# Off-Policy Multi-Agent Policy Optimization with Multi-Step Counterfactual Advantage Estimation

Anonymous Author(s)
Submission Id: ???

## ABSTRACT

In this paper, we propose a new advantage estimation method for multi-agent policy gradient in off-policy multi-agent reinforcement learning, which effectively addresses multi-agent credit assignment and guarantees policy invariance. Our method is based on reward shaping with a counterfactual potential function and off-policy $n$-step advantage estimation with our shaped reward. Empirical results on the StarCraft II and multi-agent MuJoCo environments demonstrate that our proposed algorithm significantly outperforms existing state-of-the art algorithms for these cooperative multi-agent tasks.

## KEYWORDS

Multi-Agent Reinforcement Learning, Credit Assignment, Reward Shaping, Off-Policy Learning

## 1 INTRODUCTION

Deep multi-agent reinforcement learning (MARL) is actively studied in recent years to solve many real-world problems that can be modeled as multi-agent systems such as autonomous driving, coordinated drone fleet flight, and traffic control [24, 31, 33]. In cooperative multi-agent systems, each agent learns to maximize the sum of globally-shared rewards based on samples obtained from its interaction with the environment. The simplest approach to MARL is independent learning in which each agent learns its policy by treating other agents as a part of the environment [28]. In this case, however, it is difficult to learn coordinated behavior and the environment viewed by each agent becomes non-stationary. This non-stationarity hinders stable learning. To circumvent this limitation, many MARL algorithms have been proposed recently under the framework of centralized training and decentralized execution (CTDE) under the assumption that more resources are available during the training period [14]. Under this framework, with the availability of the global state or the collection of all observations and actions from individual agents during the training period, the non-stationarity can be mitigated and coordinated behavior can be learned. There have been vigorous efforts on devising MARL algorithms that use such global information efficiently for learning coordinated behavior under CTDE based on value-based methods, e.g., VDN [26], QMIX [21], MASER [9] and based on policy gradient methods, e.g., COMA [5], FACMAC [19].

Recently, in the line of policy optimization for MARL, multi-agent PPO (MAPPO, Yu et al. [32]) is proposed by extending the PPO algorithm [23] to MARL and it shows outstanding performance compared with existing on-policy methods and other off-policy value-based methods. MAPPO uses value estimation with the global state to update the policies of all agents by using the advantage estimated by Generalized Advantage Estimation (GAE, Schulman et al. [22]). This provides MAPPO with a simple architecture but also with a limitation. That is, with MAPPO it is not easy to know how much each agent contributes to the global reward. In MARL, it is advantageous to know each agent's contribution to the globally shared reward because we can apply further measures when we know the performance of each agent, such as enhancing more exploration for agents in some local optima with low contributions.

Multi-agent credit assignment aims to handle this problem by distinguish each agent's contribution for further enhancement, and has become one of the important research directions in MARL. Recently, several approaches have been proposed to address the credit assignment problem with multi-agent policy gradient methods. COMA [5] designs a counterfactual baseline based on difference reward, which enables explicit credit assignment with marginalizing individual expected value. DAE [13] proposes a potential-based difference reward to apply difference reward to the GAE structure. However, these methods have the disadvantage of sample inefficiency since they are on-policy methods, which discard old sample batches generated by previous policies. In order to increase the sample efficiency by using old samples, we need to integrate multi-agent credit assignment with off-policy learning by going beyond multi-agent credit assignment with on-policy policy gradient.

In the case of single-agent RL, several studies showed that some off-policy methods for policy gradient are effective. For example, Retrace($\lambda$) [16] and V-trace [4] showed that off-policy value estimation based on importance sampling can improve the sample efficiency of policy gradient methods. DISC [7] and GePPO [20] improved the performance of single-agent PPO with theoretical analysis and proper application of off-policy value estimation. Despite these efforts, however, there is not much work on off-policy schemes to multi-agent policy gradient and policy optimization.

In this paper, we present an off-policy optimization approach to address both the sample efficiency and the credit assignment problem that arise in multi-agent policy gradient and policy optimization. First, to handle the credit assignment explicitly, we consider a potential-based reward shaping to guarantee policy invariance. Policy invariance is a desired property because the optimal policy does not change by reward shaping with this property. For such a reward shaping in MARL, we propose a counterfactual potential-based reward shaping that leverages marginalized action value estimation to explicitly handle credit assignment. On top of this credit assignment, we further introduce off-policy value estimation for each agent's update along with an adaptive KL-divergence loss to stabilize the reuse of old samples for improved sample efficiency. We demonstrate the effectiveness of our proposed method with experimental results in various environments.

*Proc. of the Adaptive and Learning Agents Workshop (ALA 2023), Cruz, Hayes, Wang, Yates (eds.), May 29-30, 2023, Online, https://ala2023.github.io/.* 2023, This work was submitted to ICML 2023.

## 2 RELATED WORKS

***Multi-agent policy gradient.*** Multi-agent policy gradient is a natural approach under the CTDE framework because it can exploit global state to estimate the centralized critic. MADDPG [14] uses individual critic for each agent with global information. Gupta et al. [6] propose parameter sharing for actor-critic to reduce the network size and improve learning efficiency. COMA [5] learns a single centralized critic, and estimates individual advantage with counterfactual baseline, which enables explicit credit assignment. DOP [30] and FACMAC [19] propose off-policy policy gradient methods based on multi-agent value decomposition. IPPO [2] and MAPPO [32] show that PPO performs strongly in the SMAC environment. HAPPO [11] analyzes MAPPO theoretically to guarantee monotone improvement, and proposes a sequential update structure with individual actor networks. Kuba et al. [12] analyzes the variance of multi-agent policy gradient, and proposes a baseline that minimizes the variance.

***Credit assignment and reward shaping.*** Credit assignment methods can be classified into two main approaches: implicit and explicit approaches. Sunehag et al. [26], Rashid et al. [21], Son et al. [25] enable implicit credit assignment with value decomposition networks. With using global state information, the mixing network obtains the joint value network from individual values. On the other hand, several works use reward shaping to assign credit explicitly. LIIR [3] learns an individual parameterized intrinsic reward for each agent with individual state-action pairs. Following the counterfactual baseline of COMA, DAE [13] proposes the difference advantage estimation with a potential-based reward shaping. There exists some other works in addition to the two main approaches. Iqbal and Sha [8] and Kim et al. [10] use an attention network to enhance the collaboration among agents.

***Off-policy value estimation.*** $N$-step value estimation can provide more accurate value estimation for policy gradient methods. TD($\lambda$) [27] and GAE [22] have been used widely in on-policy methods. In single-agent RL, $n$-step off-policy value estimation has been developed for the sample efficiency. Retrace($\lambda$) [16] and V-trace [4] propose low-variance value estimation by using truncated importance sampling trace. DISC [7] suggests using GAE-V (i.e., GAE with V-trace) for off-policy advantage estimation. GePPO [20] analyzes the effectiveness of reusing off-policy samples with PPO and GAE-V.

## 3 BACKGROUND

### 3.1 Decentralized POMDP

We consider a cooperative multi-agent system modeled as a decentralized partially observable MDP (Dec-POMDP) [18], where multiple agents cooperate as a team and choose sequential actions with partial observations. A Dec-POMDP can be defined by a tuple $< \mathcal{N}, \mathcal{S}, \mathcal{U}, \mathcal{P}, r, \mathcal{Z}, O, \gamma >$, where $\mathcal{N}$ is the set of agents with cardinality $N$, $\mathcal{S}$ is the set of states, $\mathcal{U}$ is the set of actions, $\mathcal{P}$ is the transition probability, $r$ is the reward function, $\mathcal{Z}$ is the observation space, $O$ is the observation function, and $\gamma$ is the discount factor. At each timestep $t$, each agent $i \in \mathcal{N}$ makes an observation $o_t^i \in \mathcal{Z}$ regarding the state $s_t \in \mathcal{S}$ according to the observation function $O(s_t, i)$. Based on the observation $o_t^i$, each agent $i$ chooses its action $u_t^i \in \mathcal{U}$ according to its own policy $\pi^i\left(a_t^i | o_t^i\right)$. Then, based on the collection of all actions from all agents $\mathbf{a}_t = (a_t^1, \cdots, a_t^N)$ and the current state $s_t$, the environment returns the global reward $r_t$ shared by all agents according to the reward function $r(s_t, \mathbf{a}_t)$ and makes a transition to a next state $s_{t+1}$ according to the transition probability $\mathcal{P}$.

We adopt the widely-used CTDE framework. Thus, global information is used to train decentralized policies, whereas each agent chooses its action with local observation during the execution phase. We consider an actor-critic approach, where the action-value function and state-value function of the joint policy $\boldsymbol{\pi} = (\pi^1, \cdots, \pi^N)$ are defined as $Q^{\boldsymbol{\pi}}(s, \boldsymbol{a}) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \boldsymbol{a_t}) | s_0 = s, \boldsymbol{a_0} = \boldsymbol{a}\right]$ and $V^{\boldsymbol{\pi}}(s, \boldsymbol{a}) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s\right]$, respectively. The goal is to find a joint policy that maximizes the discounted sum of global rewards.

### 3.2 Multi-Agent PPO

MAPPO (multi-agent proximal policy gradient) is a PPO-based MARL method and is shown to outperform many existing MARL methods. MAPPO has the following objective function:

$$L(\theta) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{i=1}^{N} \min(\rho^i \hat{A}, \text{clip}(\rho^i, 1-\epsilon, 1+\epsilon)\hat{A}),\right]$$

where $\rho^i = \frac{\pi^i(a^i|s)}{\pi_k^i(a^i|s)}$ is the individual importance sampling ratio for current policy $\pi_k$, and $\hat{A}$ is an advantage common to all agents, estimated by generalized advantage estimation (GAE, Schulman et al. [22]). GAE estimates the advantage $\hat{A}$ as

$$A_t^{GAE(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^{GAE},$$

where $\delta_{t+l}^{GAE} = r_{t+l} + \gamma V(s_{t+l+1}) - V(s_{t+l})$ is the TD-residual, $V(s)$ is the parameterized state-value function, and $\lambda$ is the bias-variance trade-off parameter.

### 3.3 Reward Shaping

In single-agent RL, policy invariance [17] is one of the major principles to guarantee that the policy can still reach an optimal one with a shaped reward. One key approach to policy invariance is reward shaping. In particular, Ng et al. [17] showed that potential-based reward shaping does not change policy gradient and optimal policy. That is, the policy maximizing $\sum_{t=0}^{\infty} \gamma^t \tilde{r}_t$ also maximizes $\sum_{t=0}^{\infty} \gamma^t r_t$ among all policies, where $r_t$ is the original reward and $\tilde{r}_t$ is the potential-based shaped reward. Potential-based reward shaping transforms the original reward $r$ to a shaped reward $\tilde{r}$ as follows:

$$\begin{aligned} \tilde{r} &= r(s, a) + F, \text{ where} \\ F(s, a, s') &= \gamma\phi(s') - \phi(s) \end{aligned} \tag{1}$$

and $\phi : \mathcal{S} \to \mathbb{R}$ is a real value function called potential function. This potential-based shaping is a necessary and sufficient condition for policy invariance by reward shaping [17].

Schulman et al. [22] interpreted GAE in terms of reward shaping and policy invariance principle. That is, the TD-residual $\delta_{t+l}^{GAE}$ of

GAE can be viewed as a potential-based shaped reward $\tilde{r}_{t+l}$, i.e.,

$$\delta_{t+l}^{GAE} = r_{t+l} + \gamma V(s_{t+l+1}) - V(s_{t+l}) =: \tilde{r}_{t+l} \tag{2}$$

due to the form of (1). Note that the form of $\delta_{t+l}^{GAE}$ directly matches the form in (1) with $\phi(s) = V(s)$. Then, the $\gamma\lambda$-discounted return of the transformed MDP becomes

$$\sum_{l=0}^{\infty} (\gamma\lambda)^l \tilde{r}_{t+l} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^{GAE} = A_t^{GAE(\gamma,\lambda)}. \tag{3}$$

Thus, maximizing the GAE advantage $A_t^{GAE(\gamma,\lambda)}$ is equivalent to maximizing the $\gamma\lambda$-discounted sum of shaped rewards. Hence, policy invariance holds for GAE when $\lambda = 1$. Note that the case of $\lambda = 1$ corresponds to the $\gamma$-discounted return exactly.

## 4 PROPOSED METHOD

In this section, we present our off-policy MARL algorithm named Off-policy Counterfactual Policy Optimization (OCPO). OCPO is an off-policy MARL algorithm that uses a Kullack-Leibler (KL) divergence-penalized PPO objective with off-policy samples and addresses multi-agent credit assignment with marginalized advantage estimation and potential-based counterfactual reward shaping.

### 4.1 Motivation

MAPPO [32] achieves impressive performance by adapting PPO to the multi-agent case. Just like single-agent PPO, MAPPO estimates an $n$-step advantage using GAE. GAE can reduce the variance of estimation with the bias-variance control parameter $\lambda$ and achieve policy invariance since it has the form of a discounted sum of potential-based shaped rewards. However, PPO has the disadvantage of sample inefficiency due to its on-policy nature. So, PPO-inherited MAPPO has the same problem of sample inefficiency.

In the case of single-agent policy gradient methods, there exist several works for reusing off-policy samples for improved sample efficiency. In particular, the $n$-step estimation with importance sampling is widely used for off-policy policy gradient methods. $N$-step estimation can help reducing the variance resulting from off-policy samples from past policies. GAE-V, which is a combination of GAE and V-trace, is an example of a successful off-policy extension of GAE. However, these methods are not directly applicable to the multi-agent case unfortunately because they target single-agent RL problems and were not designed with consideration of multi-agent credit assignment.

Several efforts have been made to solve the multi-agent credit assignment problem based on the multi-agent policy gradient methods [5, 13]. Foerster et al. [5] proposed the COMA advantage which can assign credits and guarantee policy invariance simultaneously. However, the COMA advantage does not exploit sequential behavior of agents because it assigns only 1-step credit assignment. To overcome this limitation, Li et al. [13] proposed a difference advantage to achieve $n$-step multi-agent credit assignment. The DAE advantage for agent $i$ is defined as [13]

$$A_t^{i,DAE} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^{i,DAE}, \tag{4}$$

where $\delta_{t+l}^{i,DAE} = r_{t+l} - \beta^{l+1}\mathbb{E}_{a^i}[r_{t+l}] + \gamma V_{t+l+1} - V_{t+l}$, and $a^i \sim \pi^i$ is the action of agent $i$. Here, $\delta_{t+l}^{i,DAE}$ can be viewed as the TD-residual with shaped reward $\tilde{r}_{t+l}^{i,DAE} = r_{t+l} - \beta^{l+1}\mathbb{E}_{a^i}[r_{t+l}]$. In this case, the reward shaping function of DAE is given by $F = -\beta^{l+1}\mathbb{E}_{a^i}[r_{t+l}]$. Otherwise, $\delta_{t+l}^{i,DAE}$ can be viewed as a shaped reward itself like in GAE. In the second case, $F = -\beta^{l+1}\mathbb{E}_{a^i}[r_{t+l}] + \gamma V_{t+l+1} - V_{t+l}$. In either case, the shaped reward is not in the form (1) of potential-based reward shaping, and hence policy invariance does not hold for DAE. Note that (1) is a necessary and sufficient condition for policy invariance by reward shaping.

The above facts motivate us to address the limitations of existing multi-agent policy gradient methods. In the following subsections, we introduce a new $n$-step advantage estimation method for multi-agent policy gradient with credit assignment, which controls the bias-variance trade-off, guarantees policy invariance with potential-based reward shaping, and increases sample efficiency by reusing off-policy samples. The properties of various methods are summarized in Table 1.

**Table 1: Advantage Estimators**

| Advantage | N-step Estimation | Policy-Invariance | Off-Policy Sample Reuse | Credit Assignment |
|---|---|---|---|---|
| **Ours** | **Yes** | **Yes** | **Yes** | **Yes** |
| GAE | Yes | Yes | No | No |
| GAE-V | Yes | Yes | Yes | No |
| DAE | Yes | No | No | Yes |
| COMA | No | Yes | No | Yes |

### 4.2 Counterfactual Potential-Based Advantage

As previously seen, the TD-residual $\delta_t^{GAE}$ of GAE can be viewed as a potential-based shaped reward with the potential function $\phi(s) = V(s)$. We can modify the potential function by choosing any function that has no gradient with respect to (w.r.t.) individual policy parameter $\theta^i$ of agent $i$. In our case, we choose the counterfactual baseline $\phi^i(s, \boldsymbol{a}^{-i}) = \mathbb{E}_{a^i\sim\pi^i}[Q(s, a^i, \boldsymbol{a}^{-i})]$ as the individual potential function. This allows us to shape the global reward into an individual reward for each agent. Thus, our proposed *counterfactual potential-based shaped reward* for agent $i$ is given by

$$\begin{aligned}\tilde{r}^i(s_t, \boldsymbol{a}_t) &= r(s_t, \boldsymbol{a}_t) + \gamma\phi^i(s_{t+1}, \boldsymbol{a}_{t+1}^{-i}) - \phi^i(s_t, \boldsymbol{a}_t^{-i}) \\ &= r(s_t, \boldsymbol{a}_t) + \gamma\mathbb{E}_{a_{t+1}^i\sim\pi^i}[Q(s_{t+1}, a_{t+1}^i, \boldsymbol{a}_{t+1}^{-i})] \\ &\quad - \mathbb{E}_{a_t^i\sim\pi^i}[Q(s_t, a_t^i, \boldsymbol{a}_t^{-i})].\end{aligned} \tag{5}$$

The main difference of (5) from (2) is that (5) is individual for each agent $i$, whereas (2) used for MAPPO is common to all agents. (This commonality disables credit assignment for MAPPO.) Due to the form $\mathbb{E}_{a^i\sim\pi^i}[Q(s, a^i, \boldsymbol{a}^{-i})]$, (5) captures the TD residual of agent $i$'s action for given actions of other agents, dealing with credit assignment.

Before we apply the shaped reward to $n$-step off-policy advantage estimation, we show that the proposed reward shaping guarantees policy invariance in the following proposition.

Proposition 4.1 (**Policy Invariance**). *The proposed reward shaping (5) guarantees policy invariance.*

Proof. Available at [1]. □

### 4.3 Off-Policy Value Estimation

MAPPO and PPO compute the advantage and corresponding value by using GAE($\lambda$). Since they use only on-policy sample batch generated by current policy for update, off-policy correction is not necessary for their GAE($\lambda$). To use old sample batches generated by old policies for update, however, we need off-policy correction with an importance sampling scheme for learning and stabilizing the variance. V-trace [4] presents an off-policy version of GAE($\lambda$):

$$\hat{V}_t = V_t + \sum_{l=t}^{T-1} (\gamma)^{l-t} \left( \prod_{j=t}^{l} c_j \right) \delta_l, \tag{6}$$

where $c_t = \min(\rho_t, \bar{c})$ is the truncated importance sampling weight at time $t$ with some upper bound $\bar{c}$. From (6), Han and Sung [7] showed that an off-policy estimation of $n$-step advantage is available as

$$\hat{A}_t = \sum_{l=t}^{T-1} (\gamma\lambda)^{l-t} \left( \prod_{j=t+1}^{l} c_j \right) \delta_l. \tag{7}$$

Basically, combining this importance sampling scheme and our counterfactual potential-based reward shaping, we obtain our off-policy advantage estimator. However, we slightly modify the importance sampling scheme to be suitable for multi-agent systems. That is, for more aggressive correction for change in an individual policy while being less affected by changes in other policies, we propose a double-truncated importance sampling weight

$$c_t^{i,DT} = \min\left( \min(\boldsymbol{\rho}_t^{-i},\ 1) \cdot \rho_t^i,\ 1 \right) \tag{8}$$

for the advantage estimate of each agent $i$. Thus, our final *counterfactual potential-based advantage estimator* enabling $n$-step off-policy estimation for agent $i$ is given by

$$\hat{A}_t^{i,OCPO} = \sum_{l=t}^{T-1} (\gamma\lambda)^{l-t} \left( \prod_{j=t}^{l} c_j^{i,DT} \right) \delta_l^{i,OCPO}, \tag{9}$$

where

$$\delta_l^{i,OCPO} = r_l + \gamma \mathbb{E}_i[Q_{l+1}] - \mathbb{E}_i[Q_l]$$

is the counterfactual potential-based TD residual with simplified notation $\mathbb{E}_i[Q_t] := \mathbb{E}_{a_t^i \sim \pi^i}[Q(s_t, a_t^i, a_t^{-i})]$, and $c_j^{i,DT}$ is the double-truncated importance sampling weight in (8). Like GAE($\lambda$), our counterfactual potential-based advantage guarantees policy invariance when $\lambda = 1$, and we can control the bias-variance trade-off of the estimator by choosing $\lambda \in (0, 1]$.

### 4.4 Off-Policy Sample Reuse

To improve the sample efficiency, we reuse old data samples from the replay buffer $R$. The used samples are the $M$ sample batches $B_k, B_{k-1}, \ldots, B_{k-M+1}$, generated from the current policy $\boldsymbol{\pi}_k$ and old policies $\boldsymbol{\pi}_{k-1}, \ldots, \boldsymbol{\pi}_{k-M+1}$, where $\boldsymbol{\pi}_l$ denotes the policy at the $l$-th iteration. Old sample reuse improves the learning efficiency but can make the update unstable when the data distribution is too shifted from the current policy distribution. To prevent this, we consider the penalty loss term for policy update as

$$L_l^{KL}(\pi^i) = \mathbb{E}_{o^i \in B_l} \left[ D_{KL}(\pi^i(\cdot|o^i) \parallel \pi_l^i(\cdot|o^i)) \right]. \tag{10}$$

$L_l^{KL}$ makes the sample reuse more stable by ensuring that the policy update does not deviate excessively from the behavioral policy. We assign a weight factor $\alpha_{KL}$ to $L_l^{KL}$ and update $\alpha_{KL}$ adaptively to make $L_l^{KL}$ become to get closer to a target value $L_{targ}^{KL}$ in a similar to that in [23]:

$$\alpha_{KL} = \begin{cases} \alpha_{KL} \times \sqrt{2} & \text{if } \mathbb{E}_l[L_l^{KL}] > L_{targ}^{KL} \times 1.5 \\ \alpha_{KL} / \sqrt{2} & \text{if } \mathbb{E}_l[L_l^{KL}] < L_{targ}^{KL} / 1.5 \\ \alpha_{KL} & otherwise. \end{cases} \tag{11}$$

The KL-penalty term helps prevent the batches stored in the buffer from drifting too far away from the current policy. In addition to this KL-penalty term, we apply a further step to prevent too much drifting. That is, we use a batch inclusion parameter $\epsilon_b > 0$, which allows excluding the batches that have large difference from the current policy distribution. For an old sample batch $B_l$, if the average importance sampling shift

$$s_l = \mathbb{E}_{B_l} \left[ \left| 1 - \frac{\pi_k^i}{\pi_l^i} \right| + 1 \right] \tag{12}$$

is larger than $\epsilon_b$, we exclude the batch for the update step. Thus, in our formulation, the inclusion parameter and the KL-divergence loss prevent the data distribution from changing too quickly.

Now, our final objective for policy update with reusing old sample batches is given by

$$L(\theta) = \frac{1}{MN} \sum_{i=1}^{N} \sum_{j=k-M+1}^{k} \left( \frac{\pi_\theta^i}{\pi_j^i} \sum_{t \in j} \hat{A}_t^{i,OCPO} - L_j^{KL}(\pi_\theta^i) \right). \tag{13}$$

Note that in the objective function (13), the contribution of agent $i$ is captured by $\hat{A}_t^{i,OCPO}$ which can be different across the index $i$. To implement an individual policy $\pi^i$, we use the method of parameter sharing with one-hot vector [15]. That is, we use a shared policy network with input of observation and one-hot vector indicating individual agent, and $\boldsymbol{\theta}$ is the parameter of this shared policy network.

To compute $\delta_t^{i,OCPO}$ necessary for computation of $\hat{A}_t^{i,OCPO}$, we need to learn the $Q$-function. For this, we parameterize the $Q$-function with parameter $\phi$ and use the following loss for update:

$$L(\phi) = \frac{1}{MN} \sum_{l=k-M+1}^{k} \sum_{i=1}^{N} (Q_\phi - \hat{Q}^i)^2, \tag{14}$$

where the target value is given by [16]

$$\hat{Q}^i(s_t) = Q^i(s_t) + \hat{A}_t^{i,OCPO}. \tag{15}$$

We again use the method of parameter sharing with one-hot vector [15] for the implementation of the $Q$-function. Algorithm 1 summarizes the proposed algorithm.

## 5 EXPERIMENTS

We here provide empirical results and ablation studies.

**Algorithm 1** OCPO

Initialize KL-penalty coefficient $\alpha_{KL}$, shared policy parameter $\boldsymbol{\theta}$. shared value parameter $\boldsymbol{\phi}$ for $N$ agents
Initialize replay buffer $R$ with capacity $M$
**for** each iteration $k$ **do**
  Collect a trajectory $B_k$ from the environment by the joint policy $\boldsymbol{\pi}_k$
  Store $B_k$ in the replay buffer $R$
  Initialize the update buffer $U$
  **for** each sample batch $B_l$ in $R$ **do**
    **if** $s_l$ in (12) is less than $\epsilon_b$ **then**
      Store $B_l$ in $U$
    **end if**
  **end for**
  Compute $\hat{A}_t^{i,OCPO}$ and $\hat{Q}^i$ with (9) and (15) by using the samples stored in $U$
  **for** each gradient step **do**
    Update $\boldsymbol{\theta}$ with policy objective (13)
    Update $\boldsymbol{\phi}$ with value objective (14)
  **end for**
  Update $\alpha_{KL}$ with (11)
**end for**

## 5.1 Simulation Setup

We evaluated the proposed method on two widely-used benchmark environments for MARL: Multi-Agent MuJoCo (MAMuJoCo) and StarCraft Multi-Agent Challenge (SMAC).

***Multi-Agent MuJoCo (MAMuJoCo).*** MAMuJoCo is a benchmark for cooperative multi-agent tasks with a single robot, where the single robot is represented as a body graph partitioned into disjoint subgraphs and each subgraph is one agent [19]. Each agent can observe the controllable joints of the $k$-nearest agents. We considered $k = 0$, which is the most difficult partially observable setting.

***StarCraft II Multi-Agent Challenge (SMAC).*** SMAC is one of the most popular environments to evaluate cooperative MARL algorithms. The conventional reward setting of SMAC is dense, but we consider the modified sparse SMAC environment proposed in [9]. In the modified environment, a reward is given only when some units die or win a battle and hence the problem is more difficult. The reward setting for the dense and sparse reward cases is shown in Table 2.

**Table 2: Reward setting for SMAC**

| | Dense Reward | Sparse Reward |
|---|---|---|
| All enemies die (Win) | +200 | +200 |
| One enemy dies | +10 | +10 |
| One ally dies | -5 | -5 |
| Enemy's health | -Enemy's remaining health | - |
| Ally's health | +Ally's remaining health | - |
| Other Components | +/- with other components | |

## 5.2 Performance Comparison

***MAMuJoCo.*** We compared OCPO on four MAMuJoCo tasks `6x1-Halfcheetah-v2`, `2x3-Halfcheetah-v2`, `2x3-Walker2d-v2`, and `10x2-manyagent-swimmer` with three baselines: MAPPO [32], MAPPO with COMA [5] advantage, and HAPPO [11].

Fig. 1 shows the learning curves of our method OCPO and the baseline algorithms, and Table **??** shows the maximum average episode reward for each algorithm after learning $5M$ time steps. It is observed that the on-policy OCPO performs similarly to MAPPO in some environments but performs better than the baselines in `6x1-Halfcheetah-v2`, which requires a high degree of credit assignment. Here, by on-policy OCPO we mean that we train our OCPO by using only the current sample batch without using previous sample batches, although OCPO can use previous sample batches. It is also seen that the proposed off-policy OCPO consistently outperforms all baselines on the considered environments. Indeed, our off-policy advantage estimator and sample reuse yield significant performance improvement.

Table **??** shows the max average episodic return of the considered algorithms in the MAMuJoCo tasks. It is seen that OCPO also outperforms other baselines in terms of max average episodic return.

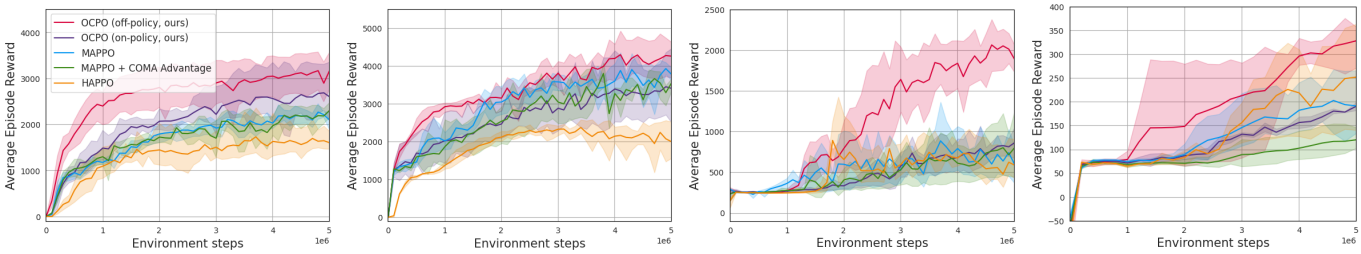
***SMAC.*** We compared OCPO on four sparse-reward SMAC tasks with five major baselines: MAPPO [32], MAPPO with COMA advantage [5], QMIX [21], MASER [9], and ROMA [29].

Fig. 2 shows the performance of the proposed algorithm OCPO and the baselines on the sparse SMAC environment. It is seen that OCPO shows superior performance to existing state-of-the-art MARL baselines. In the challenging tasks such as `3s5z-sparse` and `2m_vs_1z-sparse`, where it is difficult to obtain sparse rewards, OCPO demonstrates significant performance improvement over the baselines. Especially, our off-policy OCPO demonstrates superior performance to other baselines in `3s5z-sparse`, where all other algorithms fail to learn. Note that even on-policy OCPO fails to learn in `3s5z-sparse`. Thus, it is indeed important to use old sample batches to improve performance.

***Comparison with DAE.*** As mentioned, DAE is also a method for $n$-step advantage estimation with credit assignment for MARL, sharing the common goal with OCPO. We compared OCPO with DAE based on MAPPO as the backbone algorithm. Fig. 3a shows the performance of OCPO and DAE when both algorithms use only on-policy batch samples. In this on-policy case, OCPO yields noticeable performance gain over DAE even in the case that both algorithms use on-policy samples only. Fig. 3b shows the performance of OCPO and DAE when both algorithms use both on-policy and off-policy batch samples. Recall that DAE is proposed for on-policy learning. Hence, the performance of DAE degrades with off-policy samples, as seen by comparing Figs. 3a and 3b. On the other hand, OCPO targeting off-policy learning yields significant performance improvement with off-policy samples. Note that the performance gap between these two algorithms is large.

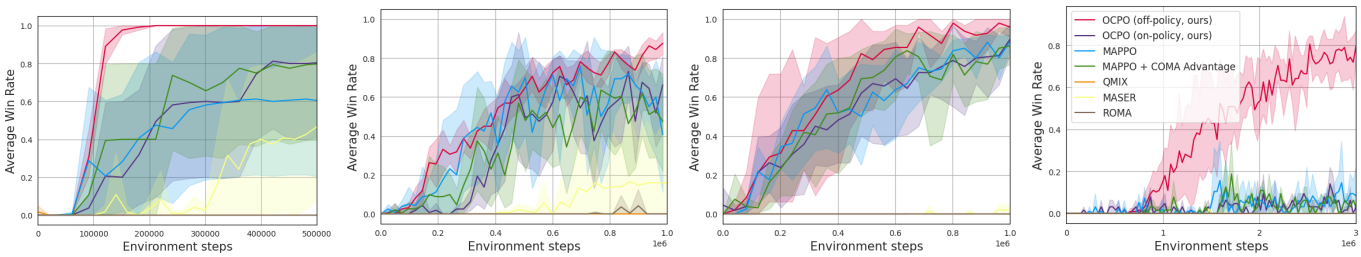
## 5.3 Ablation Study

To assess the impact of each component of OCPO, we performed an ablation study on off-policy correction (9) and KL-divergence



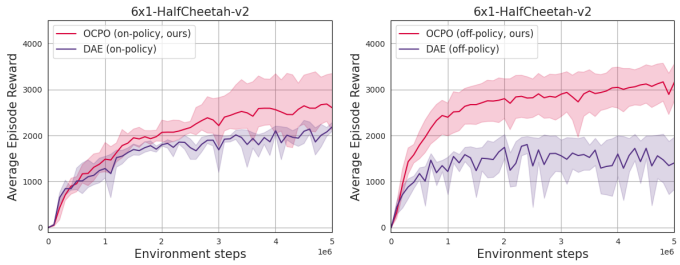

(a) 6x1-HalfCheetah-v2 (b) 2x3-HalfCheetah-v2 (c) 2x3-Walker2d-v2 (d) 10x2-manyagent-swimmer

**Figure 1: Performance comparison on MAMuJoCo tasks (the legends in (a) are valid to (b),(c) and (d).)**



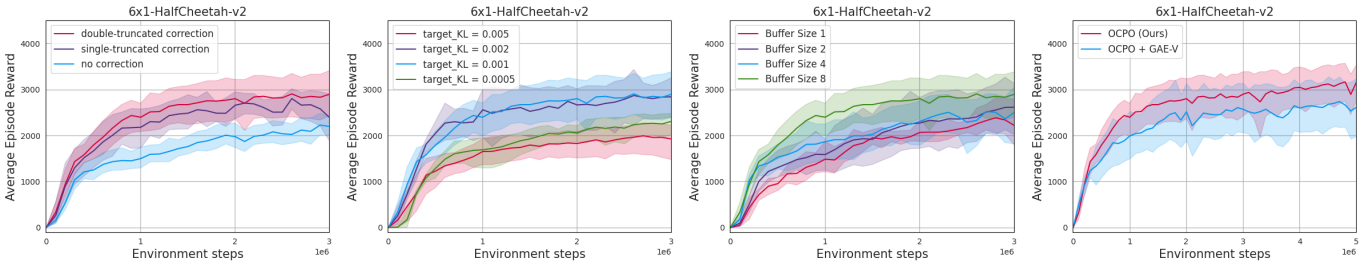

(a) 2m_vs_1z-sparse (b) 8m-sparse (c) 2s3z-sparse (d) 3s5z-sparse

**Figure 2: Performance comparison on the considered sparse SMAC tasks (the legends in (d) are valid for (a), (b) and (c).)**




(a) On-policy learning (b) Off-policy learning

**Figure 3: Performance comparison with DAE**




(a) Off-policy correction (b) KL-divergence loss (c) Sample reuse buffer size (d) Advantage Estimator

**Figure 4: Ablation study: Impact of each component of OCPO**

loss (10). To see the effectiveness and compatibility of OCPO with reusing old samples, we also investigated the performance of OCPO w.r.t. the replay buffer size. Finally, we compared our proposed advantage estimator with existing GAE-V, to see the compatibility of our advantage with OCPO.

***Off-Policy Correction.*** Off-policy correction with importance sampling ratio is a commonly-used technique for off-policy policy gradient methods. Since we proposed a new correction method of double-truncated importance sampling, we tested how this new correction affects the performance. Fig. 4a shows that our proposed correction method improves performance over single-truncated correction and our method without correction. Here, 'our method without correction' means that we eliminated the off-policy correction term $\prod_{j=t}^{l} c_j^{i,DT}$ in (9). 'Single-truncated correction' means that the off-policy correction term $\prod_{j=t}^{l} c_j^{i,DT}$ in (9) is replaced by $\prod_{j=t}^{l} c_j$ with $c_j$ in (7). Indeed, the double-truncated correction method newly proposed for MARL outperforms the conventional single-truncated correction method.

***KL Divergence Loss.*** The KL-divergence loss is also used in PPO, but we extend this to reuse off-policy samples. Policy update and PPO surrogate are obtained from the current policy, whereas the KL-divergence loss is obtained from an old policy stored in the replay buffer. We performed an experiment by varying the target value for the KL divergence loss. Fig. 4b shows that this loss is effective for preventing policy updates from moving away from past samples and causing distribution shifts.

***Replay Buffer Size.*** We performed an experiment by varying the length of the old sample batches for sample reuse to see the impact of old sample batches on the performance. In Figure 4c, it is seen that increasing the buffer size significantly improves the performance. The performance gain is almost monotonic w.r.t. the reuse batch length up to 8. Note that this corresponds to 8 times increase in sample efficiency compared with on-policy learning.

***Advantage Estimation.*** Now we address how the proposed advantage estimator, potential-based counterfactual advantage, is superior to the existing single-agent n-step off-policy advantage estimator. We display the results of OCPO and OCPO with GAE-V, to show how our advantage estimator is relevant to deal with MARL problems. Fig. 4d shows that our advantage estimator performs better than GAE-V with off-policy learning. Here, 'OCPO + GAE-V' means that we use OCPO algorithm by replacing only the advantage estimator with GAE-V.

## 6 CONCLUSION

We have proposed a new multi-agent policy gradient algorithm, OCPO, that achieves both credit assignment and off-policy sample reuse. OCPO uses a new advantage estimator, which can assign a credit to each agent and control the trade-off between variance and bias. We have proven that the estimator guarantees policy invariance and extends to using off-policy sample batches. Our empirical results demonstrate that OCPO significantly improves performance on both continuous and discrete tasks with high sample efficiency. OCPO advantage estimation is compatible with other tate-of-the-art MARL algorithms requiring advantage estimation, yielding a potential for even better algorithms.

# REFERENCES


[1] Anonymous. 2023. Off-Policy Multi-Agent Policy Optimization with Multi-Step Counterfactual Advantage Estimation. (2023). Under Review.

[2] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. 2020. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533* (2020).

[3] Yali Du, Lei Han, Meng Fang, Ji Liu, Tianhong Dai, and Dacheng Tao. 2019. Liir: Learning individual intrinsic reward in multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 32 (2019).

[4] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. 2018. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*. PMLR, 1407–1416.

[5] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[6] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative multi-agent control using deep reinforcement learning. In *International conference on autonomous agents and multiagent systems*. Springer, 66–83.

[7] Seungyul Han and Youngchul Sung. 2019. Dimension-wise importance sampling weight clipping for sample-efficient reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2586–2595.

[8] Shariq Iqbal and Fei Sha. 2019. Actor-attention-critic for multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 2961–2970.

[9] Jeewon Jeon, Woojun Kim, Whiyoung Jung, and Youngchul Sung. 2022. Maser: Multi-agent reinforcement learning with subgoals generated from experience replay buffer. In *International Conference on Machine Learning*. PMLR, 10041–10052.

[10] Woojun Kim, Jongeui Park, and Youngchul Sung. 2020. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations*.

[11] Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. 2021. Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning. In *International Conference on Learning Representations*.

[12] Jakub Grudzien Kuba, Muning Wen, Linghui Meng, Haifeng Zhang, David Mguni, Jun Wang, Yaodong Yang, et al. 2021. Settling the variance of multi-agent policy gradients. *Advances in Neural Information Processing Systems* 34 (2021), 13458–13470.

[13] Yueheng Li, Guangming Xie, and Zongqing Lu. 2022. Difference Advantage Estimation for Multi-Agent Policy Gradients. In *International Conference on Machine Learning*.

[14] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *NIPS*.

[15] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. 2019. Maven: Multi-agent variational exploration. *Advances in Neural Information Processing Systems* 32 (2019).

[16] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. 2016. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems* 29 (2016).

[17] Andrew Y. Ng, Daishi Harada, and Stuart Russel. 1999. Policy invariance under reward transformations: Theroy and applications to reward shaping. In *International Conference on Machine Learning*.

[18] Frans A Oliehoek. 2012. Decentralized pomdps. In *Reinforcement Learning*. Springer, 471–503.

[19] Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. 2021. Facmac: Factored multi-agent centralised policy gradients. *Advances in Neural Information Processing Systems* 34 (2021), 12208–12221.

[20] James Queeney, Yannis Paschalidis, and Christos G Cassandras. 2021. Generalized proximal policy optimization with sample reuse. *Advances in Neural Information Processing Systems* 34 (2021), 11909–11919.

[21] 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.

[22] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2016. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *International Conference on Learning Representations*.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[24] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. 2016. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295* (2016).

[25] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*. PMLR, 5887–5896.

[26] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296* (2017).

[27] Richard S Sutton. 1988. Learning to predict by the methods of temporal differences. *Machine learning* 3, 1 (1988), 9–44.

[28] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[29] Tonghan Wang, Heng Dong, Victor Lesser, and Chongjie Zhang. 2020. Roma: Multi-agent reinforcement learning with emergent roles. *arXiv preprint arXiv:2003.08039* (2020).

[30] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. 2020. Dop: Off-policy multi-agent decomposed policy gradients. In *International Conference on Learning Representations*.

[31] Jiachen Yang, Jipeng Zhang, and Huihui Wang. 2020. Urban traffic control in software defined internet of things via a multi-agent deep reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems* 22, 6 (2020), 3742–3754.

[32] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. 2021. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955* (2021).

[33] Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, et al. 2020. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. *arXiv preprint arXiv:2010.09776* (2020).