Individual Reward Assisted Multi-Agent Reinforcement Learning

Li Wang [*1 2] **Yupeng Zhang** [*1 2] **Yujin Hu** [2] **Weixun Wang** [3 4] **Chongjie Zhang** [5] **Yang Gao** [1] **Jianye Hao** [3 4]
Tangjie Lv [2] **Changjie Fan** [2]

Abstract

In many real-world multi-agent systems, the sparsity of team rewards often makes it difficult for an algorithm to successfully learn a cooperative team policy. At present, the common way for solving this problem is to design some dense individual rewards for the agents to guide the cooperation. However, most existing works utilize individual rewards in ways that do not always promote teamwork and sometimes are even counterproductive. In this paper, we propose *Individual Reward Assisted Team Policy Learning* (IRAT), which learns two policies for each agent from the dense individual reward and the sparse team reward with discrepancy constraints for updating the two policies mutually. Experimental results in different scenarios, such as the Multi-Agent Particle Environment and the Google Research Football Environment, show that IRAT significantly outperforms the baseline methods and can greatly promote team policy learning without deviating from the original team objective, even when the individual rewards are misleading or conflict with the team rewards.

1. Introduction

Many control problems in real life require the mutual cooperation between multiple agents. In recent years, cooperative multi-agent reinforcement learning (MARL) has been widely applied in many areas such as video games (Vinyals et al., 2019; Berner et al., 2019; Kurach et al., 2020), aerial vehicles (Jin & Ma, 2019), transportation (Shamsoshoara et al., 2019) and power grids (Jin & Ma, 2019). Cooperative multi-agent tasks generally only have rewards for team goals. In spite of the progress of the value-based and policy-based MARL algorithms (Sunehag et al., 2018; Rashid et al., 2018; Wang et al., 2021a; Lowe et al., 2017; Foerster et al., 2018; Iqbal & Sha, 2019; Wang et al., 2021b; Yu et al., 2021) made by researchers, the sparsity of the team rewards in many multi-agent systems still remains a big challenge for the state-of-the-art MARL algorithms. Practical application of these algorithms usually requires some dense individual rewards that can guide the agents perform the task. For example, in the case of solving a football game, although the team rewards (e.g., win or loss of the match, or scoring) are very sparse, we can manually assign positive rewards to the agents for performing a successful pass, a nice shot, and a good tackle, and assign negative rewards for unexpected behaviors such as offside and running out of the field.

There are usually two straightforward ways for utilizing individual rewards. The first one is to distribute the sum of the individual and team rewards of all agents equally among them. The second one is to generate a new reward function for each agent by adding its individual rewards to the team rewards. However, such simple methods face three problems: (1) The introduction of individual rewards will change the learning objectives of the agents, resulting in unexpected behaviors that deviate from the desired goal of the team. For example, for training agents to play football games, if too many individual rewards that encourage individual skills (such as passing, shooting, and tackling) are introduced, the agents may focus more on achieving those skills rather than winning the game. (2) The mixing of the individual and team rewards often involves weighting coefficients and the corresponding fine-tuning work in practice. (3) The mixture of the two types of rewards makes the credit assignment problem (Foerster et al., 2018; Son et al., 2019) more difficult. The learning of each agent is likely to be disturbed by the individual rewards of other agents. Still taking the football scenario as an example, if an agent runs out of the playing field, distributing its punishment to all agents will definitely affect the learning of other agents.

In this paper, we propose *Individual Reward Assisted Team Policy Learning* (IRAT), a novel multi-agent policy gradient

[*]Equal contribution [1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China [2]NetEase Fuxi AI Lab, Hangzhou, China [3]College of Intelligence and Computing, Tianjin University, Tianjin, China [4]Noah's Ark Lab, Beijing, China [5]Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China. Correspondence to: Yujing Hu <huyujing@corp.netease.com>, Yang Gao <gaoy@nju.edu.cn>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

algorithm which provides a new way of utilizing individual rewards to promote the cooperation between the agents. The key idea of our algorithm is to learn an individual policy and a team policy for each agent, and put discrepancy constraints on the two policies to guide the policy optimization process. On the one hand, the individual-reward policies of the agents are served as a way of exploration and sampling, and are encouraged to approach to the team policy gradually. On the other hand, the team policy, which is learnt from the sparse team rewards, constantly distills knowledge from the individual policies, with the goal of team cooperation kept unchanged at the same time. Experiments in scenarios with various individual and team reward settings demonstrate that our algorithm significantly outperforms the baseline methods and can greatly promote cooperation even when the individual rewards are misleading or conflict with the team rewards.

2. Background

In this section, we formalize the Dec-POMDPs with separated rewards and briefly introduce the MAPPO algorithm.

Dec-POMDPs: The multi-agent reinforcement learning problem is usually modeled as a *Markov game* (Littman, 1994). In this paper, we consider decentralized partially observable Markov decision processes (DEC-POMDP) (Oliehoek & Amato, 2016) with shared rewards, which is an extension of *Markov game* defined by a tuple $\left(\mathcal{N}, \mathcal{S}, \left\{\mathcal{A}^i\right\}_{i\in\mathcal{N}}, \mathcal{P}, \mathcal{R}, \left\{\mathcal{Z}^i\right\}_{i\in\mathcal{N}}, \mathcal{O}, \gamma\right)$, where $\mathcal{N} = \{1, \ldots, N\}$ is the set of agents, $\mathcal{S}$ denotes the state space of the environment, $\mathcal{A}^i$ denotes the action space of agent i. Let $\mathcal{A} := \mathcal{A}^1 \times \cdots \times \mathcal{A}^N$ be the joint action space, then $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the transition probability from any state $s \in \mathcal{S}$ to any state $s' \in \mathcal{S}$ for any joint action $\boldsymbol{a} \in \mathcal{A}$, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function that determines the immediate reward received by the team for a transition from $(s, \boldsymbol{a})$ to s'. The symbol $o^i \in \mathcal{Z}^i$, from observation function $\mathcal{O}(s, i)$, is the local observation for agent i at global state s, and $\gamma \in [0, 1]$ is the discount factor. Each agent has local action-observation history $\tau^i \in T \equiv (\mathcal{Z}^i \times \mathcal{A}^i)^*$, on which it conditions a stochastic policy $\pi^i\left(a^i \mid \tau^i\right)$ that maps each agent's local history to a distribution over it's set of actions. The set of all agents' histories is given by $\boldsymbol{\tau} := \left\{\tau^i\right\}_{i=1}^N$. For better modeling the multi-agent learning problem with designed individual rewards, in this paper, we modify the reward function as $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^{N+1}$, namely a function vector including the each agent's individual reward function and the shared team reward function. At each time step t, when all agents take the joint action $\boldsymbol{a} = \left(a_t^1, \ldots, a_t^N\right)$, the environment returns the reward $\boldsymbol{r} = \left(r_t^1, \ldots, r_t^N, \hat{r}_t\right)$, where r_t^i is the individual reward for agent i and $\hat{r}_t$ is the team reward. The team of cooperative agents attempts to learn a joint policy $\pi\left(\boldsymbol{a}|\boldsymbol{\tau}\right) := \prod_{i=1}^{N} \pi^i\left(a^i|\tau^i\right)$ that maximises their expected discounted team return, $J\left(\pi\right) \doteq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \hat{r}_t\right]$.

MAPPO: Policy gradient techniques (Sutton et al., 1999) aim to estimate the gradient of an agent's expected return with respect to the parameters of its policy. MAPPO (Yu et al., 2021) introduced single-agent PPO (Schulman et al., 2017) into the multi-agent domain under the *Centralized Training and Decentralized Execution* (CTDE) framework. For each agent i, the objective is to maximize

$$J^{CLIP}\left(\theta^i\right) = \mathbb{E}\left[\min\left(\eta_t^i\left(\theta^i\right) A_t^i, clip\left(\eta_t^i\left(\theta^i\right), 1-\epsilon, 1+\epsilon\right) A_t^i\right)\right].$$

where $\eta_t^i\left(\theta^i\right) = \frac{\pi_{\theta^i}\left(a_t^i|\tau_t^i\right)}{\pi_{\theta_{old}^i}\left(a_t^i|\tau_t^i\right)}$ denotes the probability ratio. The function $clip(.)$ removes $\eta_t^i\left(\theta^i\right)$ outside of the interval $[1-\epsilon, 1+\epsilon]$ parameterized by ϵ. A is a generalized advantage estimator (GAE) (Schulman et al., 2016)

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

where $\delta_t^i = r_t^i + \gamma V_{\phi^i}\left(s_{t+1}\right) - V_{\phi^i}\left(s_t\right)$ is the TD error at time step t for agent i and h is the length of trajectory.

3. Related Work

The state-of-the-art MARL algorithms such as QMIX (Rashid et al., 2018), QPLEX (Wang et al., 2021a), MADDPG (Lowe et al., 2017), MAPPO (Yu et al., 2021) work well in many environments, such as MPE and StarCraft II, however, these algorithms still fail to effectively solve the task with sparse team rewards. Reward Shaping (Rahmattalabi et al., 2016) is an approach for solving this problem, which utilizes additional individual rewards[1] that encodes prior knowledge to help the learning of team policy. However, this approach may alter the intended objective of the cooperative task and lead to unexpected behaviors (Randløv & Alstrøm, 1998; Russell & Norvig, 2020; Amodei et al., 2016). Moreover, this class of methods is not generic and needs to be customized according to the task. Although the method based on the potential function can ensure the learning objective unchanged (Ng et al., 1999; Devlin & Kudenko, 2011; Mannion et al., 2018), it cannot guide the algorithms to learn the optimal policy in practice. Obviously, we are more concerned with how to utilize individual rewards to learn team rewards effectively and efficiently.

For multi-agent policy gradient algorithms, many works (Burda et al., 2019; Ye et al., 2020; Li et al., 2021) adopt

[1]Note that the individual rewards also cover the shaping team rewards designed for each of the agents. For simplicity, we treat all shaping rewards as individual ones.

the multi-critic technique as a way of combining the agents' individual and team rewards, which allows each agent to maintain different critics for different rewards and update the policy according to an integration of them to decompose and simplify the learning of the original value function. By treating the maximization of individual rewards and team rewards as two tasks, the methods of Multi-Task Learning (Yu et al., 2020; Zeng et al., 2021; Omidshafiei et al., 2017) can be also adopted as a way of utilizing individual rewards. For example, Yu et al. propose a form of gradient surgery that projects a task's gradient onto the normal plane of the gradient of any other task that has a conflicting gradient, avoiding detrimental gradient interference between task gradients.

Another approach for utilizing individual rewards is Transfer Learning (Liu et al., 2019; Da Silva & Costa, 2019). Simply, the agents' policies can be pre-trained in the source task with the individual rewards, and then fine-tuned in the target task with the sparse team rewards. Multi-agent Evolutionary Reinforcement Learning (MERL) (Majumdar et al., 2020) uses a gradient-based optimizer to train policies for maximizing the dense agent-specific rewards and utilizes an evolutionary algorithm to maximize the sparse team objective through neuroevolution on a population of teams. Although MERL studies the same problem as ours, the high computational and memory cost of the evolution process would make it impractical for real-world applications. In addition, MERL only transfers the skills learned according to the agent-specific rewards to the team population, and does not consider using the team policy to guide the optimization of agent-specific policies.

4. Method

In this paper, we study how to utilize the agents' individual rewards under the policy-based CTDE framework. We propose a novel multi-agent policy gradient algorithm, which is easier to implement and does not involve high computational and space complexity. This section introduces the methodology of this work. We begin with our motivation and then provide algorithm details.

4.1. Motivation

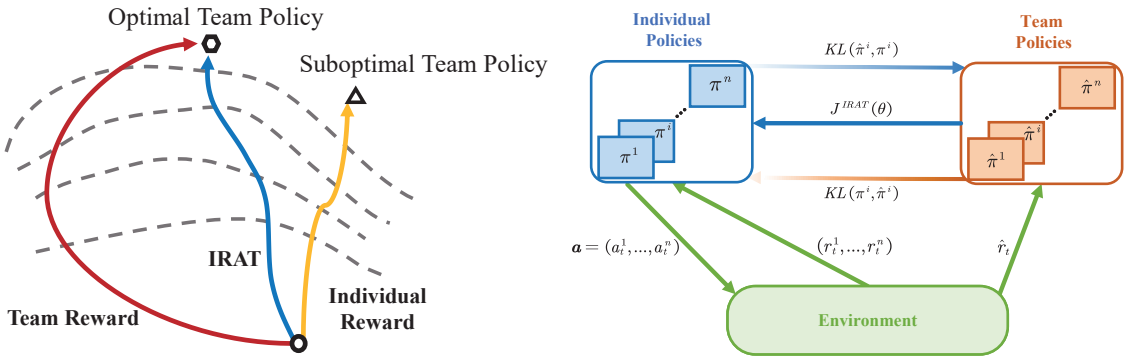
As mentioned in the last section, individual rewards can be utilized to assist the learning of team policy via Reward Shaping, Multi-Critic, Multi-Task Learning and Transfer Learning. Actually, all these approaches make a fusion of the two learning goals (i.e., maximizing the individual and team rewards) to some extent. In our opinion, Reward Shaping, Multi-Critic and Multi-Task Learning make a strong fusion of the two learning goals. However, as the team rewards are much sparser than the individual rewards, such a strong fusion may cause the learning objective to deviate from the goal of team cooperation. In contrast, Transfer learning conducts a weak fusion of the individual and team rewards, since the individual rewards are only used for pre-training. Although Transfer learning ensures that the ultimate goal of the team remains unchanged, due to the sparsity of the team rewards, the agents may quickly forget the pre-trained skills after starting to learn from the team rewards. The above methods all use one policy network to learn two rewards, and the interference between them may not guarantee the final learned policy is aimed at teamwork.

To address these issues, we propose *Individual Reward Assisted Team Policy Learning* (IRAT), which takes the advantages of the two types of fusion approaches, and meanwhile avoids their deficiencies. Firstly, for each agent, IRAT adopts two policies to learn two objectives separately without causing mutual interference between them. The individual policy directly interacts with the environment for sampling, served as a way of exploration and trajectory generation for the team policies and the team policy learns from these trajectories. Secondly, IRAT adds new policy discrepancy constraints to constrain the difference between the two policies so that individual policy explores in the direction of increasing team reward, ensuring their optimization directions are consistent. Figure 1(a) visualizes the idea of the IRAT algorithm. The details of the IRAT algorithm will be given in the following subsections.

4.2. Individual Reward Assisted Team Policy Learning

In IRAT, Each agent i learn a individual policy π^i parameterized by θ^i to maximize the expected discounted accumulated individual reward $J\left(\theta^i\right) \doteq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t^i\right]$, and a team policy $\hat{\pi}^i$ parameterized by $\hat{\theta}^i$ to maximize the expected discounted accumulated team reward $J\left(\hat{\theta}^i\right) \doteq \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \hat{r}_t\right]$. Two policies are learned simultaneously and mutually constrained.

The learning structure of IRAT algorithm is shown in Figure 1(b). The individual policies use our proposed cooperation-oriented objective to adjust their learning behaviors according to the learning situation of the team policies. Moreover, they use an increasing KL regularization term to distill team policy knowledge so that their sampling behaviors can be gradually biased towards regions with higher team reward. Team policies use a importance sampling corrected optimization objective and a decreasing-effect KL regularizer to effectively learn from trajectories sampled by individual policies. In the early stages of learning, for every agent i, the KL regularizer hardly works for individual policy π^i, The KL regularizer of the team policy $\hat{\pi}^i$ promotes $\hat{\pi}^i$ to be closer to the π^i, and the whole algorithm tends to learn individual-reward skills. With the progress of learning, the KL regularizer of π^i begins to play a role, while the KL



(a) The process of learning with different rewards.

(b) IRAT algorithm learning structure.

Figure 1. Team rewards are too sparse to guide the policy to the optimal policy (Red line). Individual rewards are dense but only sub-optimal policies can be learned (Yellow line). IRAT is able to leverage individual rewards for efficient exploration, allows the target policy to move quickly near the optimal team policy in the early learning stage and approach the optimal team policy based on the team reward in the late learning stage (Blue line).

regularizer of $\hat{\pi}^i$ gradually expires. The whole algorithm focuses on the learning of team reward.

4.2.1. Individual-Reward Policy Learning

Team policies learn from trajectories sampled by individual policies and the ultimate learning objective is to maximize the expected accumulative team reward. The individual-reward policy π^i of each agent i needs to adjust its sampling behavior based on the current learning of the team policy for producing samples with higher team reward. When two policies are consistent,, the individual policy π^i should learn quickly, while when the two policies conflict too much, the individual policy π^i should update carefully so as not to deviate too far from the team policy $\hat{\pi}^i$.

To measure the degree of conflict between individual policy and team policy. We define the similarity between π^i and $\hat{\pi}^i$ on $\left(\tau_t^i, a_t^i\right)$

$$\sigma_t^i(\theta^i) = \frac{\pi_\theta^i\left(a_t^i|\tau_t^i\right)}{\hat{\pi}_{\hat{\theta}^i}\left(a_t^i|\tau_t^i\right)}. \tag{1}$$

Based on the policy similarity defined, we propose a new cooperation-oriented objective for π^i:

$$J^{IRAT}\left(\theta^i\right) = \mathbb{E}\left[clip\left(\sigma_t^i(\theta^i), 1-\xi, 1+\xi\right) A_t^i\right], \tag{2}$$

where ξ is a varying coefficient that controls the update range of σ_t^i. When ξ is large, it indicates a weaker constraint on the similarity of the two policies, and when ξ is small, it indicates a stronger constraint.

The basic idea of J^{IRAT} is to limit the update of π^i within the range defined by the policy similarity σ^i. To be concrete, if the individual policy π^i is very different from the team policy $\hat{\pi}^i$ (i.e., σ^i is out of the interval $[1-\xi, 1+\xi]$), which may indicate that the two policies are not optimized along the same or similar directions, the gradients of the policy parameters with respect to the accumulative individual rewards will be clipped to make a small change to π^i.

Now we introduce how to combine the two objectives J^{IRAT} and J^{CLIP} for the optimization of individual policy. Generally, the combination depends on the policy similarity σ^i and the advantage function A^i that indicates the update direction of individual policy.

For a given local trajectory-action pair (τ_t^i, a_t^i), $\sigma_t^i \leq 1$ means that $\hat{\pi}^i$ has a higher probability of choosing a^i than π^i. In this situation, a positive advantage $A_t^i > 0$ indicates that the individual policy π^i and the team policy $\hat{\pi}^i$ tend to be consistent on (τ_t^i, a_t^i), because the resulted gradient will make $\pi^i(a_t^i|\tau_t^i)$ increase and get closer to $\hat{\pi}^i(a_t^i|\tau_t^i)$. To make the approaching process faster, we can choose a learning objective which can provide a steeper policy gradient between J^{IRAT} and J^{CLIP}. When $A_t^i \leq 0$, the corresponding gradient would make $\pi^i(a_t^i|\tau_t^i)$ decrease and lead to more inconsistency between $\pi^i(a_t^i|\tau_t^i)$ and $\hat{\pi}^i(a_t^i|\tau_t^i)$. So it could be better to carefully reduce the probability value of $\pi^i(a_t^i|\tau_t^i)$, which suggests a learning objective whose gradient makes a smaller change to π^i. To summarize, the learning objective of π^i when $\sigma_t^i \leq 1$ is

$$J\left(\theta^i\right) = \mathbb{E}\left[\mathbb{I}_{\sigma_t^i \leq 1} \max\left(J^{CLIP}\left(\theta^i\right), J^{IRAT}\left(\theta^i\right)\right)\right]. \tag{3}$$

Correspondingly, when $\sigma_t^i > 1$, $\hat{\pi}^i$ has a lower probability of choosing a^i than π^i. Therefore, when $A_t^i > 0$, the individual policy π^i will further increase the probability value of $\pi^i(a_t^i|\tau_t^i)$, making the choice of the two policies on (τ_t^i, a_t^i) more inconsistent. So it could be better to carefully

increase the probability value of $\pi^i(a_t^i|\tau_t^i)$, which suggests us to choose a smaller learning objective which can provide a gentler policy gradient between J^{IRAT} and J^{CLIP}. The negative advantage $A_t^i < 0$ in this situation indicates that the individual policy π^i and the team policy $\hat{\pi}^i$ are tend to be consistent on (τ_t^i, a_t^i), because the resulted gradient will make $\pi^i(a_t^i|\tau_t^i)$ has a lower probability of choosing a^i. To make the approaching process faster, a learning objective whose gradient makes a larger change to π^i is suggested. The learning objective of π^i in this situation is

$$J\left(\theta^i\right) = \mathbb{E}\left[\mathbb{I}_{\sigma_t^i>1} \min\left(J^{CLIP}\left(\theta^i\right), J^{IRAT}\left(\theta^i\right)\right)\right]. \tag{4}$$

Since both the individual policy π^i and the team policy $\hat{\pi}^i$ learn from the trajectories sampled by π^i, in order to learn the team policy efficiently, the individual policy should sample trajectories that have higher team reward. Therefore, for the learning objective of the individual reward policy π^i, we also add a regularizer that allows the individual policy π^i to distill the knowledge learned by the team policy $\hat{\pi}^i$ and gradually bias towards regions with higher team rewards. In this paper, we choose Kullback-Leibler (KL) divergence as the regularizer, with an increasing coefficient α.

All things considered, the entire objective of the individual policy π^i is

$$\begin{aligned} J\left(\theta^i\right) =& \mathbb{E}\Big[\mathbb{I}_{\sigma_t^i\leq 1} \max\left(J^{CLIP}\left(\theta^i\right), J^{IRAT}\left(\theta^i\right)\right) \\ &+\mathbb{I}_{\sigma_t^i>1} \min\left(J^{CLIP}\left(\theta^i\right), J^{IRAT}\left(\theta^i\right)\right) \\ &-\alpha KL\left(\hat{\pi}^i, \pi^i\right)\Big]. \end{aligned} \tag{5}$$

4.2.2. Team Policy Learning

The team policy $\hat{\pi}^i$ learns on the trajectories sampled by π^i, Because sampling policies and learning policies are different, it is necessary for $\hat{\pi}^i$ to introduce importance sampling to correct the learning objective. So the update ratio of $\hat{\pi}^i$ becomes

$$\hat{\sigma}_t^i\left(\hat{\theta}^i\right) = \frac{\hat{\pi}_{\hat{\theta}^i}\left(a_t^i|\tau_t^i\right)}{\pi_{\theta_{old}^i}\left(a_t^i|\tau_t^i\right)}, \tag{6}$$

where θ_{old}^i is the parameter of the individual policy at the time of sampling.

Seen from Equation 7, in the early stage of algorithm learning, the individual and team policies may be very different, so the CLIP function will clip off most of the gradients, resulting in the team policy not being updated effectively. To ensure the effective update of $\hat{\pi}^i$, a KL regularizer with an decreasing coefficient β is used to control the distance between the two policies. So for agent i, the objective of $\hat{\pi}^i$ is:

$$\begin{aligned} \hat{J}\left(\hat{\theta}^i\right) =& \mathbb{E}\Big[\min\Big(\hat{\sigma}_t^i(\hat{\theta}^i)\hat{A}_t, clip\left(\hat{\sigma}_t^i(\hat{\theta}^i),\right. \\ &\left.1-\zeta, 1+\zeta\right)\hat{A}_t\Big) - \beta KL\left(\pi^i, \hat{\pi}^i\right)\Big]. \end{aligned} \tag{7}$$

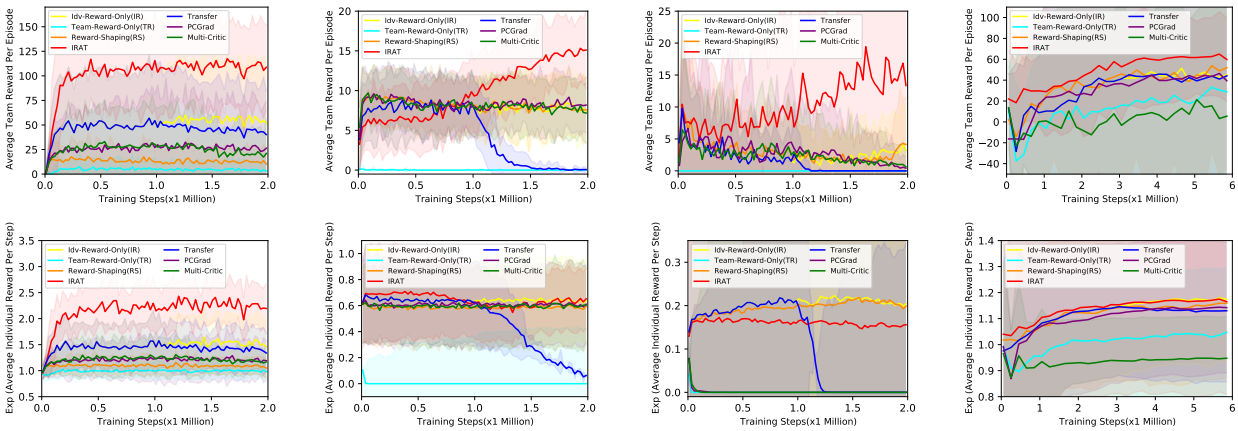
5. Experiments

For complex multi-agent cooperation problems, artificially designed dense individual rewards that help sparse teams-reward learning may not be perfect and always beneficial to the learning of team policy. We design multiple scenarios with different individual reward and team reward relationships based on Multi-Agent Particle Environment (MPE) (Lowe et al., 2017) and prove that our algorithm *Individual Reward Assisted Team Policy Learning* (IRAT), can utilize the parts of individual rewards that are useful for the learning of team team to learn team policy effectively, even when the individual rewards sometimes mislead or conflict with the team reward. Further, we demonstrated the effectiveness of IRAT in the Multiwalker scenario created by SISL (Stanford Intelligent Systems Laboratory) (Gupta et al., 2017) and Google Research Football Environment (Kurach et al., 2020).

To compare with IRAT, this chapter introduces the method utilizing individual reward mentioned in Chapter 3 as the benchmark algorithm for this problem. IR and TR are baseline algorithms that only learn from the individual reward and the team reward respectively. RS is a algorithm using Reward Shaping which adds individual reward and team reward together as the learning reward. Transfer uses individual reward for learning in the first half of the training process, and then changes to learn the team reward. Multi-Critic uses two critic networks to evaluate individual reward and team reward respectively, and then uses the sum of two advantages to guide the policy to update. PCGrad treats the learning of individual reward and team reward as two tasks, and then uses PCGrad (Yu et al., 2020), a multi-task learning approach, to make a policy learn two objectives.

All algorithms are based on the CTDE framework, and each agent uses the actor network (for IRAT, the actor network of the team policy) to make decisions based on local trajectories during execution. The experimental results show the average team reward per episode and the exponential value of the a average individual reward per step for the each algorithm. The experiments measure the performance of each algorithm by the average team reward per episode. The performance of the algorithms on individual and team rewards demonstrates the relationship between individual and team rewards in the task. More details of the experiments can be found in the Appendix B. [2]

[2]Code is available at https://github.com/MDrW/ICML2022-IRAT.



(a) Predator-Prey. (b) Spread. (c) Attack. (d) Multiwalker.

Figure 2. The results of different scenarios in MPE and Multiwalker. The first row shows the result of the average team reward per episode, and the second row shows the result of the exponential value of the average individual reward per step.

5.1. Predator-Prey: Useful Individual Reward

In this environment, 5 slower predators cooperate to capture 2 faster preys. There are 2 obstacles blocking the road in the environment and the episode length is 25. Our algorithm controls predator agents, and preys are controlled by a random policy, in which each prey randomly samples a point in the environment as its action. For each agent, there is a negative reward for the distance to the nearest good agent. If the agent hits any prey, it can get a 5 reward. But only when more than 2 agents hit the same prey, the team can get a 20 team reward. Predator agents do not know this coupling information in advance. They must learn cooperation through sparse team rewards.

The experimental results are shown in Figure 2(a). In this scenario, individual reward encourages the agent to approach and hit prey which is very helpful for team task but lack of cooperation information. It can be seen from the results that the performance of all algorithms on team reward is consistent with their performance on individual reward, i.e., algorithms with higher team reward also have higher individual reward. The individual reward in this scenario is a good decomposition of the team reward for each agent.

In this scenario, TR could hardly learn an effective policy due to the sparsity of team reward. Since individual reward is helpful and dense for team learning, IR is able to learn a sub-optimal policy that works well. RS does not perform well due to the interference between the two reward targets and the cancellation of positive and negative rewards. At the beginning, Transfer performs the same as IR, and then gradually forgets the knowledge learned before and approaches TR. Multi-Critic decomposes the learning of the value network to accurately approximate the advantage function, and PCGrad handles the gradient conflict between the two learning targets. These two methods work better than RS, but still worse than IR. This reflects that the performance of these methods using one policy network to learn two rewards is very affected by the quality of the rewards. If team reward is too sparse to learn from, it can even negatively affect the effectiveness of individual reward that can lead to a sub-optimal policy. IRAT learns to approach the preys through individual policy to increase the probability of obtaining team reward and further improve the success rate of teamwork by gradually transferring team knowledge to increase team awareness. As a result, IRAT outperforms other methods.

5.2. Spread: Misleading Individual Reward

There are 4 agents and 2 landmarks in this environment and agents learn to cooperation to find all landmarks. For each agent, Individual reward is the minimum distance to all undetected landmark. Only if more than 2 agents detect the landmark at the same time, the landmark will be discovered, and all agents will be rewarded by a positive team reward correlated with the number of landmarks found at this time. The reward for one landmark is 10. As in the above experiment, agents do not know cooperation information and can only learn cooperation through team rewards.

By learning individual reward, agents learn to get close to undiscovered landmarks which seems to be helpful for the learning of the team reward. However this individual reward will mislead the agents. When a landmark is found, the agent will suddenly receive a negative reward from the new undiscovered landmark, which makes the agents stay where they are and get into a stalemate, thus missing out on the

exploration of other team rewards. As you can see from the Figure 2(b), IRAT obtains fewer individual reward in the later stages of learning, indicating that the agents overcome the trap brought by individual rewards through cooperation. TR and Transfer cannot learn a useful policy while other methods can only find one landmark and get 10 team reward in a episode with the length of 25. The individual policy of IRAT will adjust its sampling behavior according to the team policy, which will make the algorithm slow to reach the 10 team reward, but can break through the misleading of individual reward.

5.3. Attack: Conflicting Individual Reward

In this scenario, There is one landmark with a size of 0.02 and 3 agents with a size of 0.1. The team target is that the three agents reach the landmark and attack at the same time without collision. If they complete the attack, the environment returns a team reward of 20 and ends this episode. In order to assist the learning, for each agent, a distance penalty to landmark is added, and a -1 reward is added to avoid collisions. However, this widely used design of individual rewards will have a negative impact on team learning. In this scenario, individual rewards guide agents to get close to landmark and not to collide with each other, but since the size of landmark is smaller than agents, collisions are bound to occur when three agents attack the landmark at the same time. Therefore, this reward is beneficial to learning at the beginning of learning, but it will hinder the learning of team rewards in the later stage.

As you can see from Figure 2(c), TR and Transfer cannot learn a useful policy. For IR, there is a conflict between individual reward and team reward, and optimizing individual rewards will necessarily prevent the agents from completing their attacks. Therefore, the probability that IR successfully samples non-zero team rewards decreases in the later stages. Multi-Critic and PCGrad perform similarly to IR in the early stages of learning, but their performance gradually decreases as the conflict between individual and team rewards increases in the later stages of learning. For the IRAT method, its individual policy distills the knowledge of the team policy later in the learning process, making the two policies very close to each other, thus dissolving the conflict between the individual and team rewards. As can be seen from the figure, the IRAT method obtains much higher team reward than any other benchmark algorithm. At the same time, the IRAT method has a lower individual reward than IR because the successful team attacks lead to more collisions between the agents and generate more negative rewards.

5.4. Multiwalker

In Multiwalker, A package is placed on top of 2 bipedal robots which our algorithm controls. Bipedal robots attempt to carry a package as far right as possible. The team reward is the change in the package distance summed with 130 times the change in the walker's position and the walkers are given a team reward of -50 when they fail the game by either condition. For each agent, it has a negative individual reward related to the standing, a positive reward that encourages the agent to move to the right and a -5 reward if it falls. The team reward for this scenario is not sparse, and the team reward can lead to a certain but not good policy due to the interference of environmental noise.

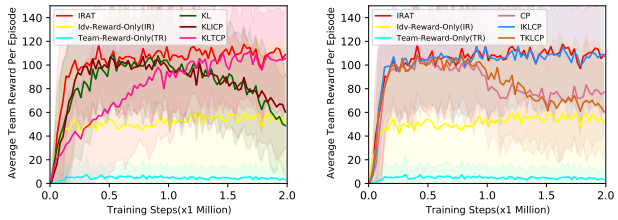
The control of the robot is very complex. Although the team reward is not sparse, due to the large exploration space, the agent can only converge to a local joint policy solution. As can be seen in Figure 2(d). TR can learn certain policy and IR performs better. Due to the mixed effect between positive and negative rewards, it is difficult for other methods to outperform the optimal performance of IR and TR. IRAT can utilize the knowledge contained in individual rewards to get higher team reward.

5.5. Ablation

In this section, we conducted an ablation experiment on Predator-Prey to prove the role of cooperation-oriented clip term ICP of individual policy, update clip term TCP modified by importance sampling of team policy, the KL regularization term IKL of individual policy and the KL regularization term TKL of team policy. To prove the role of the ICP and TCP, we implement the following variants. KL is a variant of our algorithm that uses only two KL regularizers. KLICP is a variant that only use ICP and two KL regularization terms while KLTCP is that only use TCP and two KL regularizers.

As seen in Figure 3(a), the comparison between KL and KLICP, IRAT and KLTCP shows that when ICP is used, the individual policy will adjust its learning behavior according to the performance of the team policy and can explore the team reward faster. In contrast, when the individual policy does not use ICP, the individual policy samples fewer valid team reward samples, leading to slow learning of team reward. Comparing KL and KLTCP, IRAT and KLICP, it can be observed that TCP enables a stable improvement in the performance of team policy.

In order to analyze the impact of KL term, we implemented and compared the following variant algorithms. CP is a variant that removes two KL regularizers and uses ICP and TCP. IKLCP is a variant removing TKL while TKLCP is a variant removing IKL. The experimental results are shown in Figure 3(b). ICP provides cooperation-oriented guid-



(a) Clip term ablation. (b) KL term ablation.

Figure 3. Ablation Experiment in Predator-Prey.

ance for individual policy updates, which enables individual policy to adjust the update magnitude according to policy similarity, but individual policy is still updated based on its own advantage function, that is, it is still learning individual reward. Therefore, as learning proceeds, the similarity between the two policies becomes weaker, the performance of individual policy sampling team reward gradually returns to IR, and the team policy effective updating decreases accordingly. Comparing CP and IKLCP, IRAT and TKLCP, it can be seen from their performance that IKL allows the individual policy to gradually distill the knowledge of the team policy, keeping the sense of cooperation all the time, so that when IKL is used, the algorithm gradually focuses on learning the team reward and obtaining higher team reward. TKL controls the distance between the two policies in the pre-learning period to ensure effective updating of the team policy. When the two rewards are consistent and the policies do not differ much, the effect of TKL is not obvious, but it does not harm the performance.

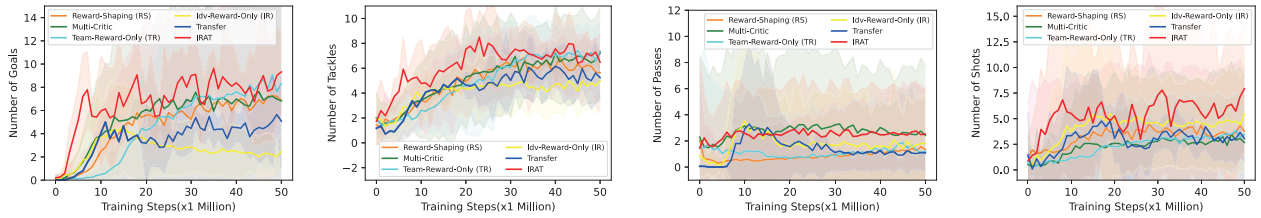
5.6. Google Research Football

We also conduct experiments in the very challenging Google Research Football Environment (GRF) to verify the performance of IRAT in complex scenarios. Specifically, we test IRAT and 5 baseline methods in the 5-vs-5 half-court offense GRF task. Each tested method should learn to control the players of the left team (dressed in yellow) except the goalie to play football match against the right team (dressed in blue). The goalie and all players of the right team are controlled by the rule-based program provided by GRF. The playing area of the game is restricted within the frontcourt of the left team. In the beginning of each episode, the positions of all non-goalie players are randomly set, and the ball is randomly assigned to one player of the left team. An episode will end when the left team scores a goal, or the ball gets into the backcourt of the left team, or the maximal time step 3000 is reached. All of the tested methods adopt MAPPO as the base learning algorithm.

The team reward for the left team is 1 only when the team goals and is 0 in other cases. We design four types of individual rewards to help with learning, including the position rewards, shooting rewards, ball-passing rewards, and ball-possession rewards. The position rewards are punishments for running out of the field or moving beyond the offside line. The shooting rewards encourage the shots in the area with high goal-scoring chance and punish very long or meaningless shots. The ball-passing rewards depends on how the ball passes between players are performed. While successful ball passes are positively rewarded, the passes leading to the loss of ball possession, worse offense status, and offside are punished. The ball-possession rewards are individual rewards with respect to the switch of the ball possession. The reward details are given in Appendix.

Each method is trained for 50 million steps and is evaluated every $500,000$ steps. The evaluation process of a method contains 50 games and each game lasts for 3000 steps. We record the number of goals scored by each method in each game and get the average goal scores in the 50 games. All tests of the algorithms are conducted with 4 random seeds and the results are given in Figure 4. It can be found that IRAT significantly outperforms the other methods, with higher goal scores and much faster convergence. While IRAT achieves a goal-score value of 6 within 5 million steps, the other methods would take at least 20 million steps to reach such performance. As for the asymptotic performance, both IRAT and the Team-Reward-Only (TR) method finally achieve the highest average goal scores, but the latter converges much slower and reaches the performance level of IRAT almost at the end of the training process.

It is also interesting to investigate the learning process of the other methods. During the early training phase, the Individual-Reward-Only (IR) method has a much higher goal-score value than the TR method. This is because the four types of individual rewards can guide the agents take behaviors beneficial to score a goal. For example, the ball-possession rewards require the agents to keep the ball possession when they control the ball and try to get the possession when the ball is owned by the opponent team. The ball-passing and shooting rewards together can guide the agents pass the ball to the area with high scoring chance and make a shot. However, since the individual rewards are not completely consistent with the team rewards, the goal-score value of IR decreases after about 15 million steps, which indicates that IR finds a better way to get higher individual rewards. The Transfer method uses the individual rewards to train the agents' policies in the first 20 million steps, and switches to use the team rewards to train thereafter. Therefore, the learning curves of the Transfer and IR methods almost overlap during the initial phase, and after switching the learning objective, the average goal scores of the Transfer method begin to increase. Like our IRAT method, the Reward-Shaping (RS) and Multi-Critic (MC) methods also combine the team and individual rewards.



(a) Number of Goals (b) Number of Tackles (c) Number of Passes (d) Number of Shots

Figure 4. The results of the Google Research Football experiment

With the individual rewards, both the methods can quickly raise their goal scores in the early stage, and based on the team rewards, their score values keep increasing as training proceeds. However, due to the conflict between the individual and team rewards, the asymptotic performance of the two methods fail to reach the performance level of the IRAT and TR methods.

Besides the indicator of goal scores, we also examine what behaviors have been learnt by the agents with the tested methods. Specifically, in each evaluation game of the methods, we record the number of tackles, ball passes, and shots performed by the left team, which directly indicate how the learning methods deal with the individual rewards. The results are also shown in Figure 4. Compared with the baseline methods, the agents controlled by IRAT learn better skills of tackling and shooting, as the numbers of tackles and shots performed by the IRAT agents are always the largest during training. Surprisingly, Figure 4(c) shows that none of the test methods learn a ball-pass-preferring policy. As we have observed in the gameplay videos after training, the offense tactics learnt by these methods are all to let the agents dribble the ball into the box and take a shot, which is a simple but effective tactic for beating the rule-based bot. Obviously, learning such a tactic is much easier than learning ball-passing tactics.

6. Conclusion

In this paper, we investigate how individual rewards can be used to assist multi-agent reinforcement learning in cooperative tasks with sparse team rewards. We introduce IRAT, a novel multi-agent policy gradient algorithm which provides a new way of utilizing individual rewards to promote cooperation. The basic idea of IRAT is to learn an individual policy and a team policy for each agent, update the two policies in their common trust regions, and meanwhile put discrepancy constraints on them to distill knowledge from each other. Experiments in various scenarios such as Multi-Agent Particle Environment and Google Research Football Environment demonstrate that IRAT can greatly promote team policy learning with individual rewards and significantly outperforms the state-of-the-art methods.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments and helpful suggestions. This work is supported by Science and Technology Innovation 2030 New Generation Artificial Intelligence Major Project(No.: 2018AAA0100904, 2018AAA0100905), the National Natural Science Foundation of China(No.: 62192783, 62176135, U1836214), Primary Research and Developement Plan of Jiangsu Province (No. BE2021028), the new Generation of Artificial Intelligence Science and Technology Major Project of Tianjin under grant (19ZXZNGX00010).

References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P. F., Schulman, J., and Mané, D. Concrete problems in AI safety. *CoRR*, abs/1606.06565, 2016.

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by random network distillation. In *Seventh International Conference on Learning Representations*, pp. 1–17, 2019.

Da Silva, F. L. and Costa, A. H. R. A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research*, 64:645–703, 2019.

Devlin, S. and Kudenko, D. Theoretical considerations of potential-based reward shaping for multi-agent systems. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*, pp. 225–232. ACM, 2011.

Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, pp. 2974–2982. AAAI Press, 2018.

Gupta, J. K., Egorov, M., and Kochenderfer, M. J. Cooperative multi-agent control using deep reinforcement learning. In *Autonomous Agents and Multiagent Systems - AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers*, volume 10642 of *Lecture Notes in Computer Science*, pp. 66–83. Springer, 2017.

Iqbal, S. and Sha, F. Actor-attention-critic for multi-agent reinforcement learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2961–2970. PMLR, 2019.

Jin, J. and Ma, X. A multi-objective agent-based control approach with application in intelligent traffic signal system. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3900–3912, 2019.

Kurach, K., Raichuk, A., Stańczyk, P., Zając, M., Bachem, O., Espeholt, L., Riquelme, C., Vincent, D., Michalski, M., Bousquet, O., et al. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4501–4510, 2020.

Li, Q., Zhou, W., Zhou, Y., and Li, H. Attentive update of multi-critic for deep reinforcement learning. In *2021 IEEE International Conference on Multimedia and Expo, ICME 2021, Shenzhen, China, July 5-9, 2021*, pp. 1–6. IEEE, 2021.

Littman, M. L. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Liu, Y., Hu, Y., Gao, Y., Chen, Y., and Fan, C. Value function transfer for deep multi-agent reinforcement learning based on n-step returns. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 457–463, 2019.

Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Majumdar, S., Khadka, S., Miret, S., Mcaleer, S., and Tumer, K. Evolutionary reinforcement learning for sample-efficient multiagent coordination. In *International Conference on Machine Learning*, pp. 6651–6660. PMLR, 2020.

Mannion, P., Devlin, S., Duggan, J., and Howley, E. Reward shaping for knowledge-based multi-objective multi-agent reinforcement learning. *The Knowledge Engineering Review*, 33, 2018.

Ng, A. Y., Harada, D., and Russell, S. J. Policy invariance under reward transformations: Theory and application to reward shaping. In Bratko, I. and Dzeroski, S. (eds.), *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27 - 30, 1999*, pp. 278–287. Morgan Kaufmann, 1999.

Oliehoek, F. A. and Amato, C. *A Concise Introduction to Decentralized POMDPs*. Springer Briefs in Intelligent Systems. Springer, 2016.

Omidshafiei, S., Pazis, J., Amato, C., How, J. P., and Vian, J. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2681–2690. PMLR, 2017.

Rahmattalabi, A., Chung, J. J., Colby, M., and Tumer, K. D++: Structural credit assignment in tightly coupled multiagent domains. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4424–4429. IEEE, 2016.

Randløv, J. and Alstrøm, P. Learning to drive a bicycle using reinforcement learning and shaping. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 463–471, 1998.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4295–4304. PMLR, 2018.

Russell, S. J. and Norvig, P. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson, 2020.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shamsoshoara, A., Khaledi, M., Afghah, F., Razi, A., and Ashdown, J. Distributed cooperative spectrum sharing in uav networks using multi-agent reinforcement learning. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 1–6. IEEE, 2019.

Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 5887–5896. PMLR, 2019.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V. F., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., and Graepel, T. Value-decomposition networks for cooperative multi-agent learning based on team reward. In André, E., Koenig, S., Dastani, M., and Sukthankar, G. (eds.), *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M. M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. QPLEX: duplex dueling multi-agent q-learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net, 2021a.

Wang, Y., Han, B., Wang, T., Dong, H., and Zhang, C. DOP: off-policy multi-agent decomposed policy gradients. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021b.

Ye, D., Chen, G., Zhang, W., Chen, S., Yuan, B., Liu, B., Chen, J., Liu, Z., Qiu, F., Yu, H., et al. Towards playing full moba games with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 621–632, 2020.

Yu, C., Velu, A., Vinitsky, E., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33: 5824–5836, 2020.

Zeng, S., Anwar, M. A., Doan, T. T., Raychowdhury, A., and Romberg, J. A decentralized policy gradient approach to multi-task reinforcement learning. In *Uncertainty in Artificial Intelligence*, pp. 1002–1012. PMLR, 2021.

A. Pseudo-Code of IRAT

The pseudo-code of the IRAT algorithm is shown in Algorithm 1.

Algorithm 1 Individual Reward Assisted Team Policy Learning (IRAT)

Initialize the parameters θ^i of individual policy π^i for each agent, the parameters ϕ^i of individual critic V^i for each agent, the parameters $\hat{\theta}^i$ of team policy $\hat{\pi}^i$ for each agent and the parameters $\hat{\phi}^i$ of team critic $\hat{V}^i$ for each agent, using Orthogonal initialization
Set learning rate α
while $step \leq step_{\text{max}}$ **do**
  set data buffer $D = \{\}$
  for $b = 1$ **to** $batch\_size$ **do**
    $\mathcal{T} =$ [] empty list
    for $t = 1$ **to** T **do**
      for all agents i **do**
        $p_t^i = \pi^i(\tau_t^i; \theta^i)$
        $a_t^i \sim p_t^i$
        $v_t^i = V^i(s_t; \phi^i)$
        $\hat{p}_t^i = \hat{\pi}^i(\tau_t^i; \hat{\theta}^i)$
        $\hat{v}_t^i = \hat{V}^i(s_t; \hat{\phi}^i)$
      end for
      Execute actions $\boldsymbol{a} \equiv (a_t^1, ..., a_t^n)$, observe $r_t^1, ..., r_t^n, \hat{r}_t, s_{t+1}, o_{t+1}^1, ..., o_{t+1}^n$
      $\mathcal{T} += [s_t, o_t^1, ..., o_t^n, \boldsymbol{a}, p_t^1, ..., p_t^n, \hat{p}_t^1, ..., \hat{p}_t^n, r_t^1, ..., r_t^n, \hat{r}_t, o_{t+1}^1, ..., o_{t+1}^n]$
    end for
    Compute advantage estimate A_t^i for individual policy via GAE on $\mathcal{T}$, using PopArt
    Compute advantage estimate $\hat{A}_t$ for team policy via GAE on $\mathcal{T}$, using PopArt
    $D = D \cup \mathcal{T}$
  end for
  for $p = 1$ **to** $train\_epoch$ **do**
    Sample data d from D
    Adam update θ^i on $J(\theta^i)$ with data d for each agent
    Adam update ϕ^i on $L(\phi^i)$ with data d for each agent
    Adam update $\hat{\theta}^i$ on $\hat{J}(\hat{\theta}^i)$ with data d for each agent
    Adam update $\hat{\phi}^i$ on $\hat{L}(\hat{\phi}^i)$ with data d for each agent
  end for
  for $e = 1$ **to** $eval\_episodes$ **do**
    Evaluate team reward policy $\hat{\pi}^i$ of each agent
  end for
end while

The individual critic network for agent i is trained to minimize the loss function

$$L(\phi^i) = \frac{1}{B}\sum_{k=1}^{B} \max\left[(V_{\phi^i}^i(s_k) - R_k^i)^2, (\text{clip}(V_{\phi^i}^i(s_k), V_{\phi_{old}^i}^i(s_k) - \epsilon, V_{\phi_{old}^i}^i(s_k) + \epsilon) - R_k^i)^2\right]$$

The team critic network for agent i is trained to minimize the loss function

$$\hat{L}(\hat{\phi}^i) = \frac{1}{B}\sum_{k=1}^{B} \max\left[(\hat{V}_{\hat{\phi}^i}^i(s_k) - \hat{R}_k)^2, (\text{clip}(\hat{V}_{\hat{\phi}^i}^i(s_k), \hat{V}_{\hat{\phi}_{old}^i}^i(s_k) - \hat{\epsilon}, \hat{V}_{\hat{\phi}_{old}^i}^i(s_k) + \hat{\epsilon}) - \hat{R}_k)^2\right]$$

where B refers to the batch size, R_k^i is the discounted reward-to-go of agent i's individual reward and $\hat{R}_k$ is the discounted reward-to-go of team reward.

Table B.1. Common hyperparameters used in all algorithms in MPE and Multiwalker.

Common hyperparameters	Value
num GRU layers	1
RNN hidden state dim	64
fc layer dim	64
num fc	2
num fc after	1
recurrent data chunk length	10
gradient clip norm	10.0
gae lamda	0.95
gamma	0.99
value loss	huber loss
huber delta	10.0
batch size	num envs × buffer length × num agents
mini batch size	batch size / mini-batch
optimizer	Adam
optimizer epsilon	1E-5
weight decay	0
network initialization	Orthogonal
use reward normalization	True
use feature normalization	True
activation	ReLU
mini-batch	1
epoch	10
gain	0.01
clip	0.2
num envs	8

B. Implementation Details

B.1. MPE and Multiwalker Experiment

The common parameters of all algorithms in different scenarios are shown in Table B.1, and the parameters of IRAT are shown in Table B.2.

B.2. Google Research Football Experiment

Problem Setting: In the football experiment, the playing field of the game is restricted within the frontcourt of the left team, which is also the backcourt of the right team. According to the internal coordinate representation, the coordinates of the four corners of the playing field are $(0, -0.42)$, $(0, 0.42)$, $(1.0, -0.42)$, and $(1.0, 0.42)$. The 4 non-goalie players of the left team are randomly placed onto the field at the beginning of an episode, with their x-coordinates and y-coordinates uniformly sampled from $[0.51, 0.65]$ and $[-0.42, 0.42]$, respectively. The initial positions of the non-goalie players of the right team are also randomly initialized, with the x-coordinates and y-coordinates uniformly sampled from $[0.66, 0.8]$ and $[-0.24, 0.24]$, respectively. In each episode, the ball is randomly assigned to one of the 4 non-goalie players of the left team initially. The roles of the four players are *center forward* (CF), *right midfield* (RM), *left back* (LB), and *right back* (RB) in both the two teams. The difficulty level of the right team is 0.05. Although the highest difficulty level is 1.0, beating a 0.05 difficulty-level opponent team is not that easy for the tested methods.

Reward Setting: As mentioned in the paper, the team reward is 1 only when the yellow team scores a goal and is 0 in other cases. The setting of the individual rewards is much more complicated. We summarize the specific rules for setting the position reward r_{xy}, shooting reward r_{sh}, ball-passing reward r_{pass}, and ball-possession reward r_{poss} in Table B.3.

Observation Representation: As the 5-vs-5 football task is extremely difficult, we have designed a very comprehensive representation of the agents' observations in our experiment. Specifically, we extract 14 groups of information from a single agent's perspective and generate a 14-vector representation of the agent's observation, which is adopted as the input of the agent's policy network. The input of each agent's critic networks is the concatenation of the first layers of all agents' policy networks. The details of the 14 groups of observation information are listed in Table B.4.

Table B.2. Hyperparameters of IRAT used in different scenarios of MPE and Multiwalker.

Scenarios	Hyperparameters of IRAT
Predator-Prey	The Policy Clipping Ratio ξ in J^{IRAT}: $3.0 \rightarrow 0.5$, decaying in 2 million steps The Policy Clipping Ratio ζ in $\hat{J}$: 0.2 KL Loss Coefficient in J^{IRAT}: $0 \rightarrow 1.0$, growing in 2 million steps KL Loss Coefficient in $\hat{J}$: $1.0 \rightarrow 0$, decaying in 2 million steps
Spread	The Policy Clipping Ratio ξ in J^{IRAT}: $3.0 \rightarrow 0.5$, decaying in 2 million steps The Policy Clipping Ratio ζ in $\hat{J}$: 0.2 KL Loss Coefficient in J^{IRAT}: $0.2 \rightarrow 2.0$, growing in 2 million steps KL Loss Coefficient in $\hat{J}$: $1.5 \rightarrow 0$, decaying in 1.6 million steps
Attack	The Policy Clipping Ratio ξ in J^{IRAT}: $3.0 \rightarrow 0.5$, decaying in 2 million steps The Policy Clipping Ratio ζ in $\hat{J}$: 0.2 KL Loss Coefficient in J^{IRAT}: $0 \rightarrow 1.0$, growing in 2 million steps KL Loss Coefficient in $\hat{J}$: $1.0 \rightarrow 0$, decaying in 2 million steps
Multiwalker	The Policy Clipping Ratio ξ in J^{IRAT}: $0.2 \rightarrow 0.5$, growing in 6 million steps The Policy Clipping Ratio ζ in $\hat{J}$: 0.2 KL Loss Coefficient in J^{IRAT}: $0 \rightarrow 1.0$, growing in 6 million steps KL Loss Coefficient in $\hat{J}$: $1.5 \rightarrow 0.5$, decaying in 6 million steps

Hyperparameters: The hyperparameters chosen for the tested methods in the football experiment are listed in Table B.5.

C. Additional Experiments

C.1. Positive Individual Reward in Predator-Prey

To verify whether individual rewards with the same guiding meaning but different positive and negative values have an effect on the effectiveness of algorithms, we designed a Predator-Prey scenario with totally positive individual rewards. The scenario uses 1.0/(0.05 + nearest distance to prey) instead of the negative distance penalty of the previous experiment to guide the predator closer to the prey. The results of the experiment are shown in Figure 5(a).

Comparing Figures 5(a) and Figures 2(a), it can be seen that negative individual reward cancels out with positive team

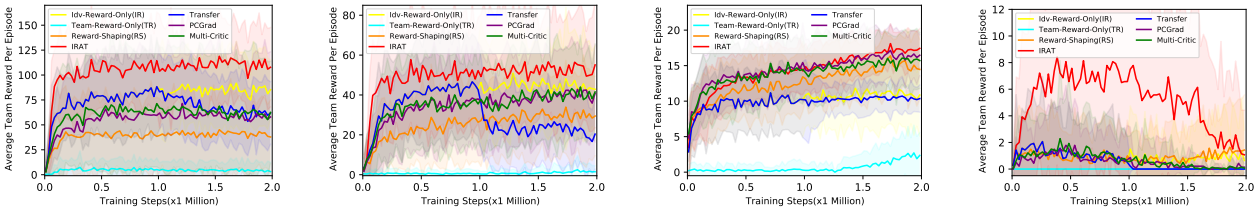
Table B.3. The individual reward setting adopted in the football experiment

Reward Type	Rules
Position Reward	$r_{xy} = -0.02$ if a player runs out of the playing field $r_{xy} = -0.04$ if a player goes beyond the offside line $r_{xy} = -0.01$ if a player is still in an offside position
Shooting Reward	$r_{sh} = 0.16$ if a player shoots when its x-coordinate is larger than 0.66 and its y-coordinate is in $[-0.25, 0.25]$ $r_{sh} = -0.08$ if a player shoots when its x-coordinate is smaller than 0.3
Ball-Passing Reward	$r_{pass} = -0.2$ if a player performs an offside pass $r_{pass} = 0.4$ if the ball is successfully passed from one player to another $r_{pass} = -0.4$ if the ball get lost after a player performs a pass
Ball-Possession Reward	$r_{poss} = -0.3$ if a player takes the ball into the backcourt of the left team $r_{poss} = -0.03$ if the ball player loses the ball possession $r_{poss} = 0.03$ if a player gets the ball possession back

Table B.4. The observation features designed for the tested methods in the football experiment

Observation Info	Description
Ball	ball position, ball direction, ball rotation
Ball Possession	one-hot id of the ball-owned team, one-hot id of the player having the ball
Player Info	the player's one-hot id, position, moving direction, tired factor, and yellow-card indicator
Player-Ball Info	the player's relative position w.r.t. the ball, the distance between the player and the ball, and the reciprocal of the distance
Player-Ball Player Info	the player's relative position w.r.t. the ball player, the distance between the player and the ball player, and the reciprocal of the distance
Player-Teammates Info	the player's relative positions w.r.t. the teammates, the distance between the player and the teammates, the reciprocals of the distances, cosine values of the angles between the player's and the teammates' moving directions, cosine values of the angles between the player's moving direction and the teammates' position vectors
Player-Opponents Info	the player's relative positions w.r.t. the opponents, the distance between the player and the opponents, the reciprocals of the distances, cosine values of the angles between the player's and the opponents' moving directions, cosine values of the angles between the player's moving direction and the opponents' position vectors
Left Team Info	the positions, moving directions, tired factors, offside and yellow-card indicators of all players in the left team
Right Team Info	the positions, moving directions, tired factors, offside and yellow-card indicators of all players in the right team
Special Player Info	the relative positions, distances, distance reciprocals, cosine values w.r.t. two nearest teammates, two nearest opponents, and the goalie of the right team
Game Mode	one-hot id of the game mode (e.g., normal, penalty, corner, free kick)
Action Mask	a vector indicating which actions of the player are legal currently
Action Sequence	one-hot ids of the actions taken by the player in the last 4 steps
Team Formation	the mean and variance of the positions of the left-team players, the mean and variance of the positions of the right-team players

rewards, which can interfere with learning. With positive individual reward, IR, RS, Transfer, Multi-Critic and PCGrad all achieve higher team reward. IRAT utilizes individual policies and their sampled trajectories, a high-level knowledge, rather than raw reward values, that is free from such cancellation. Thus IRAT can perform equally excellent performance in both scenarios.



(a) Positive Individual Reward in Predator-Prey. (b) Predator-Prey. (c) Spread. (d) Attack.

Figure 5. The additional results of different scenarios in MPE.

C.2. MPE with Discrete Action Space

We also append the experiment results of the algorithms in three scenarios in MPE with discrete action space, and it can be seen in Figure 5(b), 5(c), 5(d) that IRAT still outperforms the other algorithms in discrete action space.

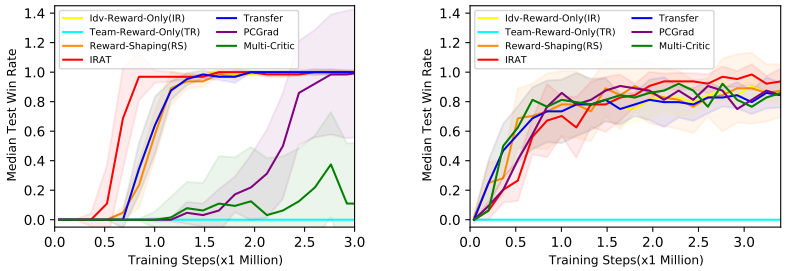
C.3. SMAC Experiment

Further, we verify the performance of the IRAT algorithm in two maps, 3s_vs_5z and 10m_vs_11m, of the StarCraft II Multi-Agent Challenge Environment (SMAC). We directly adopt the default dense rewards of SMAC (which are commonly used by most multi-agent algorithms such as MAPPO) as the agents' individual rewards. The team reward is set 20 when

Table B.5. The hyperparameters of the tested methods in the *Google Research Football* experiment

Method	Hyperparameters
Base Learner (MAPPO)	Policy Network: The *first layer* is the concatenation of 14 private FC layers, which have 16, 16, 16, 16, 16, 32, 32, 32, 32, 32, 16, 32, 64, and 32 units, respectively. Each of the private layers is fully connected to the corresponding feature vector input. The *second layer* is a 128-unit FC layer. The *third layer* is a 96-unit FC layer. *Relu* activation Value Network: The *first layer* is a 128-unit FC layer. The *second layer* is a 96-unit FC layer. *Relu* activation The Policy Clipping Ratio ϵ: 0.2 Update Period: $1,024$ steps Number of Epochs Per Update: 8 Batch Size: 256 GAE Parameter (λ): 0.97 Optimizer: Adam ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon_{adam} = 10^{-5}$) Learning Rate: 0.00019896 Value Loss Coefficient: 0.25 Entropy Term Coefficient: 0.001 Gradient Clip Norm: 40.0 Discount Rate (γ): 0.993 Reward Scaling Factor: 5.0
IRAT	The Policy Clipping Ratio ξ in J^{IRAT}: $0.5 \rightarrow 0.2$, decaying in 10 million steps The Policy Clipping Ratio ζ in $\hat{J}$: 0.2 KL Loss Coefficient in J^{IRAT}: $0 \rightarrow 0.005$, growing in 10 million steps KL Loss Coefficient in $\hat{J}$: $0.005 \rightarrow 0$, decaying in 10 million steps

the agents' team wins the game. The experimental results are shown in Figure 6. In 3s_vs_5z, IRAT is able to reach the 100 percent win rate faster while in 10m_vs_11m, IRAT eventually converges to higher win rate.



(a) 3s_vs_5z. (b) 10m_vs_11m.

Figure 6. The results of StarCraft II.