

Multi-level Advantage Credit Assignment for Cooperative Multi-Agent Reinforcement Learning

Xutong Zhao

Mila - Quebec AI Institute
Polytechnique Montréal

Yaqi Xie

School of Computer Science,
Carnegie Mellon University

Abstract

Cooperative multi-agent reinforcement learning (MARL) aims to coordinate multiple agents to achieve a common goal. A key challenge in MARL is credit assignment, which involves assessing each agent's contribution to the shared reward. Given the diversity of tasks, agents may perform different types of coordination, with rewards attributed to diverse and often overlapping agent subsets. In this work, we formalize the credit assignment level as the number of agents cooperating to obtain a reward, and address scenarios with multiple coexisting levels. We introduce a multi-level advantage formulation that performs explicit counterfactual reasoning to infer credits across distinct levels. Our method, Multi-level Advantage Credit Assignment (MACA), captures agent contributions at multiple levels by integrating advantage functions that reason about individual, joint, and correlated actions. Utilizing an attention-based framework, MACA identifies correlated agent relationships and constructs multi-level advantages to guide policy learning. Comprehensive experiments on challenging Starcraft v1&v2 tasks demonstrate MACA's superior performance, underscoring its efficacy in complex credit assignment scenarios.

1 INTRODUCTION

Multi-agent systems (MAS) are inherently applicable to a broad spectrum of real-world applications, ranging from smart grid (Roesch et al., 2020) to swarm robotics

(Hüttenrauch et al., 2017), and autonomous vehicles (Shalev-Shwartz et al., 2016). Cooperative multi-agent reinforcement learning (MARL) emerges as a general learning framework for MAS with a common objective among agents. Naively applying single-agent RL methods by regarding multiple agents as one single agent with exponentially large joint action space is limited by its scalability, and potentially leads to non-stationary learning. A key challenge in MARL is multi-agent credit assignment, the task of identifying the contribution of each individual agent's action to the global reward (Albrecht et al., 2023). Consider a warehouse task where agents receive collective rewards for moving objects. In this scenario, one agent A might collaborate with agents B and C to carry a fridge, while another agent, D , does nothing. Disentangling each agent's contribution to the total collective reward is crucial to promoting effective cooperation strategies. However, credit assignment is highly non-trivial given only the joint actions and shared rewards. Real-world scenarios often involve more complex multi-agent interactions, rendering credit assignment more challenging.

Addressing the multi-agent credit assignment challenge efficiently and scalably remains an open question. Prior efforts to address it can be mainly categorized into: (1) implicit methods, such as QMix (Rashid et al., 2018), which learns a mixing network to decompose joint values, and (2) explicit methods, such as COMA (Foerster et al., 2018), which performs counterfactual reasoning to guide the learning of individual policies. However, both lines of works only consider cooperation among a certain fixed number of agents and overlook contributions from different or overlapping agent subsets.

Inspired by the study of human collaboration in cognitive neuroscience (Richerson et al., 2016), we propose a *multi-level* formulation to model the credit assignment in multi-agent cooperation. This approach considers the general case where each agent collaborates with distinct subsets of agents. We formalize the type of cooperation into distinct *levels*, where each level corresponds to the number of agents needed to obtain the reward

(detailed in Section 3.3). For example, in the warehouse scenario described above, agent A is involved in a 3-level cooperation. Additionally, agent A may simultaneously carry a backpack, resulting in both 1-level and 3-level cooperations. This coexistence of *multi-level* cooperation underscores the necessity of our *multi-level* credit assignment formulation, designed to explicitly recognize and assign credit across different levels of cooperation. More specifically, we propose a k -level counterfactual credit assignment formulation, further extending it to a general multi-level credit assignment to accommodate coexisting levels. Inspired by Foerster et al. (2018), we introduce a k -level counterfactual advantage function. The advantage is defined as the discrepancy between the state-action value of the taken joint action and a counterfactual baseline, which marginalizes out the k agents’ actions by their policies while keeping other agents’ actions fixed. This formulation *explicitly* performs counterfactual reasoning that deduces the joint contribution by the k -agent subset. Accounting for circumstances with coexisting different levels, we extend k -level credit assignment to multi-level credit assignment by amalgamation of multiple k -level advantages with different k ’s. Since each k -level advantage function is tailored to a specific level, the multi-level advantage captures contributions across different levels.

We then introduce Multi-level Advantage Credit Assignment (MACA), an actor-critic based approach to address the multi-agent credit assignment challenge based on our *multi-level* counterfactual credit assignment formulation. MACA’s multi-level advantage is constructed by three k -level advantages that respectively attribute contributions to individual actions, joint actions, and actions taken by the subset of strongly correlated agents. Although tasks may involve a variety of credit assignment levels, MACA focuses on the most significant ones. It adapts the transformer encoder architecture (Vaswani et al., 2017), leveraging its attention mechanism to model correlations among agents (Zhang et al., 2022), which is further used to construct the multi-level counterfactual advantage. The *dynamically* determined agent correlation serves as a basis for adaptive credit assignment at different states, which promotes effective contribution estimation and efficient learning across various scenarios.

We evaluate MACA on two popular cooperative MARL benchmarks, the StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019), and the newly extended SMACv2 benchmark, which introduces significantly greater stochasticity (Ellis et al., 2022). Tasks in both benchmarks cover various credit assignment types, making them ideal for testing multi-agent credit assignments. Our empirical findings highlight MACA’s ability to enhance performance effectively and

robustly, particularly in the more demanding contexts of the SMACv2 tasks. Additionally, ablation studies showcase that every single level is essential in MACA’s multi-level advantage. Theoretical analysis establishes an in-depth understanding of the strong performance.

Our main contributions are summarized as follows:

- Inspired by human collaboration, we propose a multi-level credit assignment formulation that recognizes and assigns credit to individual agents across different levels of collaboration.
- Based on the formulation, we present MACA, an actor-critic approach that addresses the multi-agent credit assignment challenge by leveraging multi-level counterfactual advantage functions to capture contributions of individual actions, joint actions, and actions by strongly correlated agents.
- Experiments on challenging Starcraft benchmarks demonstrate MACA outperforms previous state-of-the-art. Notably, it shows more significant advantages in tasks of higher complexity.

2 RELATED WORK

Recent works on multi-agent credit assignment can be categorized into two lines of approaches based on the utilized assignment mechanism: (1) *implicit methods*, which implicitly learns agent contribution via a decomposition of the joint value function, and (2) *explicit methods*, which rely on some explicit mechanism to infer agents’ contributions.

Implicit Credit Assignment Most prior works perform implicit credit assignment by learning a decomposition of the joint value function to per-agent value functions (Sunehag et al., 2017; Rashid et al., 2018; Son et al., 2019; Wang et al., 2020a). VDN (Sunehag et al., 2017) assumes linearity and obtains the joint value by summing over all individual values. QMix (Rashid et al., 2018) lifts the linearity assumption by learning a mixing neural network, but its monotonicity constraints on the mixing weights also limit its expressiveness. HAPPO (Kuba et al., 2021a) is based on the multi-agent advantage decomposition and employs a sequential policy update scheme, but the computational complexity of sequential updates limits its scalability.

Explicit Credit Assignment One exemplar family calculates *difference rewards* against a reward baseline (Tumer and Agogino, 2007; Wolpert and Tumer, 2001). COMA (Foerster et al., 2018) is an actor-critic (AC) method implementing this idea through a counterfactual advantage baseline. However,

it treats each agent as an independent entity without considering interactions within the group, thus can be inefficient in learning complex cooperative behaviours (Papoudakis et al., 2020; Li et al., 2021). Other AC methods such as MAPPO (Yu et al., 2021) and MAA2C (Papoudakis et al., 2020) share a similar formulation, but their advantage is only concerned with joint actions’ contribution, without distinguishing individual agents’ credits. Limited by their credit assignment mechanism, these AC methods only deduce contributions of the same set of agents, ignoring the potential situations where one agent may contribute through different or multiple overlapping agent subsets.

Other representative methods leverage the Shapley Value (Shapley, 1953), but its computational complexity grows factorially with the number of agents (Kumar et al., 2020), handicapping efficient training in practice. Li et al. (2021) seeks to reduce computations by approximating Shapley Value through Monte Carlo sampling, but this formulation sacrifices certain properties of Shapley Value, such as symmetry. Other explicit methods employ reward shaping. Li et al. (2022) applies potential-based difference rewards, but learning its reward model introduces more computational cost compared to model-free approaches.

Attention in MARL Adopting transformers or attention mechanisms (Vaswani et al., 2017) to MARL has grown in popularity in recent years. Meng et al. (2021) extends the Decision Transformer (Chen et al., 2021) to the offline MARL regime. Baker et al. (2019) uses an attention-based architecture to learn permutation-invariant state representation and demonstrates self-supervised autocurriculum in the hide-and-seek task. Seraj et al. (2022) constructs a heterogeneous graph-attention network architecture to promote efficient multi-agent communication. Nayak et al. (2023) also use the attention mechanism for inter-agent communication, seeking to tackle the 2D navigation problem. However, few works have leveraged attention-captured correlations to address the credit assignment challenge in cooperative MARL.

3 PRELIMINARIES

3.1 Dec-POMDP

We consider the cooperative multi-agent setting as a decentralized partially observable Markov decision process (Dec-POMDP) (Oliehoek and Amato, 2016), which is formally defined as a tuple $G = \langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, R, \gamma, \mathcal{O}, \Omega \rangle$, where $\mathcal{N} = \{1, \dots, n\}$ is the set of agents, \mathcal{S} is the state space, $\mathcal{A} = \mathcal{A}_1 \times \dots \times \mathcal{A}_n$ is the joint action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition probability, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is the

global reward function, $\gamma \in [0, 1]$ is the discount factor, $\Omega = \Omega_1 \times \dots \times \Omega_n$ is the joint observation space, $\mathcal{O} : \mathcal{S} \times \mathcal{N} \rightarrow \Omega$ is the observation function. At each timestep t , agents are in a state $s^t \in \mathcal{S}$. Each agent $i \in \mathcal{N}$ observes a local observation $o_i^t = \mathcal{O}(s^t, i) \in \Omega_i$, and selects an action $a_i^t \in \mathcal{A}_i$. All actions together form the joint action $a^t = [a_i^t]_{i=1}^n$. The environment transitions to the next state $s^{t+1} \sim P(s'|s^t, a^t)$, and outputs a global reward $r \in \mathbb{R}$ shared across all agents.

Let $\mathcal{H}_i = (\Omega_i \times \mathcal{A}_i)^* \times \Omega_i$ represent the space of agent i ’s action-observation history. A trajectory is a full joint history sequence $\tau = \langle o^t, a^t \rangle_{t=0}^\infty$. We use $\pi = \pi_1 \times \dots \times \pi_n$ to denote the joint policy, where each $\pi_i(a_i^t|h_i^t) : \mathcal{H}_i \times \mathcal{A}_i \rightarrow [0, 1]$ is agent i ’s policy distribution that samples its action at every timestep. The marginal state distribution d^π is induced by the policy π and the transition probability P . For the simplicity of notations, in subsequent sections we assume the Markov property and full observability, and we may ignore superscripts/subscripts if the context is unambiguous. We use $-i$ to denote all agents except agent i .

The discounted return is defined as $G^t = \sum_{k=0}^\infty \gamma^k r^{t+k}$. Based on agents’ joint policy, the state-value function and action-value function are defined as $V_\pi(s) = \mathbb{E}_{s^{1:\infty} \sim d^\pi, a^{0:\infty} \sim \pi} [G^0 | s^0 = s]$ and $Q_\pi(s, a) = \mathbb{E}_{s^{1:\infty} \sim d^\pi, a^{1:\infty} \sim \pi} [G^0 | s^0 = s, a^0 = a]$, respectively. By definition $V_\pi(s) = \mathbb{E}_{a \sim \pi} [Q_\pi(s, a)]$. The advantage function is defined as $A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s)$. The MARL objective is to find a joint policy that maximizes the expected return $J(\theta) = \mathbb{E}_\tau [G^0]$.

3.2 Policy Gradient Methods

In RL, policy gradient (PG) methods optimize the policy π by performing gradient updates on the objective $J(\theta)$, where π is parameterized by θ . The Policy Gradient Theorem (Sutton et al., 1999) provides the gradient with respect to θ as $\nabla_\theta J(\theta) = \mathbb{E}_{s \sim d^\pi, a \sim \pi} [Q(s, a) \nabla_\theta \log \pi_\theta(a|s)]$. The MAPG theorem (Foerster et al., 2018; Zhang et al., 2018) extends the single-agent policy gradient to

$$g_{\theta_i} = \nabla_{\theta_i} J(\theta) = \mathbb{E}_{s \sim d^\pi, a \sim \pi} [Q(s, a) \nabla_{\theta_i} \log \pi_i(a_i|s)] \quad (1)$$

In the AC framework, a critic learns the value function, constructing the gradient to update the actor, i.e. the policy. A popular variance-reduction approach is to subtract the Q -value estimate by a baseline function b (Weaver and Tao, 2013). One common choice $b(s) = V(s)$ gives the advantage function $Q(s, a) - V(s)$ by definition. We can show that any action-independent baseline function preserves unbiasedness of the gradient estimate in expectation, i.e., $\mathbb{E}_{s \sim d^\pi, a \sim \pi} [b_i \nabla_{\theta_i} \log \pi_i(a_i|s)] = 0$ for any agent $i \in \mathcal{N}$

if b_i does not depend on a_i (see Appendix A for proof).

In this work, we follow the centralized training with decentralized execution (CTDE) (Bernstein et al., 2002) MARL learning paradigm. Extending AC approaches to MARL, the CTDE formulation is usually centralized critic and decentralized actors, where the critic learns the value function based on the global state information (and potentially joint actions), which is used to optimize the policies following Equation (1).

3.3 Credit Assignment Level

The concept of credit assignment level is inspired by the coordination level in MARL tasks (Liu et al., 2022). Let $\mathcal{G} \subset \mathcal{N}$ denote a subset of $|\mathcal{G}| = k$ agents. Different subsets are allowed to overlap. At each timestep t , we use $r_{\mathcal{G}}(s^t, a^t) = r_{\mathcal{G}}(s^t, a_{\mathcal{G}}^t)$ to denote the joint reward that can only be obtained if the subset of k agents cooperate, where $a_{\mathcal{G}}^t = \{a_i^t | i \in \mathcal{G}\}$. We then represent the global reward as the sum of rewards contributed by all subsets of agents: $r(s^t, a^t) = \sum_{\mathcal{G} \subset \mathcal{N}} r_{\mathcal{G}}(s^t, a^t)$. We consider the number of agents k as a credit assignment level. We then consider credit assignment and associated contributions with different levels as different types. This formalism intends to cover complex situations that involve an individual agent in different credit assignment types across timesteps, or even coexisting different types at the same timestep.

4 METHOD

In this section, we introduce MACA, an actor-critic MARL method that *explicitly* models multiple levels of credit assignment. This approach delineates three distinct advantage functions, each designed to assess contributions from individual actions, joint actions, and actions taken by strongly correlated partners. With multi-level contributions encoded by the combination of these advantages, MACA provides a comprehensive framework for addressing multi-agent credit assignment challenges, accommodating the diverse range of interactive behaviors among agents.

4.1 Multi-Level Counterfactual Formulation

k -Level Advantage Motivated by Foerster et al. (2018), we consider the advantage in the MAPG estimator as an approach to counterfactually assessing agents' contributions. With a centralized critic learning $Q(s, a)$, when we compute the advantage function $A_i(s, a)$ for each agent i , the actions marginalized out in the Q -based baseline correspond to the agents whose contributions we reason about. In particular we can

write a k -level counterfactual baseline

$$b_i^{\text{CF}}(s, a) = \mathbb{E}_{a_{\mathcal{G}_i}}[Q(s, a)] \quad (2)$$

where $a_{\mathcal{G}_i} = \{a_j \sim \pi_j : j \in \mathcal{G}_i\}$ and \mathcal{G}_i is an arbitrary subset of $|\mathcal{G}_i| = k$ agents that satisfies $\{i\} \subset \mathcal{G}_i \subset \mathcal{N}$. The resulting advantage $A_i(s, a) = Q(s, a) - b_i^{\text{CF}}(s, a)$ provides a general form of counterfactual advantage that explicitly reasons about credit assignment by the k -agent subset. For instance, COMA computes the baseline $b_i^{\text{IND}}(s, a) = \mathbb{E}_{a_i \sim \pi_i}[Q(s, a)]$, thereby reasoning about the *individual action's* contribution. MAPPO/MAA2C estimates the baseline $b_i^{\text{JNT}}(s, a) = V(s) = \mathbb{E}_{a \sim \pi}[Q(s, a)]$. Intuitively, this advantage evaluates how much the *joint action* is better or worse than the default behaviour. As all actions are marginalized out, they do not differentiate each particular agent's contribution. It is trivial to show that the b_i^{CF} encompasses both b_i^{IND} and b_i^{JNT} .

While the minimum-variance baseline b_i^* (see Theorem A.2) is usually infeasible to compute in complex MDPs, our k -level counterfactual baseline b_i^{CF} also reduces the variance of MAPG estimates and assists learning in practice. The rational is clear when each k -level baseline is expressed as $b_i^{\text{CF}} = b_i^* - \frac{\text{Cov}(Q, \|\nabla_i \log \pi_i\|^2)}{\mathbb{E}[\|\nabla_i \log \pi_i\|^2]}$ (see Lemma A.3). The term $\text{Cov}(Q, \|\nabla_i \log \pi_i\|^2)$ typically becomes negative as the gradient becomes smaller on actions with high returns during the optimization process, leading to the baseline being an optimistic baseline (Chung et al., 2021), which justifies our design choice.

We consider one important subset \mathcal{G}_i whose agents are strongly correlated with agent i . We name this subset the *CorrSet* \mathcal{C}_i , and the corresponding baseline the *CorrSet* baseline b_i^{COR} . We evaluate inter-agent correlations conditioned on the state information so that b_i^{COR} is capable of adapting dynamically to different credit assignment levels across timesteps. We discuss how we determine \mathcal{C}_i in detail in the following subsection.

Multi-Level Advantage As discussed in previous sections, we consider the situation where one agent's contribution may simultaneously involve a mixture of *multiple* credit assignment levels. We aim to reason about each of them using a different k -level advantage function, and combine them into one advantage, which we call the *multi-level* counterfactual advantage. To obtain such an advantage, due to the same action value function used in all advantage functions, equivalently we only need to compute corresponding k -level baselines and a combination of them. In particular, for each agent i , we compute the three most important k -level baselines: (1) $k = n$: the joint action set baseline b_i^{JNT} , (2) $k = 1$: the individual action set baseline b_i^{IND} , and (3) $k = |\mathcal{C}_i|$: a *CorrSet* baseline

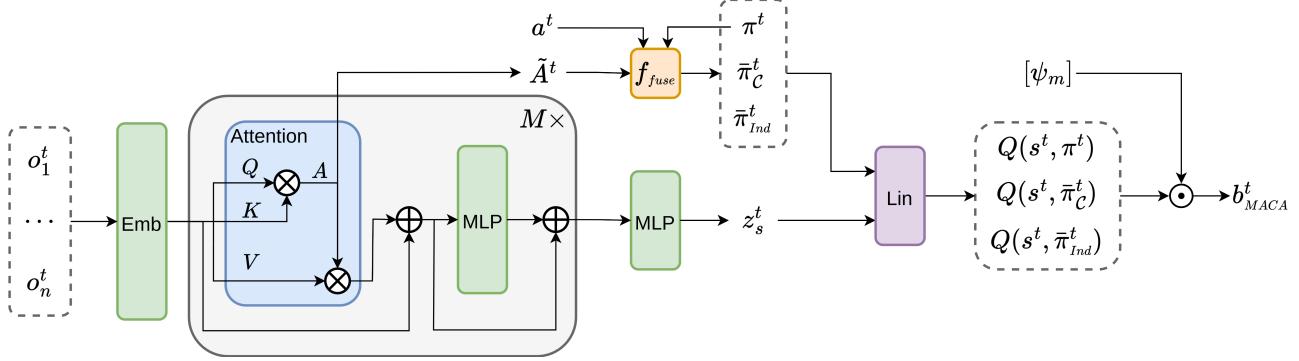


Figure 1: MACA critic architecture. The sequence of agent observations (o_1^t, \dots, o_n^t) inputs to an embedding layer and a self-attention encoder, and the output passes through an MLP layer to produce the state embedding z_s . The attention weight matrix \tilde{A}^t , joint actions a^t , and joint policy distribution π^t pass through the function f_{fuse} to obtain action distributions $\bar{\pi}_C^t, \bar{\pi}_{Ind}^t$, corresponding to CorrSet and individual actions, respectively. The embedding z_s and action distributions are fed to a linear layer to get respective Q values, and the coefficients $[\psi_m]$ weight them to obtain the final MACA baseline b_{MACA}^t .

b_i^{COR} . In principle, a multi-level advantage does not limit the number of k -level advantage components.

We then obtain our MACA baseline by integrating these three baselines by a weighted sum

$$b_i^{\text{MACA}} = \psi_i^{\text{JNT}} b^{\text{JNT}} + \psi_i^{\text{IND}} b^{\text{IND}} + \psi_i^{\text{COR}} b^{\text{COR}} \quad (3)$$

$$A_i^{\text{MACA}} = Q(s, a) - b_i^{\text{MACA}} \quad (4)$$

where the weighting coefficients $[\psi^m]_{m \in [3]} \in \Delta(3)$ are state-dependent. Hence the resulting advantage function A_i^{MACA} reasons different credit assignment levels via its multi-level baselines. We discuss how we learn the coefficients using stochastic optimization in the following subsection. As each of the k -level baselines marginalizes out the action a_i , the resulting MACA baseline is also independent of a_i , which preserves unbiasedness in policy gradient estimates (see Lemma A.1). This property ensures MACA advantage is generally compatible with the MAPG framework and applicable to various algorithms. Lemma A.4 derives the convergence of MACA to a local optimal policy, by following the convergence proof of single-agent actor-critic (Sutton et al., 1999; Konda and Tsitsiklis, 1999) subjecting to the same assumptions. We hereby provide a proof sketch. The detailed proof is deferred to Appendix A.

Proof sketch of Lemma A.4. Since each k -level baseline marginalizes out the action a_i , the multi-level baseline as a linear combination of all k -level baselines is independent of a_i . By Lemma A.1 it does not introduce bias to MAPG estimates. Therefore the multi-level baseline does not affect the convergence in expectation. Writing the joint policy as the product of individual policies, the remaining proof directly follows Konda and Tsitsiklis (1999).

4.2 Attention-based Framework

In this section, we introduce our learning framework that constructs the MACA advantage function. We discuss how we utilize the self-attention mechanism to quantify agent-wise relative correlation and construct the *CorrSet*. We then present how we perform optimization to learn policies, values, and parameters that compute the MACA baseline. The MACA architecture is illustrated in Figure 1.

Critic Encoder Architecture The critic leverages the multi-agent transformer encoder (Vaswani et al., 2017; Wen et al., 2022). It inputs the sequence of agents' observations (o_1^t, \dots, o_n^t) to an embedding layer and M encoding blocks. Each block consists of a self-attention mechanism, a multi-layer perceptron (MLP) layer, and residual connections and layer normalizations (Ba et al., 2016) to avoid gradient vanishing. The output passes through an MLP layer to produce a state embedding z_s . We compute the attention rollout weights (Abnar and Zuidema, 2020) of the last layer, denoted as \tilde{A} (referred to as attention weights for simplicity).

Value Estimation We aim to compute any k -level counterfactual baseline $b_i^{\text{CF}}(s, a)$. To improve scalability and flexibility, we adopt a design that inputs the joint *action distribution parameters* (Wierstra and Schmidhuber, 2007) by $b_i^{\text{CF}}(s, a) = \mathbb{E}_{a_{\mathcal{G}_i}}[Q(s, a)] = Q_\phi(s, \bar{\pi}_{\mathcal{G}_i})$, where $\bar{\pi}_{\mathcal{G}_i} = \mathbb{E}_{a_{\mathcal{G}_i}}[a] \in \mathbb{R}^{|\mathcal{A}|}$ is the action/policy distribution, and Q_ϕ is the value approximation parameterized by ϕ . Regarding per-agent policies in $\bar{\pi}_{\mathcal{G}_i}$, the marginalized actions $\{a_j : j \in \mathcal{G}_i\}$ correspond to the original policies, and all other policies are one-hot with the probability of taken actions equal to 1. We overuse the same notation to denote the policy

Table 1: Overview of advantage functions.

Algorithm	Update	Advantage function $A_i(s, a)$
COMA	Simultaneous	$Q(s, a) - \mathbb{E}_{a_i \sim \pi_i} [Q(s, a)]$
MAPPO	Simultaneous	$Q(s, a) - \mathbb{E}_{a \sim \pi} [Q(s, a)]$
IPPO	Simultaneous	$Q_i(s, a_i) - \mathbb{E}_{a_i \sim \pi_i} [Q_i(s, a_i)]$
PPO-Sum	Simultaneous	$Q_i(s, a_i) - \mathbb{E}_{a_i \sim \pi_i} [Q_i(s, a_i)] ; Q = \sum_i Q_i$
PPO-Mix	Simultaneous	$Q_i(s, a_i) - \mathbb{E}_{a_i \sim \pi_i} [Q_i(s, a_i)] ; Q = f_{\text{Mix}}([Q_i]), \frac{\partial Q}{\partial Q_i} \geq 0$
HAPPO	Sequential	$(\prod_{j=1}^{i-1} \frac{\bar{\pi}_j(a_j s)}{\pi_j(a_j s)}) (Q(s, a) - \mathbb{E}_{a \sim \pi} [Q(s, a)])$
MACA-COR (ours)	Simultaneous	$Q(s, a) - \mathbb{E}_{a_{\mathcal{C}_i}} [Q(s, a)] ; \{i\} \subset \mathcal{C}_i \subset \mathcal{N}$
MACA (ours)	Simultaneous	$Q(s, a) - (\psi^1 \mathbb{E}_{a \sim \pi} [Q(s, a)] + \psi^2 \mathbb{E}_{a_i \sim \pi_i} [Q(s, a)] + \psi^3 \mathbb{E}_{a_{\mathcal{C}_i}} [Q(s, a)]) ; \{i\} \subset \mathcal{C}_i \subset \mathcal{N}, [\psi^m] \in \Delta(3)$

distribution whose joint action and individual action are marginalized out by $\bar{\pi}_i^{\text{JNT}}$ and $\bar{\pi}_i^{\text{IND}}$, respectively. Note that $\bar{\pi}_i^{\text{JNT}} = \pi$, the original joint policy.

It is worth noting that in general $\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a)] \neq Q(s, \bar{\pi}_{\mathcal{G}_i})$. While recent work has ignored this issue (Zhou et al., 2020), we seek a sound solution that ensures equality. As per Jensen’s inequality (Jensen, 1906) we feed the state embedding z_s and $\bar{\pi}_{\mathcal{G}_i}$ through a *linear* layer $b_i^{\text{CF}}(s, a) = \text{Lin}(z_s, \bar{\pi}_{\mathcal{G}_i})$. To validate this layer’s expressivity, we investigate a transformer-decoder modeling option in ablation experiments in Section 5.3.

CorrSet Construction We utilize inter-agent relationships represented by the pre-computed attention weights \tilde{A} to infer the *CorrSet* \mathcal{C}_i . Note that this component only relies on linear computations.

For each agent i , $\tilde{A} \in \mathbb{R}^n$ contains weights associated with all n agents, each of which is denoted by $\tilde{A}_{i,j}, j \in \mathcal{N}$. The self-attention mechanism reasons correlations among input tokens (Zhang et al., 2022). Intuitively, higher attention weights indicate stronger correlations between input tokens, and vice versa. Hence to identify the subset of strongly correlated agents, we find agents with high attention weights – i.e., let $j \in \mathcal{C}_i$ if $\tilde{A}_{i,j} \geq \sigma$, where $\sigma \in [0, 1]$ is a thresholding hyperparameter. To preserve unbiasedness in gradient estimates, we additionally enforce $i \in \mathcal{C}_i$ regardless of $\tilde{A}_{i,i}$ as per definition of k -level baseline. With the *CorrSet* \mathcal{C}_i settled, we can follow the procedure described above to compute all baselines ($b^{\text{JNT}}, b_i^{\text{IND}}, b_i^{\text{COR}}$) and further the MACA advantage.

Training Losses We follow the standard actor-critic training paradigm. We optimize the actors through the MAPG as in Equation (1). We update value function parameters ϕ on-policy by optimizing the mean-squared

TD error over collected trajectories:

$$L_V(\phi) = \mathbb{E}_{\tau} [| | V(s^t) - \text{sg}(r^t + \gamma V(s^{t+1})) | |_2^2]$$

$$L_Q(\phi) = \mathbb{E}_{\tau} [| | Q(s^t, a^t) - \text{sg}(r^t + \gamma Q(s^{t+1}, a^{t+1})) | |_2^2]$$

where $V(s^t) = Q(s^t, \pi)$ as discussed above, and $\text{sg}(\cdot)$ indicates the stop-gradient operation.

To learn the weighting coefficients $[\psi_i^{\text{JNT}}, \psi_i^{\text{IND}}, \psi_i^{\text{COR}}]$ involved in the computation of MACA advantage (Equation (4)), it is important to realize they indirectly affect performance improvement. Since policy gradient updates only rely on the value of advantage functions, the MAPG objective is indifferentiable with respect to coefficients ψ ’s. Similarly the TD updates performed on value functions do not optimize the weights either. We hence consider the stochastic optimization setting. We compute each coefficient by a linear layer $\text{Lin}(z_s; \eta)$ parameterized by η , and apply a softmax over all coefficients to obtain a probability distribution. We leverage the CMA-ES method (Hansen et al., 2019) to optimize the performance difference between policies after and before policy updates using MACA advantages, that is, $L(\eta) = -\mathbb{E}_{\tau}[R(\theta^{n+1}) - R(\theta^n)]$, where $R(\theta)$ is the cumulative trajectory reward achieved by policy parameterized by θ .

5 EXPERIMENTS

In this section, we present empirical experiments and discuss the results. We first describe the experimental setup. Then we compare the performance of MACA and other state-of-the-art approaches. We also present ablation studies to demonstrate the critical role of MACA’s different components.

Table 2: Mean evaluation win rate and standard deviation for different methods on SMAC v1&v2 tasks. A win rate is marked in bold if it is within the critical region of the significance test.

	Task	Task Type	MACA	MAPPO	IPPO	PPO-Mix	PPO-Sum	COMA	HAPPO	Steps
SMAC	25m	homo.	99.3 (0.1)	99.5 (0.3)	99.5 (0.5)	25.0(3.4)	68.0(14.9)	0.0(0.0)	85.8(0.8)	3e6
	5m_vs_6m	homo.	87.0 (2.0)	75.2(1.5)	78.0(0.9)	24.5(13.3)	67.0(3.2)	0.9(0.5)	76.7(3.0)	8e6
	8m_vs_9m	homo.	99.0 (0.6)	94.5(1.9)	95.0(1.4)	57.0(8.1)	66.0(13.2)	1.0(0.6)	86.7(5.1)	8e6
	10m_vs_11m	homo.	100.0 (0.0)	87.0(8.6)	98.8(0.7)	22.4(1.7)	69.0(16.7)	3.0(2.0)	93.3(1.7)	8e6
	3s5z	hetero.	99.2 (0.8)	96.5(0.9)	99.0 (1.0)	97.5(0.8)	0.5(1.1)	0.5(1.1)	99.2 (0.8)	8e6
SMACv2	protoss_5_vs_5	hetero.	79.0 (3.4)	56.5(4.6)	54.2(2.7)	61.1(2.7)	32.5(6.8)	2.0(1.5)	50.0(2.5)	1e7
	terran_5_vs_5	hetero.	74.4 (8.3)	50.5(2.7)	57.7(2.5)	57.0(3.8)	44.1(6.7)	7.0(2.2)	55.8(3.6)	1e7
	zerg_5_vs_5	hetero.	63.4 (5.5)	42.3(1.4)	37.5(3.6)	41.5(4.4)	32.8(5.5)	4.5(0.9)	42.5(1.4)	1e7
	protoss_10_vs_10	hetero.	75.8 (3.9)	53.0(3.1)	38.0(4.0)	33.0(4.1)	26.0(3.4)	0.5(1.1)	21.7(4.2)	1e7
	terran_10_vs_10	hetero.	75.0 (5.2)	40.0(5.2)	34.7(0.8)	37.6(3.3)	33.1(6.6)	2.5(1.4)	17.5(3.8)	1e7
	zerg_10_vs_10	hetero.	62.9 (8.3)	39.8(3.0)	21.8(2.8)	32.0(2.2)	21.3(3.9)	1.0(0.6)	17.5(5.2)	1e7

5.1 Experimental Setup

Baseline Methods We compare MACA with SOTA multi-agent credit assignment approaches. To promote fair comparison, we adopt all methods into the MAPPO learning framework. We ensure the only difference among all methods is advantage function computation unless otherwise specified. We use well-established implementations of MAPPO/IPPO/HAPPO (Zhong et al., 2023), transformer encoder for MACA (Karpathy, 2023; Wen et al., 2022), and COMA/PPO-Mix/PPO-Sum (Papoudakis et al., 2020). We include the following methods, whose corresponding advantage functions are summarized in Table 1.

MAPPO optimizes all decentralized PPO (Schulman et al., 2017) actors with a centralized critic. MAPPO and IPPO are the SOTA on-policy MAPG algorithms in SMACv1 tasks (Yu et al., 2021).

IPPO (Independent PPO) directly adopts PPO that learns an individual critic for each actor. IPPO performs competitively with MAPPO across different environments (Papoudakis et al., 2020; Yu et al., 2021; de Witt et al., 2020).

HAPPO (Kuba et al., 2021a) extends MAPPO by the sequential policy update scheme. Experimental results show competitive performance to MAPPO on SMACv1&v2 (Zhong et al., 2023).

COMA (Foerster et al., 2018) performs counterfactual reasoning by marginalizing out the current agent’s action in the baseline, as mentioned in previous sections.

PPO-Mix is built upon the value decomposition method QMix (Rashid et al., 2018) that learns the joint Q -value as a monotonic function of individual Q values. PPO-Mix shares the same architecture as FACMAC (Peng et al., 2021), but the skeleton algorithm is the MAPPO rather than MADDPG to ensure a fair comparison.

PPO-Sum is identical to PPO-Mix, except the value

decomposition method is VDN (Sunehag et al., 2017), which represents the joint value function as the sum of individual value functions. PPO-Sum is analogous to DOP (Wang et al., 2020b), with MAPPO backbone instead of MADDPG for the same reason above.

Evaluation We evaluate MACA on challenging benchmark StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) (referred to as SMACv1 hereafter), and the recently proposed SMACv2 testbed (Ellis et al., 2022). We evaluate methods on five tasks from SMACv1, and six tasks from SMACv2, covering a broad spectrum of agent types and task scenarios. These tasks include both homogeneous and heterogeneous domains that ensure adequate coverage of task diversity and complexity, enabling a comprehensive empirical evaluation. We adopt the same hyperparameter settings from the original codebase and performed coarse finetuning, as detailed in Appendix C. We train each algorithm for 8M timesteps in SMACv1 tasks and 10M timesteps in SMACv2 tasks, with five random seeds per task. During training, agents are evaluated every 160k timesteps with 40 independent runs. We report the mean values and the standard deviation from evaluation win rates. We also conduct experiments on three Multi-Agent Particle (Lowe et al., 2017; Terry et al., 2021) (MPE) tasks following the same protocol. We report the mean value and the standard deviation of evaluation episodic returns.

5.2 Empirical Results

In this section, we present and discuss evaluation results from experiments on SMACv1&v2 testbeds. Table 2 summarizes the evaluation win rates for all methods. To determine the best performances, we perform a two-sample t-test (Snedecor and Cochran, 1980) with a significance level of 0.05 between the algorithm with the maximum mean and each of the other algorithms in every task. Win rates that are not significantly different from the best performance are marked in bold.

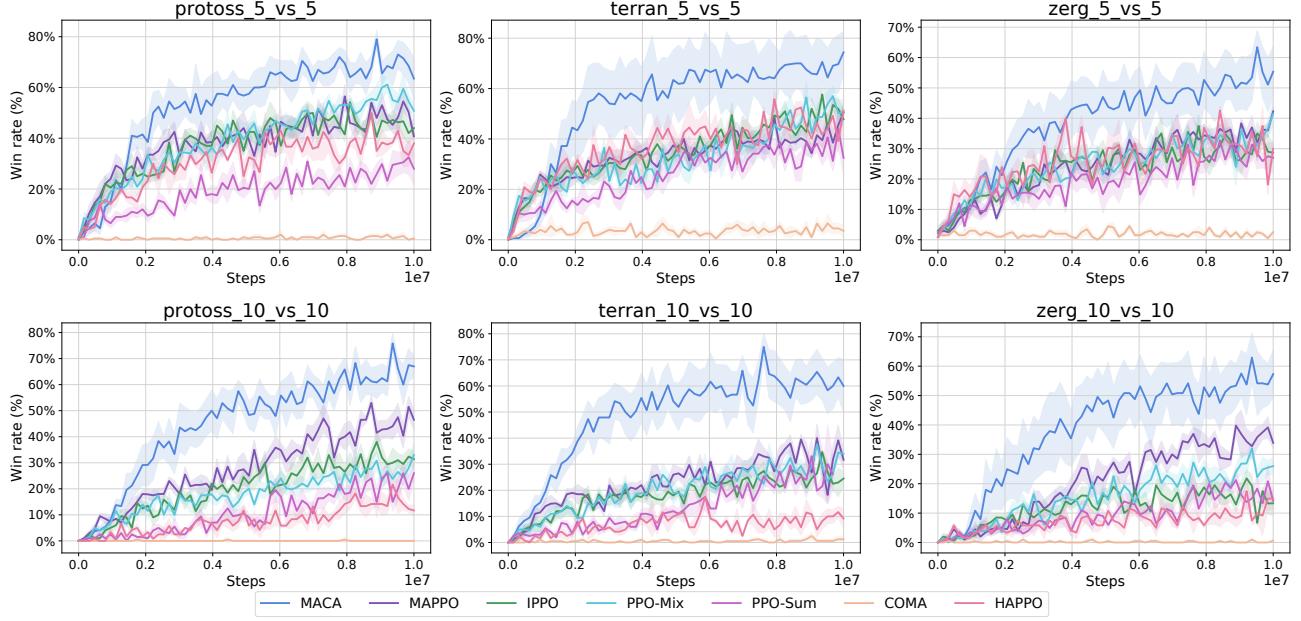


Figure 2: Performance on the SMACv2 benchmark.

Figure 2 shows the learning progress of all methods in SMACv2 benchmarks. We present learning curves in SMACv1 benchmarks, MPE results, and additional discussions in Appendix B.

Results in Table 2 and Figure 2 show that MACA reaches stronger overall performance compared to the SOTA in SMACv1 benchmark, and superior performance in more challenging SMACv2 benchmarks. Particularly in SMACv2 where MACA significantly outperforms all other methods, MACA gains higher sample efficiency. MACA improves much faster as learning progresses, while other methods tend to reach convergence. Such sample efficiency gain leads to higher final win rates. Although MAPPO and PPO-Mix also demonstrate strong performance in two tasks, by our statistical test MACA still significantly outperforms both.

In the SMACv1 benchmark, MACA achieves a superior overall performance. Although MAPPO/IPPO/HAPPO performs competitively in the majority of tasks, they show inferior performance in certain tasks; thus MACA is significantly better than at least one of them in four out of five tasks. Therefore, MACA demonstrates more *robustness* than other methods in general.

PPO-Mix generally performs well in SMACv2, but it occasionally fails in some SMACv1 tasks, e.g. in 25m. PPO-Sum shows large variances in SMACv1 tasks, and in general downperforms PPO-Mix.

COMA empirically performs poorly across all tasks in both SMACv1&v2. Such failure is consistent with

prior works (Papoudakis et al., 2020; Kuba et al., 2021b; Wang et al., 2022). Wang et al. (2022) argues COMA’s poor performance may be due to relative overgeneralization, a common game theoretic pathology that the suboptimal actions are preferred (Wei et al., 2018).

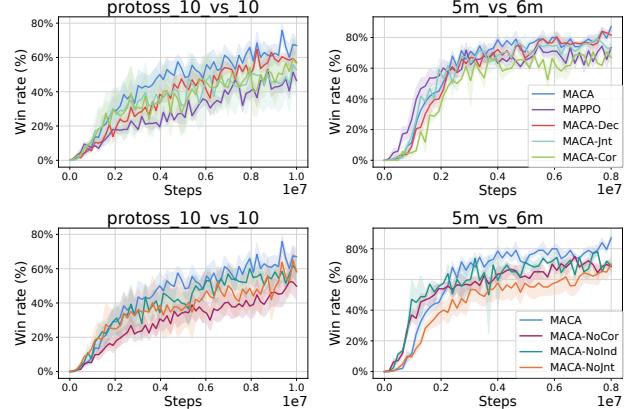


Figure 3: Ablation performance.

5.3 Ablations

In this section, we perform ablation experiments on MACA’s key components. Results are shown in Table 3 and Figure 3. We evaluate the following variants on 5m_vs_6m and protoss_10_vs_10.

MACA-Dec is algorithmically equivalent to MACA, with the linear layer replaced with a transformer decoder (Vaswani et al., 2017) to perform value estimation. A more detailed illustration is presented in Figure 7

Table 3: Mean evaluation win rate and standard deviation for MACA and its ablation variants.

Task	MACA	MACA-Dec	MACA-Jnt	MACA-Cor	MACA-Ind	MACA-NoCor	MACA-NoInd	MACA-NoJnt
5m_vs_6m	87.0(2.0)	84.0(2.3)	79.2(3.0)	70.6(1.9)	0.9(0.6)	75.0(1.8)	78.1(1.9)	70.0(4.7)
protoss_10_vs_10	75.8(3.9)	64.7(5.1)	55.8(15.2)	59.2(11.7)	0.3(0.9)	52.5(6.0)	61.9(6.7)	65.0(7.6)

in Appendix B.5. This variant aims to validate the expressivity of linear value estimation in MACA. Results show that MACA is comparable with *MACA-Dec* in both tasks, suggesting that the linear layer is capable of effectively estimating the value function.

To ablate the importance of the MACA advantage, we remove different baseline components by enforcing their corresponding weighting coefficients to 0 and applying softmax to the remaining coefficients to obtain a valid categorical distribution. *MACA-Jnt* only uses the joint action set baseline b^{JNT} . Since this variant is equivalent to MAPPO with a transformer encoder-based critic, we attempt to verify (1) the effect of transformer encoder compared with MLP, and (2) the importance of multi-level advantage. Results show that *MACA-Jnt* reaches similar performance as MAPPO yet still downperforms MACA. This demonstrates that improvement by only introducing a complex architecture is rather limited. The MACA advantage plays a more important role in performance improvement. *MACA-Cor* only uses the *CorrSet* baseline b^{COR} . This variant similarly downperforms MACA, highlighting the importance of the multi-level advantage.

MACA-NoCor, *MACA-NoInd*, and *MACA-NoJnt* remove the b^{COR} , b^{IND} , and b^{JNT} , respectively, seeking to evaluate individual components within the advantage baseline. Results indicate that each advantage term is essential and contributes to performance improvement.

In addition, we conducted ablation experiments on *MACA-Ind*, the variant that only keeps the b^{Ind} baseline. Similar to the equivalence between *MACA-Jnt* and MAPPO, *MACA-Ind* is essentially COMA, where the only difference is its attention-based critic. *MACA-Ind* can also demonstrate the limited effect of only introducing the attention encoder, which aligns with conclusions from *MACA-Jnt*. However, both COMA and *MACA-Ind* empirically perform poorly, making the comparison not as informative as that between *MACA-Jnt* and MAPPO.

6 CONCLUSIONS

In this work, we tackle the credit assignment problem in cooperative MARL by considering different levels of credit assignment explicitly. We formalize the per-level advantage that counterfactually deduces contributions over a specific level. We propose MACA,

an actor-critic method that constructs three different counterfactual advantage functions to respectively infer contributions from individual actions, joint actions, and actions taken by strongly correlated partners. MACA leverages a transformer-based architecture to capture agents' correlations via the attention mechanism. With multi-level contributions encoded by the combination of these advantages, MACA provides a generic formulation to address credit assignment challenges. Empirical evaluations on challenging Starcraft benchmarks underscore MACA's superior performance, showcasing its efficacy in complex cooperative MARL scenarios. Theoretical results provide support for MACA's strong performance.

In the future, it would be interesting to establish a connection between MACA and exploration methods via the advantage function, aiming to enhance performance across both regimes.

Limitations. This work addresses the multi-agent credit assignment challenge in the cooperative setting, where agents receive collective rewards. Other complex settings such as the mixed cooperative-competitive scenario could further complicate the problem, which is beyond the scope of this work.

Acknowledgments

We appreciate the anonymous reviewers for their insightful comments and constructive suggestions that improved the quality of this manuscript. We acknowledge the computational resources provided by Mila and the Digital Research Alliance of Canada.

References

- Abnar, S. and Zuidema, W. (2020). Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*.
- Albrecht, S. V., Christianos, F., and Schäfer, L. (2023). Multi-agent reinforcement learning: Foundations and modern approaches. *Massachusetts Institute of Technology: Cambridge, MA, USA*.
- Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2019). Emer-

- gent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*.
- Bernstein, D. S., Givan, R., Immerman, N., and Zilberman, S. (2002). The complexity of decentralized control of markov decision processes. *Mathematics of operations research*, 27(4):819–840.
- Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. (2021). Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097.
- Chung, W., Thomas, V., Machado, M. C., and Le Roux, N. (2021). Beyond variance reduction: Understanding the true impact of baselines on policy optimization. In *International Conference on Machine Learning*, pages 1999–2009. PMLR.
- Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. (2022). Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359.
- de Witt, C. S., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H., Sun, M., and Whiteson, S. (2020). Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*.
- Ellis, B., Cook, J., Moalla, S., Samvelyan, M., Sun, M., Mahajan, A., Foerster, J. N., and Whiteson, S. (2022). Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2212.07489*.
- Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018). Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.
- Hansen, N., Akimoto, Y., and Baudis, P. (2019). CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634.
- Hüttenrauch, M., Šošić, A., and Neumann, G. (2017). Guided deep reinforcement learning for swarm systems. *arXiv preprint arXiv:1709.06011*.
- Jensen, J. L. W. V. (1906). Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30(1):175–193.
- Karpathy, A. (2023). nanogpt. <https://github.com/karpathy/nanoGPT>.
- Konda, V. and Tsitsiklis, J. (1999). Actor-critic algorithms. *Advances in neural information processing systems*, 12.
- Kuba, J. G., Chen, R., Wen, M., Wen, Y., Sun, F., Wang, J., and Yang, Y. (2021a). Trust region policy optimisation in multi-agent reinforcement learning. *arXiv preprint arXiv:2109.11251*.
- Kuba, J. G., Wen, M., Meng, L., Zhang, H., Mguni, D., Wang, J., Yang, Y., et al. (2021b). Settling the variance of multi-agent policy gradients. *Advances in Neural Information Processing Systems*, 34:13458–13470.
- Kumar, I. E., Venkatasubramanian, S., Scheidegger, C., and Friedler, S. (2020). Problems with shapley-value-based explanations as feature importance measures. In *International Conference on Machine Learning*, pages 5491–5500. PMLR.
- Li, J., Kuang, K., Wang, B., Liu, F., Chen, L., Wu, F., and Xiao, J. (2021). Shapley counterfactual credits for multi-agent reinforcement learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 934–942.
- Li, Y., Xie, G., and Lu, Z. (2022). Difference advantage estimation for multi-agent policy gradients. In *International Conference on Machine Learning*, pages 13066–13085. PMLR.
- Liu, D., Shah, V., Boussif, O., Meo, C., Goyal, A., Shu, T., Mozer, M., Heess, N., and Bengio, Y. (2022). Stateful active facilitator: Coordination and environmental heterogeneity in cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2210.03022*.
- Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30.
- Meng, L., Wen, M., Yang, Y., Le, C., Li, X., Zhang, W., Wen, Y., Zhang, H., Wang, J., and Xu, B. (2021). Offline pre-trained multi-agent decision transformer: One big sequence model tackles all smac tasks. *arXiv preprint arXiv:2112.02845*.
- Nayak, S., Choi, K., Ding, W., Dolan, S., Gopalakrishnan, K., and Balakrishnan, H. (2023). Scalable multi-agent reinforcement learning through intelligent information aggregation. In *International Conference on Machine Learning*, pages 25817–25833. PMLR.
- Oliehoek, F. A. and Amato, C. (2016). *A concise introduction to decentralized POMDPs*. Springer.
- Papoudakis, G., Christianos, F., Schäfer, L., and Albrecht, S. V. (2020). Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. *arXiv preprint arXiv:2006.07869*.
- Peng, B., Rashid, T., Schroeder de Witt, C., Kamienny, P.-A., Torr, P., Böhmer, W., and Whiteson, S. (2021). Facmac: Factored multi-agent centralised policy gradients. *Advances in Neural Information Processing Systems*, 34:12208–12221.

- Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. (2018). Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International conference on machine learning*, pages 4295–4304. PMLR.
- Richerson, P., Baldini, R., Bell, A. V., Demps, K., Frost, K., Hillis, V., Mathew, S., Newton, E. K., Naar, N., Newson, L., et al. (2016). Cultural group selection plays an essential role in explaining human cooperation: A sketch of the evidence. *Behavioral and Brain Sciences*, 39:e30.
- Roesch, M., Linder, C., Zimmermann, R., Rudolf, A., Hohmann, A., and Reinhart, G. (2020). Smart grid for industry using multi-agent reinforcement learning. *Applied Sciences*, 10(19):6900.
- Samvelyan, M., Rashid, T., De Witt, C. S., Farquhar, G., Nardelli, N., Rudner, T. G., Hung, C.-M., Torr, P. H., Foerster, J., and Whiteson, S. (2019). The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Seraj, E., Wang, Z., Paleja, R., Patel, A., and Gombolay, M. (2022). Learning efficient diverse communication for cooperative heterogeneous teaming. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).
- Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016). Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*.
- Shapley, L. S. (1953). A value for n-person games. *Contribution to the Theory of Games*, 2.
- Snedecor, G. W. and Cochran, W. G. (1980). Statistical methods. iowa. *Iowa State University Press*.
- Starkstein, SE, & Robinson, RG (1989). Affective disorders and cerebral vascular disease. *The British Journal of Psychiatry*, 154:170–182.
- Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. (2019). Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pages 5887–5896. PMLR.
- Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. (2017). Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*.
- Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.
- Terry, J., Black, B., Grammel, N., Jayakumar, M., Hari, A., Sullivan, R., Santos, L. S., Dieffendahl, C., Horsch, C., Perez-Vicente, R., et al. (2021). Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:15032–15043.
- Tumer, K. and Agogino, A. (2007). Distributed agent-based air traffic flow management. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–8.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. (2020a). Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*.
- Wang, J., Zhang, Y., Gu, Y., and Kim, T.-K. (2022). Shaq: Incorporating shapley value theory into multi-agent q-learning. *Advances in Neural Information Processing Systems*, 35:5941–5954.
- Wang, Y., Han, B., Wang, T., Dong, H., and Zhang, C. (2020b). Off-policy multi-agent decomposed policy gradients. *arXiv preprint arXiv:2007.12322*.
- Weaver, L. and Tao, N. (2013). The optimal reward baseline for gradient-based reinforcement learning. *arXiv preprint arXiv:1301.2315*.
- Wei, E., Wicke, D., Freelan, D., and Luke, S. (2018). Multiagent soft q-learning. *arXiv preprint arXiv:1804.09817*.
- Wen, M., Kuba, J. G., Lin, R., Zhang, W., Wen, Y., Wang, J., and Yang, Y. (2022). Multi-agent reinforcement learning is a sequence modeling problem. *arXiv preprint arXiv:2205.14953*.
- Wierstra, D. and Schmidhuber, J. (2007). Policy gradient critics. In *European Conference on Machine Learning*, pages 466–477. Springer.
- Wolpert, D. H. and Tumer, K. (2001). Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(02n03):265–279.
- Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A., and Wu, Y. (2021). The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*.
- Zhang, F., Liu, B., Wang, K., Tan, V., Yang, Z., and Wang, Z. (2022). Relational reasoning via set transformers: Provable efficiency and applications to marl. *Advances in Neural Information Processing Systems*, 35:35825–35838.
- Zhang, K., Yang, Z., Liu, H., Zhang, T., and Basar, T. (2018). Fully decentralized multi-agent reinforcement

- learning with networked agents. In *International Conference on Machine Learning*, pages 5872–5881. PMLR.
- Zhong, Y., Kuba, J. G., Hu, S., Ji, J., and Yang, Y. (2023). Heterogeneous-agent reinforcement learning. *arXiv preprint arXiv:2304.09870*.
- Zhou, M., Liu, Z., Sui, P., Li, Y., and Chung, Y. Y. (2020). Learning implicit credit assignment for cooperative multi-agent reinforcement learning. *Advances in neural information processing systems*, 33:11853–11864.

Checklist

1. For all models and algorithms presented, check if you include:
 - (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes/No/Not Applicable] Yes.
 - (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes/No/Not Applicable] Yes.
 - (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes/No/Not Applicable] Not Applicable.
2. For any theoretical claim, check if you include:
 - (a) Statements of the full set of assumptions of all theoretical results. [Yes/No/Not Applicable] Yes.
 - (b) Complete proofs of all theoretical results. [Yes/No/Not Applicable] Yes.
 - (c) Clear explanations of any assumptions. [Yes/No/Not Applicable] Yes.
3. For all figures and tables that present empirical results, check if you include:
 - (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes/No/Not Applicable] Yes.
 - (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes/No/Not Applicable] Yes.
 - (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes/No/Not Applicable] Yes.
 - (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster,

- or cloud provider). [Yes/No/Not Applicable] Yes.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
 - (a) Citations of the creator If your work uses existing assets. [Yes/No/Not Applicable] Yes.
 - (b) The license information of the assets, if applicable. [Yes/No/Not Applicable] Yes.
 - (c) New assets either in the supplemental material or as a URL, if applicable. [Yes/No/Not Applicable] Yes.
 - (d) Information about consent from data providers/curators. [Yes/No/Not Applicable] Yes.
 - (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Yes/No/Not Applicable] Not Applicable.
 5. If you used crowdsourcing or conducted research with human subjects, check if you include:
 - (a) The full text of instructions given to participants and screenshots. [Yes/No/Not Applicable] Not Applicable.
 - (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Yes/No/Not Applicable] Not Applicable.
 - (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Yes/No/Not Applicable] Not Applicable.

Multi-level Advantage Credit Assignment for Cooperative Multi-Agent Reinforcement Learning: Supplementary Materials

A Theoretical Results

Lemma A.1. *Action-independent baseline functions do not affect the bias of the policy gradient estimate in expectation, i.e.,*

$$g_b = \mathbb{E}_{s \sim d^\pi, a \sim \pi} [b_i \nabla_{\theta_i} \log \pi_i(a_i|s)] = 0$$

for any agent $i \in \mathcal{N}$ if b_i does not depend on a_i .

Proof. We assume continuous action space. Proof for the discrete case is analogous.

$$\begin{aligned} g_b &= \mathbb{E}_{s \sim d^\pi, a \sim \pi} [b_i \nabla_{\theta_i} \log \pi_i(a_i|s)] \\ &= \mathbb{E}_{s \sim d^\pi, a_{-i} \sim \pi_{-i}} [b_i \mathbb{E}_{a_i \sim \pi_i} [\nabla_{\theta_i} \log \pi_i(a_i|s)]] . \end{aligned}$$

We have

$$\begin{aligned} \mathbb{E}_{a_i \sim \pi_i} [\nabla_{\theta_i} \log \pi_i(a_i|s)] &= \int_{\mathcal{A}_i} \pi_i(a_i|s) \nabla_{\theta_i} \log \pi_i(a_i|s) da_i \\ &= \int_{\mathcal{A}_i} \nabla_{\theta_i} \pi_i(a_i|s) da_i = \nabla_{\theta_i} \int_{\mathcal{A}_i} \pi_i(a_i|s) da_i = \nabla_{\theta_i} 1 = 0 \end{aligned}$$

which concludes the proof. □

Theorem A.2 (Minimum-variance baseline). *Adapted from single-agent results by Chung et al. (2021). The minimum-variance baseline for the MAPG estimator is*

$$b_i^*(s, a_{-\mathcal{G}_i}) = \frac{\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \|\nabla_{\theta_i} \log \pi_i(a_i|s)\|^2]}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla_{\theta_i} \log \pi_i(a_i|s)\|^2]}$$

if the baseline is conditioned on the state s and actions by $-\mathcal{G}_i$, where $a_{\mathcal{G}_i} = \{a_j \sim \pi_j : j \in \mathcal{G}_i\}$, $\{i\} \subset \mathcal{G}_i \subset \mathcal{N}$, and $-\mathcal{G}_i = \mathcal{N} \setminus \mathcal{G}_i$.

Proof. Assuming access to the Q-value for each state-action pair $Q(s, a) = Q^\pi(s, a)$, the gradient is given in Equation (1) as $\nabla_{\theta_i} J(\theta) = \mathbb{E}_{s \sim d^\pi, a \sim \pi} [Q(s, a) \nabla_{\theta_i} \log \pi_i(a_i|s)] = \mathbb{E}_{s \sim d^\pi, a \sim \pi} [g_i]$, where the estimator is $g_i = (Q(s, a) - b_i(s, a_{-\mathcal{G}_i})) \nabla_{\theta_i} \log \pi_i(a_i|s)$. We have

$$\text{Var}_{a \sim \pi} [g_i] = \underbrace{\text{Var}_{a_{-\mathcal{G}_i}} [\mathbb{E}_{a_{\mathcal{G}_i}} [g_i]]}_{\text{variance from given actions}} + \underbrace{\mathbb{E}_{a_{-\mathcal{G}_i}} [\text{Var}_{a_{\mathcal{G}_i}} [g_i]]}_{\text{variance from unknown actions}}$$

We now derive the baseline that minimizes the second term.

$$\begin{aligned} \text{Var}_{a_{\mathcal{G}_i}} (g_i) &= \mathbb{E}_{a_{\mathcal{G}_i}} [\|g_i\|^2] - \|\mathbb{E}_{a_{\mathcal{G}_i}} [g_i]\|^2 \\ &= \mathbb{E}_{a_{\mathcal{G}_i}} [\|g_i\|^2] - \|\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \nabla_{\theta_i} \log \pi_i(a_i|s)]\|^2 \end{aligned}$$

The equality holds by proof of Lemma A.1. Then we only need to consider the first item

$$\begin{aligned} \frac{\partial}{\partial b} \mathbb{E}_{a_{\mathcal{G}_i}} [\|g_i\|^2] &= \frac{\partial}{\partial b} \mathbb{E}_{a_{\mathcal{G}_i}} [\|Q(s, a) \nabla \log \pi_i(a_i | s)\|^2 - 2 \cdot Q(s, a) b_i(s, a_{-\mathcal{G}_i}) \|\nabla \log \pi_i(a_i | s)\|^2 \\ &\quad + b_i(s, a_{-\mathcal{G}_i})^2 \|\nabla \log \pi_i(a_i | s)\|^2] \\ &= 2(b_i(s, a_{-\mathcal{G}_i}) \cdot \mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \log \pi_i(a_i | s)\|^2] - \mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \|\nabla \log \pi_i(a_i | s)\|^2]) \end{aligned}$$

The minimum variance can be obtained by setting the gradient above to 0, i.e.,

$$b_i^*(s, a_{-\mathcal{G}_i}) = \frac{\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \|\nabla \theta_i \log \pi_i(a_i | s)\|^2]}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]}.$$

□

Lemma A.3. *Adapted from single-agent results by Chung et al. (2021).*

The minimum-variance baseline b_i^ from Theorem A.2 and the baseline $b_i = \mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a)]$ satisfies*

$$b_i = b_i^* - \frac{\text{Cov}_{a_{\mathcal{G}_i}} (Q(s, a), \|\nabla \theta_i \log \pi_i(a_i | s)\|^2)}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]}.$$

Proof. We have

$$\begin{aligned} b_i^*(s, a_{-\mathcal{G}_i}) &= \frac{\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \|\nabla \theta_i \log \pi_i(a_i | s)\|^2]}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]} \\ &= \frac{\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \|\nabla \theta_i \log \pi_i(a_i | s)\|^2]}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]} - b_i + b_i \\ &= \frac{\mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a) \|\nabla \theta_i \log \pi_i(a_i | s)\|^2] - \mathbb{E}_{a_{\mathcal{G}_i}} [Q(s, a)] \mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]} + b_i \\ &= \frac{\text{Cov}_{a_{\mathcal{G}_i}} (Q(s, a), \|\nabla \theta_i \log \pi_i(a_i | s)\|^2)}{\mathbb{E}_{a_{\mathcal{G}_i}} [\|\nabla \theta_i \log \pi_i(a_i | s)\|^2]} + b_i \end{aligned}$$

□

Lemma A.4. *Adapted from Foerster et al. (2018).*

For an actor-critic algorithm with a compatible TD(1) critic following a policy gradient

$$g^k = \mathbb{E}_{s \sim d^\pi, a \sim \pi} \left[\sum_i \nabla_{\theta^k} \log \pi_i(a_i | s) (Q(s, a) - b_i) \right]$$

at each iteration k , where the baseline function b_i does not depend on action a_i ,

$$\liminf_k \|\nabla J\| = 0 \quad w.p. 1.$$

Proof. By Lemma A.1, the per-agent baseline does not change the expected gradient, thereby not affecting the convergence. Hence we have the policy gradient

$$\begin{aligned} g &= \mathbb{E}_{s \sim d^\pi, a \sim \pi} \left[\sum_i \nabla_{\theta} \log \pi_i(a_i | s) (Q(s, a) - b_i) \right] \\ &= \mathbb{E}_{s \sim d^\pi, a \sim \pi} \left[\sum_i \nabla_{\theta} \log \pi_i(a_i | s) Q(s, a) \right] \\ &= \mathbb{E}_{s \sim d^\pi, a \sim \pi} \left[\nabla_{\theta} \log \prod_i \pi_i(a_i | s) Q(s, a) \right] \\ &= \mathbb{E}_{s \sim d^\pi, a \sim \pi} [\nabla_{\theta} \log \pi(a | s) Q(s, a)] \end{aligned}$$

Table 4: SMAC v1&v2 task scenarios with unit and task types.

	Task	Ally Unit Types	Task Type
SMAC	25m	Marine	homogeneous
	5m_vs_6m	Marine	homogeneous
	8m_vs_9m	Marine	homogeneous
	10m_vs_11m	Marine	homogeneous
	3s5z	Stalker, Zealot	heterogeneous
SMACv2	protoss_5_vs_5	Stalker, Zealot, Colossus	heterogeneous
	terran_5_vs_5	Marine, Marauder, Medivac	heterogeneous
	zerg_5_vs_5	Zergling, Hydralisk, Baneling	heterogeneous
	protoss_10_vs_10	Stalker, Zealot, Colossus	heterogeneous
	terran_10_vs_10	Marine, Marauder, Medivac	heterogeneous
	zerg_10_vs_10	Zergling, Hydralisk, Baneling	heterogeneous

yielding the standard single-agent actor-critic policy gradient.

Konda and Tsitsiklis (1999) prove that an actor-critic following this gradient converges to a local maximum of the expected return $J(\theta)$, given that:

1. the policy π is differentiable,
2. the update timescales for Q and π are sufficiently slow, and that π is updated sufficiently slower than Q , and
3. Q uses a representation compatible with π ,

amongst several further assumptions. The policy parameterization (i.e., the joint-action agent is decomposed into independent actors) is immaterial to convergence, as long as it remains differentiable. Note that a centralized critic with access to the global state is essential for this proof to hold. □

B Experiment Details

B.1 Experimental Setup

SMACv1 contains a diverse set of battle scenarios in which a team of ally units aims to defeat the enemy team. SMACv2 extends SMACv1 with procedurally generated scenarios, with team compositions and positions spawned according to different probabilities, which introduces significantly more stochasticity that requires agents' generalization to unseen settings. We follow the default reward setting, which returns positive rewards for damage dealt to the enemy, killing each enemy unit, and winning the battle. Tasks in both benchmarks embody complicated multi-agent credit assignment challenges, due to diverse cooperative behaviours involved in defeating the opponent. Table 4 provides an overview of SMAC v1&v2 task scenarios.

B.2 Computational Requirements

It is important to highlight that MACA utilizes the attention encoder module of the transformer, rather than the entire transformer architecture. Besides self-attention's internal computations (i.e., QKV projection, convex combination), other operations within MACA's critic, including policy distribution computation, value estimation, and weighted sum, are all linear operations. MACA hence enjoys reasonable time and space complexity compared to other baseline methods. In practice, MACA has memory and runtime on the same scale as other methods despite its transformer module, as shown in Table 6. Moreover, MACA's attention-based critic only affects parameter updates, while the majority of time consumed lies in agent-environment interactions, where only the actor networks are involved. As modern deep learning frameworks optimize the efficiency of attention (e.g. FlashAttention (Dao et al., 2022)), the use of attention would not bottleneck MACA's scalability.

We conduct all our experiments entirely on CPUs in compute clusters with multiple nodes. All experiments were run on a single CPU core. The main CPU model types are AMD Rome 7532 @ 2.40 GHz 256M cache L3 and AMD Rome 7502 @ 2.50 GHz 128M cache L3. Each SMACv1/SMACv2/MPE job respectively took 24/48/12 CPU hours. The total number of CPU hours spent (including hyperparameter search) is 33,708.

B.3 Additional Results and Discussions

Table 5: Mean evaluation episodic returns and standard deviation for different methods on MPE tasks.

	Task	MACA	MAPPO	IPPO	PPO-Mix	PPO-Sum	COMA	HAPPO	Steps
MPE	Spread	-61.95 _(1.44)	-67.96 _(4.22)	-67.01 _(2.40)	-106.56 _(0.92)	-84.29 _(1.14)	-109.09 _(3.61)	-72.25 _(2.05)	8e6
	Reference	-12.04 _(0.45)	-12.19 _(0.43)	-29.63 _(1.16)	-27.84 _(2.64)	-34.40 _(0.79)	-39.71 _(1.76)	-12.44 _(0.43)	8e6
	Speaker Listener	-9.17 _(0.53)	-9.10 _(0.53)	-9.41 _(0.58)	-21.26 _(2.52)	-26.06 _(1.55)	-31.80 _(2.94)	-9.06 _(0.51)	8e6

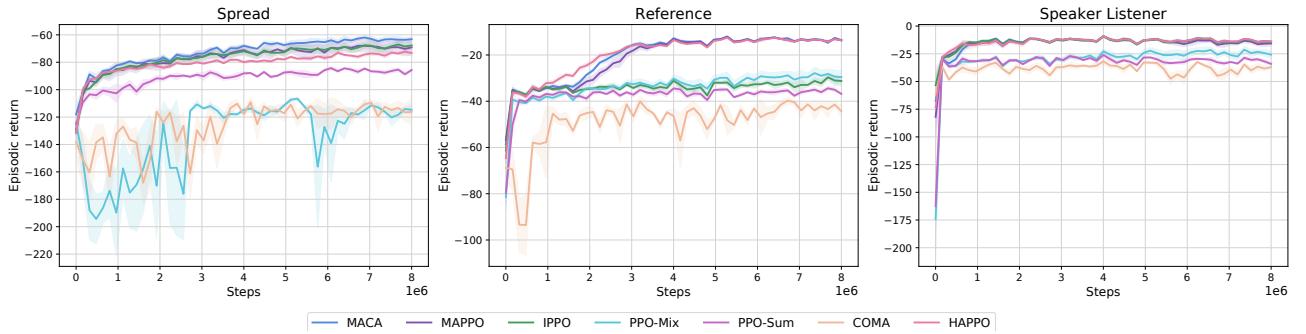


Figure 4: Performance on the MPE benchmark.

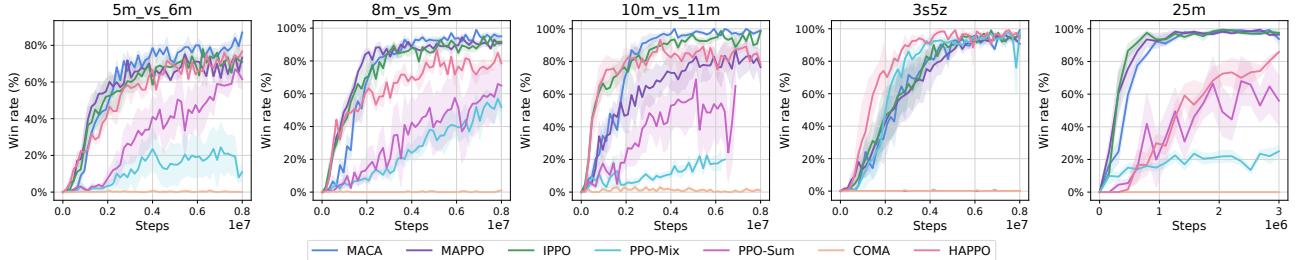


Figure 5: Performance on the SMAC benchmark.

On MPE tasks, multiple baseline methods exhibit strong performance. Despite continuous action space, MPE tasks are easy because they involve similar cooperation types and simple state-action spaces, leading to limited evaluation challenges. MACA marginally outperforms other methods unlike in the more challenging SMAC environments.

Table 6: Number of parameters in the critic model and experiment runtime. All values are reported in the case of 5m_vs_6m. MACA has memory and runtime requirements on the same scale as other methods.

Algorithm	Number of parameters	Runtime (hours)
MAPPO	165,747	15.1
IPPO	149,497	14.8
COMA	174,966	18.6
HAPPO	165,747	15.9
PPO-Mix	361,349	18.5
PPO-Sum	134,148	17.4
MACA	239,674	16.7

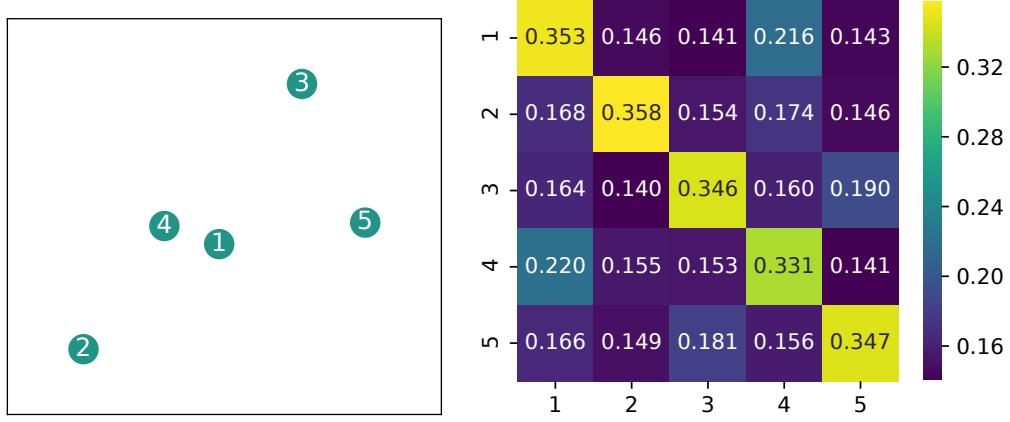


Figure 6: Example visualized ally agents’ coordinates (left) and corresponding attention weight (right) in $5m_vs_6m$. The attention matrix shows that agents 1 and 4 have high scores, which aligns with the task map where these agents are allied together.

B.4 Visualization

We present an exemplary visualization of ally coordinates and corresponding attention weights in Figure 6, which shows that the attention map reflects meaningful inter-agent correlations.

B.5 Ablation Details

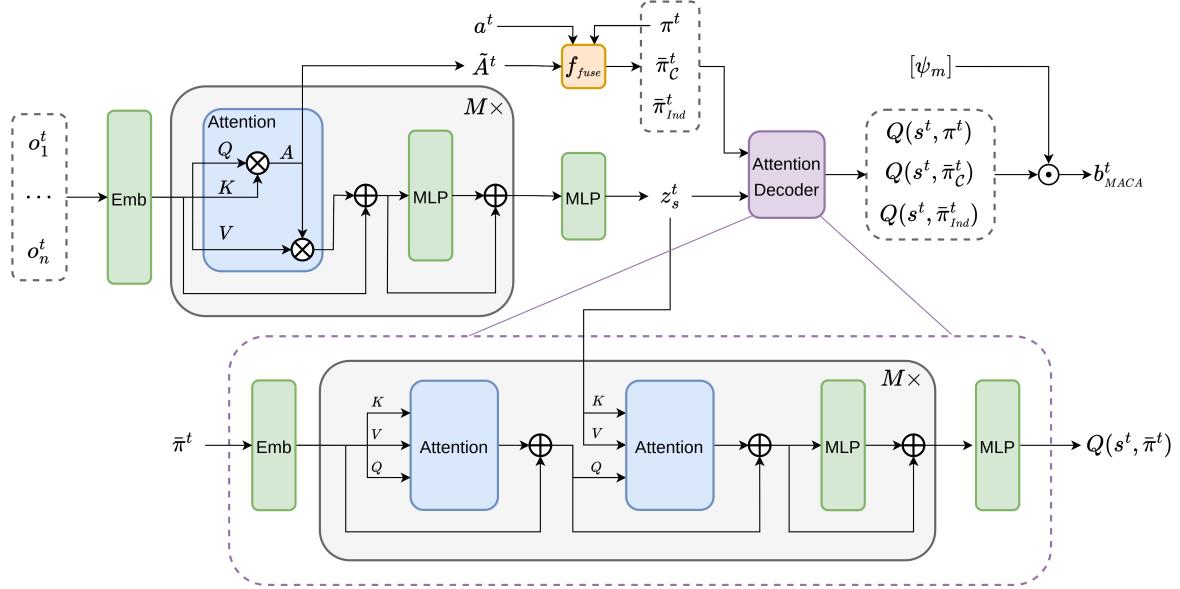


Figure 7: MACA critic architecture with transformer decoder.

MACA-Dec replaces the linear layer with a transformer decoder to perform value estimation. The architecture is illustrated in Figure 7.

C Hyperparameters

Table 7: Common hyperparameters across tasks.

Hyperparameter	Value
state type	EP
lr	0.0005
actor hidden sizes	[64,64,64]
critic hidden sizes	[128,128,128]
actor num mini batch	1
critic num mini batch	1
gamma	0.99
entropy coef	0.01
use valuenorm	True
use linear lr decay	False
use proper time limits	True
activation	ReLU
use feature normalization	True
initialization method	orthogonal
gain	0.01
use naive recurrent policy	False
use recurrent policy	True
num GRU layers	1
data chunk length	10
optim eps	1e-5
weight decay	0
std x coef	1
std y coef	0.5
use clipped value loss	True
value loss coef	1
use max grad norm	True
max grad norm	10.0
use GAE	True
GAE lambda	0.95
use huber loss	True
use policy active masks	True
huber delta	10.0
action aggregation	prod
share param	True

Table 8: Hyperparameters for MAPPO.

Task	ppo/critic epoch	clip param
SMAC	10	0.1
SMACv2	10	0.1
MPE	10	0.1

Table 9: Hyperparameters for IPPO.

Task	ppo/critic epoch	clip param
SMAC	10	0.1
SMACv2	10	0.1
MPE	10	0.1

Table 10: Common hyperparameters for MACA across tasks.

Hyperparameter	Value
critic hidden sizes	[64,64,64]
n encode layer	1
n head	1
n embed	64
zs dim	256
bias	True
active fn	gelu
weight decay	0.01
betas	[0.9, 0.95]
weight init	TFixup
warmup epochs	10
v value loss coef	1.0
q value loss coef	0.5
att sigma σ	1.0/n

Table 11: Hyperparameters for MACA.

Task	ppo/critic epoch	clip param
SMAC	10	0.05
SMACv2	10	0.1
MPE	10	0.1

Table 12: Hyperparameters for PPO-Mix.

Task	ppo/critic epoch	clip param
SMAC	10	0.05
SMACv2	10	0.05
MPE	10	0.05

Table 13: Hyperparameters for PPO-Sum.

Task	ppo/critic epoch	clip param
SMAC	10	0.05
SMACv2	10	0.05
MPE	10	0.05

Table 14: Hyperparameters for COMA.

Task	ppo/critic epoch	clip param
SMAC	5	0.05
SMACv2	5	0.05
MPE	5	0.05

Table 15: Hyperparameters for HAPPO.

Task	ppo/critic epoch	clip param
SMAC	10	0.1
SMACv2	10	0.1
MPE	10	0.1

Table 16: Hyperparameter sweep for MACA. Results demonstrate MACA’s robustness across ppo/critic epoch and clip hyperparameter settings in multiple representative environments.

epoch/clip	<i>25m</i>	<i>5m_vs_6m</i>	<i>protooss_5_vs_5</i>	<i>protooss_10_vs_10</i>
10/0.05	99.3 (0.1)	87.0 (2.0)	76.1 (3.5)	70.1 (2.6)
10/0.075	99.3 (0.1)	85.5 (2.8)	77.5 (4.8)	72.4 (4.1)
10/0.1	99.5 (0.1)	82.5 (4.1)	79.0 (3.4)	75.8 (3.9)
10/0.125	99.1 (0.0)	80.6 (3.7)	78.2 (2.7)	74.6 (5.2)
5/0.05	99.3 (0.3)	83.4 (2.3)	70.6 (6.4)	67.9 (6.0)
5/0.1	99.1 (0.1)	80.2 (2.5)	76.3 (5.6)	71.5 (7.6)

Table 17: Hyperparameter sweep for MACA. Results demonstrate MACA’s robustness across thresholding hyperparameter settings in multiple representative environments.

threshold σ	0.9/n	0.95/n	1.0/n	1.05/n	1.1/n
<i>5m_vs_6m</i>	86.5 (1.5)	87.4 (1.2)	87.0 (2.0)	86.6 (0.9)	85.7 (3.3)
<i>protooss_10_vs_10</i>	74.2 (4.4)	75.3 (4.6)	75.8 (3.9)	74.7 (3.6)	74.5 (5.0)