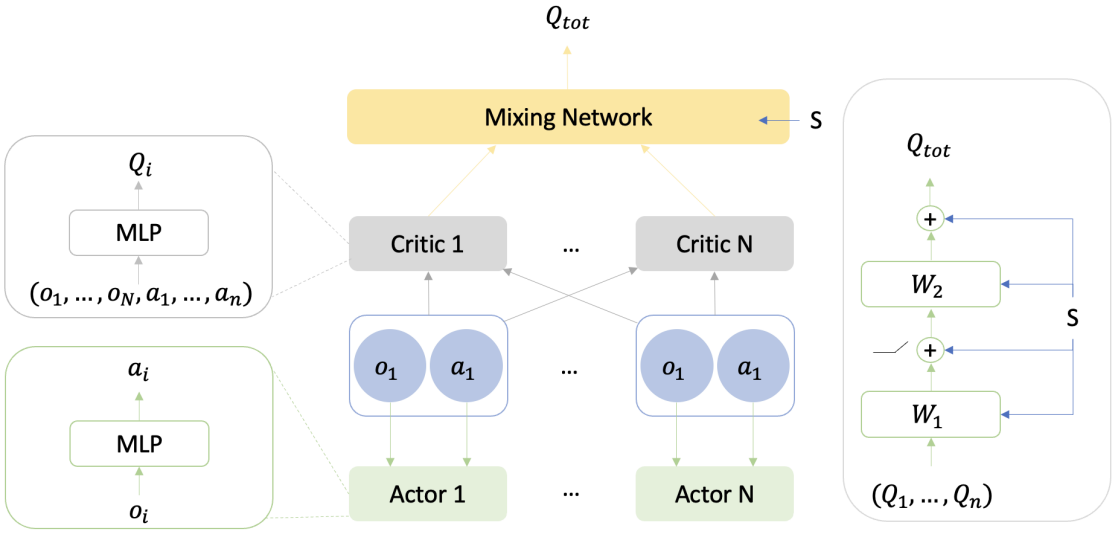

Figure 3: Illustrations of QMIX algorithm. The state information will be input into the mixing network to produce a Q_{tot} value.

Algorithm 1: Deterministic QMIX for N agents

for episode $= 1 \dots K$ **do**

Initialize a random process $\mathcal{N}$ for action exploration, and receive initial state $\mathbf{x}$

for time $t = 1$ to max-episode-length **do**

For each agent i, select action $a_i = \mu_i\left(o_i \mid \theta^{\mu_i}\right) + \mathcal{N}_t$ w.r.t. the current policy and exploration noise $\mathcal{N}_t$

Execute action $\mathbf{a} = (a_1, \dots, a_N)$ and observe reward r and new state $\mathbf{x}'$

Store $(\mathbf{x}, \mathbf{a}, r, \mathbf{x}')$ in replay buffer D

$\mathbf{x} \leftarrow \mathbf{x}'$

Sample a minibatch of S transitions $\left(\mathbf{x}^j, \mathbf{a}^j, r^j, \mathbf{x}'^j\right)$ from D

Set $y_{tot}^j = \sum_{i=1}^{N} r_i^j + \gamma \mathbb{Q}'\left(Q'_1(\mathbf{x}'^j, \mathbf{a}' \mid \theta^{Q'_1}), \dots, Q'_N(\mathbf{x}'^j, \mathbf{a}' \mid \theta^{Q'_N}) \mid \theta^{\mathbb{Q}'}\right)\Big|_{a'_k = \mu'_k\left(o'^j_k\right)}$

Update all critics and the hypernetwork by minimizing the loss:

$$\mathcal{L} = \frac{1}{S} \sum_{j=1}^{S} \left(\mathbb{Q}\left(Q_1^j, \dots, Q_N^j \mid \theta^{\mathbb{Q}}\right) - y_{tot}^j\right)^2$$

Update all actors using the sampled policy gradient:

$$\nabla_{\theta^{\mu_i}} J \approx \frac{1}{S} \sum_{j=1}^{S} \nabla_{Q_i} \mathbb{Q}(Q_1^j, \dots, Q_N^j \mid \theta^{\mathbb{Q}}) \nabla_{a_i} Q_i\left(\mathbf{x}^j, \mathbf{a}^j \mid \theta^{Q_i}\right) \nabla_{\theta^{\mu_i}} \mu_i\left(o_i^j \mid \theta^{\mu_i}\right)\Big|_{a_k = \mu_k\left(o_k^j\right)}$$

end

update target actor and critic networks parameters for each agent i:

$$\theta^{Q'_i} \leftarrow \tau \theta^{Q_i} + (1 - \tau)\theta^{Q'_i}$$

$$\theta^{\mu'_i} \leftarrow \tau \theta^{\mu_i} + (1 - \tau)\theta^{\mu'_i}$$

update target hypernetwork parameters: $\theta^{\mathbb{Q}'} \leftarrow \tau \theta^{\mathbb{Q}} + (1 - \tau)\theta^{\mathbb{Q}'}$

end

References

[1] A. G. Sutton, R. S.; Barto. Reinforcement learning: An introduction, volume 1. 1998.

[2] K.; Silver D.; Rusu A. A.; Veness J.; Bellemare M. G.; Graves A.; Riedmiller M.; Fidjeland A. K.; Ostrovski G.; et al. Mnih, V.; Kavukcuoglu. Human-level control through deep reinforcement learning. *Nature, 518(7540):529–533*, 2015.

[3] A.; Maddison C. J.; Guez A.; Sifre L.; van den Driessche G.; Antonoglou I.; Panneershelvam V.; Lanctot M.; Dieleman S.; Grewe D.; Nham J.; Kalchbrenner N.; Sutskever I.; Lillicrap T.; Leach M.; Kavukcuoglu K.; Graepel T.; Silver, D.; Huang and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature, 529(7587):484 – 489*, 2016.

[4] J. J.; Pritzel A.; Heess N.; Erez T.; Tassa Y.; Silver D.; Lillicrap, T. P.; Hunt and D. Wierstra. Continuous control with deep reinforcement learning. *arxiv preprint arXiv:1509.02971*, 2015.

[5] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Neural Information Processing Systems (NIPS)*, 2017.

[6] D. A.; Singh S. P.; Sutton, R. S.; McAllester and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *Neural Information Processing Systems 12*, page 1057–1063, 1999.

[7] M.; Schroeder de Witt C.; Farquhar G.; Foerster J.; Rashid, T.; Samvelyan and Whiteson S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. *International Conference on Machine Learning*, page 4292–4301, 2018.

[8] Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. *arXiv preprint arXiv:1703.04908*, 2017.

[9] J Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis S Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043, 2021.