Neural Computing and Applications (2024) 36:7203–7219
https://doi.org/10.1007/s00521-024-09455-x

ORIGINAL ARTICLE

From mimic to counteract: a two-stage reinforcement learning algorithm for Google research football

Junjie Zhao[1] · Jiangwen Lin[1] · Xinyan Zhang[1] · Yuanbai Li[1] · Xianzhong Zhou[1,2] · Yuxiang Sun[1,2]

Received: 11 May 2023 / Accepted: 14 January 2024 / Published online: 22 February 2024

Abstract

Deep reinforcement learning has proven to be effective in various video games, such as Atari games, StarCraft II, Google research football (GRF), and Dota II. We participated in the 2022 IEEE Conference on Games Football AI Competition and ranked in the top eight. Despite recent efforts, building agents for GRF still suffers from multi-agent coordination, sparse rewards, and stochastic environments. To address these issues and achieve good outcomes in the competition, we devised a reinforcement learning algorithm that uses deep reinforcement learning from demonstrations and policy distillation. In this study, we innovatively propose a two-stage algorithm named mimic-to-counteract reinforcement learning (MCRL) based on the historical game logs of opponents, we encountered during the warm-up session and formulated partner agents function similarly to human sparring partners, whereby they simulate opponents with diverse styles of play, enabling primary players to practice against a range of policies, they may encounter in real competitions. Additionally, we trained numerous mentor agents capable of restraining the sparring partners. We distilled their policies and amalgamated them to train a potent primary agent. Empirical results show that the proposed MCRL algorithm can efficiently search for valuable strategies with stable updates and balance the relationship between policy iteration and policy style deviation. Also, the primary agent can learn diverse but coordinated counteracting strategies and ranks in the top eight in the competition.

Keywords Multi-agent reinforcement learning · Google research football · Wasserstein distance · Policy distillation

1 Introduction

Deep reinforcement learning (DRL) has proven to be effective in various video games [1], such as Atari games [2], StarCraft II [3], Google research football (GRF) [4], and Dota II [5]. However, DRL systems still face challenges such as multi-agent coordination [6–8], sparse rewards [9, 10], and stochastic environments. We participated in the 2022 IEEE Conference on Games Football AI Competition and secured a fourth place ranking in the warm-up session and advances to the top eight in the IEEE Conference on Games 2022 Football AI Competition.

GRF is a reinforcement learning platform designed to provide a realistic football game environment for researchers and developers. The platform offers a new reinforcement learning environment where agents are trained to play football in an advanced, physics-based 3D simulator [4]. GRF provides a highly customizable game engine that allows users to modify various game rules and

✉ Xianzhong Zhou
zhouxz@nju.edu.cn

✉ Yuxiang Sun
sunyuxiang@nju.edu.cn

Junjie Zhao
junjiezhao@smail.nju.edu.cn

Jiangwen Lin
522022150065@smail.nju.edu.cn

Xinyan Zhang
522022150085@smail.nju.edu.cn

Yuanbai Li
mf21150056@smail.nju.edu.cn

[1] School of Management and Engineering, Nanjing University, Nanjing 210093, People's Republic of China

[2] Research Center for New Technology in Intelligent Equipment, Nanjing University, Nanjing 210093, People's Republic of China

parameters, such as the number of agents controlled and the number of agents controlled. The platform features challenging AI opponents to compete against users and provides various learning algorithms and benchmark tests to evaluate the performance of different algorithms. Table 1 contrasts GRF with some other popoular DRL environments, illustrating its challenge. As shown in Fig. 1, three types of observations are offered. The first type consists of 1280 × 720 RGB images that correspond to the displayed screen. The second type is the super mini-map (SMM), which consists of four 72 × 96 matrices to record current situational data. The third sort of observation is the RAW observations, which encompass a dict that encapsulates the up-to-date information pertaining to the ongoing game. GRF is a powerful tool for reinforcement learning research that can help researchers and developers gain a better understanding of and solve various problems in football games.

Despite recent efforts, building agents for GRF still suffers from many difficulties [16–19]. First, the game involves both cooperative and competitive players, which results in a huge joint action space and the need to adapt to various opponents. Second, the goal of the game is to maximize the goal score, which requires a long sequence of perfect decisions and is challenging to achieve from random starting points. Third, the GRF introduces stochasticity into the environment, which improves agent robustness but also makes training more difficult by rendering the outcomes of specific actions uncertain.

Leading up to the competition, we tried to train the proposed agent using the self-play method and the league-learning method used by AlphaStar [3]. However, due to the sparsity of the reward in the GRF environment, the self-play algorithm does not converge well. The league-learning method also cannot perform properly because it is difficult to obtain multiple playing styles of multi-agent football AIs to form an opponent pool. This phenomenon occurs because the IEEE Conference on Games 2022 Football AI Competition is, to our knowledge, the first multi-agent football AI competition in which all players, excluding goalkeepers, are controlled.

To tackle the aforementioned issues and achieve promising outcomes in the 2022 IEEE Conference on Games Football AI Competition, we propose a two-stage reinforcement learning algorithm called mimic-to-counteract reinforcement learning (MCRL), which utilizes opponent demonstrations and policy distillation. This algorithm aims to counter the participating agents in the competition and creates two distinctive agents: sparring partner and primary agent. Drawing from the historical game logs of opponents, we encountered during the warm-up session, the sparring partner functions similar to human sparring partners in certain sports teams, whereby they simulate opponents with diverse styles of play, enabling primary players to practice against a range of policies, they may encounter in real competitions and thus improve their skills more effectively. Additionally, we developed multiple mentor agents, each with distinct strategies to counter the sparring partner. Their policies were subsequently distilled into a potent primary agent.

The key contributions of this paper are summarized as follows:

- This study represents a pioneering effort in building a policy distillation-based AI system that can take over the multi-agent GRF full game.
- By innovatively introducing the Wasserstein distance into the distributed PPO algorithm, MCRL can efficiently search for valuable strategies with stable updates and balance the relationship between policy iteration and policy style deviation.
- Through extensive experimentation, we demonstrate that MCRL outperforms existing algorithms in challenging GRF full-game scenarios.

The rest of the paper is structured as follows. Section 2 provides a brief overview of related work and preliminaries. Section 3 describes the MCRL algorithm in detail, including the sparring partner and primary agent architecture, deep reinforcement learning from demonstrations

Table 1 Comparison of some popular DRL benchmarks

Game	Competitive?	Multi-agent?	Stochastic?	Sparse reward?	Game length
Overcooked [11]	✗	✔	✗	✗	Typically 10^2
Atari games [12]	✗	✗	✗	Uncertain	Typically 10^2
Go [13]	✔	✗	✗	✔	10^2
ProcGen [14]	✗	✗	✔	✗	10^2
Honor of Kings [15]	✔	✔	✗	✗	10^3
GRF 11v11 [4]	✔	✔	✔	✔	10^3

GRF 11v11 has an extended game length and presents two major challenges for agent training: sparse rewards and a stochastic environment

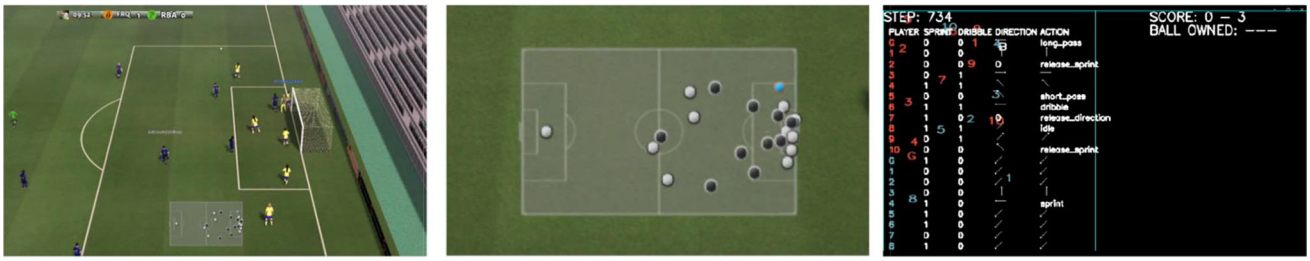

Fig. 1 Three types of observation in GRF. The first type consists of 1280 × 720 RGB images that correspond to the displayed screen. The second type is the super mini-map (SMM), which consists of four 72 × 96 matrices to record current situational data. The third sort of observation is the RAW observations, which encompass a dict that encapsulates the up-to-date information pertaining to the ongoing game

approach, and policy distillation technique. Section 4 presents the experimental setup and results of the MCRL in the Google research football environment. In Sect. 4.3, we conduct an ablation study to analyze the effects of our MCRL algorithm and its variants on RLChina AI Ranking. Finally, Sect. 5 discusses the limitations and future directions of our work.

2 Related work and preliminaries

2.1 Related work

2.1.1 Multi-agent reinforcement learning

Multi-agent reinforcement learning (MARL) [20] is a thriving research area, attracting significant attention within the machine learning community. In MARL [21, 22], agents work together or compete in complex environments, addressing challenges related to coordination and resource efficiency in multi-agent settings. The most mainstream training paradigm in MARL is centralized training with decentralized execution (CTDE) [23].

MARL algorithms are broadly categorized into two types: value-based algorithms and policy-based algorithms. Value-based algorithms estimate joint action-values [24]. This allows for decentralized execution [25], where each agent can independently select actions based on its individual Q-values [26]. Policy-based algorithms learn individual policies based on local observations and a centralized value function.

Current research in MARL aims to enhance multi-agent coordination and scalability, often through novel algorithm variants. For instance, MACKRL [27] extends CTDE with attention mechanisms to improve coordination. Population-based training methods like PBT [28] tune hyperparameters during training by exploiting mutations across a population of agents, improving scalability. EvoMARL [29] combines PBT with evolution strategies for large-scale MARL. Hierarchical frameworks decompose complex multi-agent tasks into high-level goals and low-level actions. HiPPO [30] learns goal-conditioned policies via hierarchical policy optimization, enhancing scalability across tasks. Transfer learning methods like ROMA [31] pretrain robust MARL policies in varied environments and then transfer to the target task, accelerating training. MATL [32] enables transfer at both the agent policy and execution levels. MAL [33] proposes a multi-agent locus algorithm to dynamically adjust agent behavior modes during training via an intrinsic reward, improving adaptation. STRAT [34] learns joint strategies over groups of agents via a consistency loss, enabling emergent team coordination. Works like MAGNet [35] incorporate agent-wise graph attention layers into policy networks, enhancing representation learning. In this work, we propose a new variant of MARL, which utilizes opponent demonstrations and policy distillation to train agents that can counter diverse opponent strategies in a multi-agent setting.

2.1.2 Deep reinforcement learning from demonstrations

Deep reinforcement learning from demonstrations (DfD) is a research area that aims to accelerate and enhance the training of DRL agents by leveraging expert or demonstrator data [36]. In DfD, the actor–critic architecture plays a pivotal role [37] which consists of two components: the actor and the critic. The actor determines the agent's actions based on the current state, representing its policy. The critic evaluates the quality of these actions by estimating the expected cumulative reward [38].

DfD uses optimization techniques [39] to update the policy and value networks iteratively, maximizing the expected cumulative reward and facilitating efficient learning. Trajectories [40] are sequences of states, actions, and rewards that agents experience during interactions with their environments. In DfD, demonstrations are considered

expert trajectories, providing valuable references for the agent's trajectory generation [41].

The current research in DfD focuses on refining techniques, such as reward shaping, imitation learning, and meta-learning [36], to seamlessly integrate demonstration data into DRL training. DDPGfD proposes a novel two-stage learning framework to identify an optimal restoration strategy which builds on the deep deterministic policy gradient from demonstrations [42]. CSGP proposes a closed-loop safe grasp planning approach via attention-based DRL from demonstrations [43]. We design a new DfD method which tilizes opponent demonstrations to learn how to counter their strategies, rather than accelerating the training of agents with the same objectives as experts from expert demonstrations.

2.1.3 AI for football games

Football environments are crucial for AI research, blending multiple challenges such as control, strategy, cooperation, and competition. Various simulators beyond GRF have been introduced, including rSoccer [44] and JiDi Olympics Football [17]. These platforms offer basic environments where players, depicted as rigid bodies, perform limited actions such as moving or pushing the ball. On the other hand, GRF expands the action space to include additional mechanics such as slide-tackling and sprinting.

Other platforms such as the RoboCup Soccer Simulator [45, 46] and DeepMind MuJoCo Multi-Agent Soccer Environment [47] prioritize low-level robotic control, necessitating intricate manipulation of a player's joints. In contrast, GRF simplifies this, allowing agents to focus on honing advanced behaviors and strategies. A 2020 competition in the GRF environment on Kaggle drew over a thousand teams. Participants were tasked with creating an agent to control a single player, while a built-in AI managed the teammates.

The winning team, WeKick [48], employed imitation learning and distributed league training. Unlike this arrangement, our system ambitiously undertakes the control of all 10 outfield players simultaneously in a decentralized manner. The only preceding work with a similar goal, TiKick [16], utilized a demonstration dataset from WeKick and offline RL techniques for agent training. TiZero [17], a modified version of TiKick, employs self-play for training agents without the need for demonstration data. In this work, we propose MCRL, a novel two-stage MARL algorithm for GRF. MCRL distinctively employs opponent demonstrations to create diverse sparring partners, enhancing primary agents' adaptability. Additionally, it integrates the Wasserstein distance into the PPO algorithm, ensuring efficient strategy updates and a balance between policy iteration and style deviation.

2.2 Preliminaries

2.3 Reinforcement learning

MARL involves learning how to make decisions in a decentralized and partially observable environment. This type of learning is often formalized as decentralized partially observable Markov decision processes (Dec-POMDPs) [49]. In this framework, agents work together to select joint actions that maximize the expected cumulative reward over time. The Dec-POMDP is defined as a tuple $(\mathcal{N}, S, A, P, r, O, G, \gamma)$, where $\mathcal{N} \equiv \{1, \ldots, n\}$ is the set of n agents, S is the state space, A is the action space, $r(s, \boldsymbol{a})$ is the global reward function, O is the observation space, and $\gamma \in [0, 1)$ is the discount factor. At each time step, each agent $i \in \mathcal{N}$ receives an observation $o \in O$ according to the observation function $G(s, i)$ and selects an action $a_i \in A$ to form a joint action $\boldsymbol{a}$. The environment then transitions to a new state s' based on the transition function $P(s' \mid s, \boldsymbol{a})$. Each agent only has access to its own local observations $o^i_{1:t}$ and uses a policy function $\pi^i(a_i \mid o^i_{1:t})$ to decide what action to take. A trajectory $\tau^i = (s_1, o^i_1, a^i_1, \ldots)$ for agent $i \in \mathcal{N}$ is a sequence of states, observations, and actions. For multi-agent scenarios, the joint trajectory comprises individual agent trajectories. The optimization objective of MARL is to learn a joint policy π that maximizes the expected cumulative reward $\mathbb{E}_{s_t, \boldsymbol{a}_t}\left[\sum_t \gamma^t r(s_t, \boldsymbol{a}_t)\right]$ over time.

In this work, we follow the standard actor–critic framework. The actor, denoted as $\pi(a|s; \theta)$, defines the policy, mapping states s to actions a using parameters θ. The critic, $V(s; \phi)$, estimates the expected return from state s with parameters ϕ. The actor's objective function is given by $J(\theta) = \mathbb{E}_{\pi(a|s;\theta)}\left[\sum_t \gamma^t r(s_t, \boldsymbol{a}_t)\right]$, and the critic's temporal-difference error is represented as $\delta_t = r_t + \gamma V(s_{t+1}; \phi) - V(s_t; \phi)$. Using these, the actor–critic framework adjusts θ and ϕ to optimize the policy and value function, respectively.

2.4 Learning from opponent demonstrations

In this work, we apply DfD to train the sparring partners. Rather than leveraging expert demonstrations to accelerate and guide the learning process [50–53], we leverage opponent demonstrations to learn how to counter their strategies. Given opponent trajectories $\mathcal{D} = \{(s_t, a_t)\}_{t=1}^n$, a policy π is initialized via $\min_\theta \mathbb{E}_{(s_t, a_t) \sim \mathcal{D}}[\mathcal{L}(\pi(\cdot|s_t; \theta), a_t)]$, to mimic the policy demonstrated in $\mathcal{D}$. Various f-divergences [54, 55] and mean square error can be used to measure the loss. In this work, we make $\mathcal{L}(\pi(\cdot|s_t; \theta), a_t) = \frac{1}{T}\sum_{t=0}^{T}[\pi(\cdot|s_t; \theta) - a_t]^2$. Post-

initialization, π is refined using DRL by maximizing the expected cumulative reward $\mathbb{E}_{s_t,\boldsymbol{a}_t}\left[\sum_t \gamma^t r(s_t,\boldsymbol{a}_t)\right]$ over time.

To ensuring efficient strategy updates and a balance between policy iteration and style deviation, it is necessary to introduce a regularization term to measure the difference between the original policy and the updated policy. Following WDAIL [56], we use Wasserstein distance [57, 58] to weight the discrepancy of trajectory distribution between the policy π and the expert policy μ. The distance is formulated as follows:

$$\mathcal{L}_{WD} = \sup_{\|f\|_L=1} \left\{E_{x\in\tau_\pi}[f(x)] - E_{y\in\tau_\mu}[f(y)]\right\} \tag{1}$$

The Lipschitz function f, used to measure the disparity between the two distributions τ_π and τ_μ, should be restrained by the 1-Lipschitz condition $\|f\|_L = 1$. In practice, the weight clipping and gradient penalty are the effective methods for enforcing the L1-Wasserstein distance satisfying the 1-Lipschitz constraint.

2.5 Policy distillation

Policy distillation was first presented at the International Conference on Learning Representations as a novel method [59, 60] to extract the policy of a reinforcement learning agent and train a new network that performs at the expert level while being dramatically smaller and more efficient [61]. In this work, we apply this algorithm to distill the policies of the mentors into the primary agent. In the actor–critic framework of policy distillation, the actor of the mentor model, denoted as $\pi_{\text{mentor}}(a|s;\theta)$, dictates the policy or action selection, and the critic of the mentor model, represented as $V_{\text{mentor}}(s;\phi)$, estimates the value or advantage of such actions, with θ and ϕ being the respective parameters. Analogously, the student model has an actor $\pi_{\text{student}}(a|s;\theta')$ and a critic $V_{\text{student}}(s;\phi')$ with parameters θ' and ϕ'. Typically, the f-divergence, $D_f(\pi_{\text{mentor}}(\cdot|s;\theta)||\pi_{\text{student}}(\cdot|s;\theta'))$ is employed for the actors [62]. Simultaneously, for the critics, a mean squared error $(V_{\text{mentor}}(s;\phi) - V_{\text{student}}(s;\phi'))^2$ elucidates the difference [15]. To amalgamate these, the composite loss is delineated as follows:

$$\begin{aligned}\mathcal{L}_{\text{distill}} = \mathbb{E}_s\big[&D_f(\pi_{\text{mentor}}(\cdot|s;\theta)||\pi_{\text{student}}(\cdot|s;\theta')) \\ &+\lambda\frac{1}{2}(V_{\text{mentor}}(s;\phi) - V_{\text{student}}(s;\phi'))^2\big]\end{aligned} \tag{2}$$

here λ is a scalar weight adjudicating the trade-off between actor and critic losses. In this paper, following WDAIL [56], we innovatively employ the Wasserstein distance [57, 58] as a yardstick, heralding a paradigm shift in discerning the nuances between the mentor and student actors. The composite loss is renewed as follows:

$$\begin{aligned}\mathcal{L}_{\text{distill}} = \mathbb{E}_s\Bigg[&\sup_{\|f\|_L=1}\left\{E_{x\in\tau_{\pi_{\text{mentor}}}}[f(x)] - E_{y\in\tau_{\pi_{\text{student}}}}[f(y)]\right\} \\ &+\lambda\frac{1}{2}(V_{\text{mentor}}(s;\phi) - V_{\text{student}}(s;\phi'))^2\Bigg]\end{aligned} \tag{3}$$

3 Methodology

3.1 Overview of MCRL

3.1.1 Two distinctive agents

The objective of the IEEE Conference on Games 2022 Football AI Competition is to score more goals than the opposition. Therefore, the proposed multi-agent football AI must be able to manage opponents with various policies. For example, if an opponent is running away from the proposed player during a chase or face-off, it must be able to predict that this player may be looking for space to receive a pass from a teammate and then quickly intercept the ball before it enters open space [16, 17].

Leading up to the competition, we assumed that if the proposed multi-agent football AI is trained through self-play, it will become self-improving through trial and error. However, empirically, the outcomes of self-play using the MAPPO algorithm were unsatisfactory [63, 64], which we hypothesize and may result from the sparsity of reward in the GRF environment. Thus, we used the league-learning method used by AlphaStar [3]. However, in the IEEE Conference on Games 2022 Football AI Competition, the league-learning method was challenging because it was difficult to obtain multiple playing styles of multi-agent football AIs to directly train the proposed agent. This issue occurred because the IEEE Conference on Games 2022 Football AI Competition is, to our knowledge, the first multi-agent football AI competition in which all players, excluding goalkeepers, are controlled.

Therefore, in MCRL, multiple single-agent football AIs (henceforth referred to as the sparring partner) are introduced, aiming to simulate opponents with various policies that we may face in real competitions. The partners were trained from opponent demonstrations as mentioned in Sect. 2.2, whose strategic styles were similar to those of the opponents we encountered in the IEEE Conference on Games 2022 Football AI Competition warm-up session, and to achieve a counter-policy effect, these sparring partners were used to train a multi-agent football AI (henceforth referred to as the primary agent), which will learn to counter these partners' strategies as the training advances.

3.1.2 Two stages of MCRL

The MCRL methodology unfolds in two pivotal stages, each contributing to the robust training of multi-agent football AI, ensuring that it is well-equipped to counter diverse opponent strategies encountered in competitions such as the IEEE Conference on Games 2022 Football AI Competition.

In the first stage, partners are meticulously trained utilizing opponent demonstrations. This approach ensures the assimilation of diverse strategic styles akin to those encountered in formal competitions. The incorporation of the Wasserstein distance and PPO-Clip loss within the training paradigm guarantees the retention of policy style while enabling effective policy iteration and updates. This stage lays the foundational groundwork, preparing the agents with a comprehensive understanding and adaptation to varied strategic approaches.

The second stage embarks on the training of the primary agent, leveraging the competencies and insights garnered from the trained partners in the initial stage. Engaging in self-play with these partners, the mentor agents undergo a rigorous learning process, honing its ability to adeptly counteract policy styles of partner. The employment of policy distillation further augments this learning phase, with mentor agents guiding the primary agent to seamlessly adapt to multifaceted opponent strategies, ensuring its preparedness and agility in formal competition scenarios.

3.2 Sparring partner training

3.2.1 Observation and model design

Creating observations is the initial stage in developing a DRL model. Initially, the GRF environment offered three types of observations. The first type consists of 1280×720 RGB images that correspond to the screen shown. The second type is the super mini-map (SMM), which consists of four 72×96 matrices to record current situational data. The third type of observation is the RAW observations, which encompass a dict that encapsulates the up-to-date information pertaining to the ongoing game. There are two variations of the 115-dimensional vector representation that describes all the state information. To extract the hidden features, the pixel-level and SMM representations require increasingly complex depth models. Even using the lightweight MobileNetV2 [65, 66] for feature extraction, the real results showed that the model's training pace and memory consumption were inadequate. Therefore, the input of the proposed deep model was RAW observations. To extract hidden characteristics from the original input, we constructed an actor and critic network with shared parameters comprising five fully connected layers, one convolutional layer, and a layer of LSTM [67–69]. An overview of the model architecture is shown in Fig. 2. Except for the final output layer, which uses softmax, all hidden layers are followed by a ReLu. Similar to open AI's baseline and MAPPO [63, 64], the learning rate during training is set to $1e-5$ and is fixed during training. The network parameters were initialized using an orthogonal matrix [70] and updated using the Adam optimizer [71].

3.2.2 Training partners from opponent demonstrations

The first stage of MCRL is training partners from opponent demonstrations which adheres to the classical actor–critic framework as mentioned in Sect. 2.2, where the actor network and the critic network are analogous to the contestant and the judge, respectively. The actor network infers the action distribution based on the current state, while the critic network outputs the value of that state under the current policy. The procedure of the first stage of MCRL is shown in Fig. 3.

The historical game logs of multi-style opponents we encountered during the warm-up session at the IEEE Conference on Games 2022 Football AI Competition constitute the training data $\mathcal{D}_p \equiv \{\mathcal{D}_p^1, \ldots, \mathcal{D}_p^i\}$ for the multi-style pre-trained model. And historical game logs with the ith policy style are represented by a sequence of state–action pairs of length n:

$$\mathcal{D}_p^i = \left\{\left(s_t^i, a_t^i\right)\right\}_{t=1}^n, \tag{4}$$

These game logs, which are in the form of dump files, provide information about the actions taken by each player, the state of the game at each time step, and any relevant metadata, which can be used to train the pre-trained model through the extraction and labeling of relevant features. Our primary goal was to train a parameterized pre-trained policy $\mu_\psi^i(\cdot \mid s)$ that mimicked the implicit policy in the i-th data, where ψ as a pre-trained policy parameter and is updated using the mean squared error loss, computed only on the demonstration examples for training the actor:

$$\mathcal{L}_{\text{pre-trained}} = \frac{1}{\left|\mathcal{D}_p^i\right| T} \sum_{\tau \in \mathcal{D}_p^i} \sum_{t=0}^{T} \left[\mu_\psi\left(\cdot \mid s_t^i\right) - a_t^i\right]^2, \tag{5}$$

This loss is a standard component in imitation learning. For actions, we apply an action mask to the pre-trained policy $\mu_\psi^i(\cdot \mid s)$, preventing it from selecting built-in action (action 19), which is generated by the built-in agent of the GRF environment and retrieved via rule-based strategies. All non-designated players are assigned to built-in action (action 19). The built-in action is solely used to create partners; it was not used in the production of the primary agent.

The weights of pre-trained policy $\mu_\psi^i(\cdot \mid s)$ were utilized for the initialization of the actor model $\pi_\theta^i(\cdot \mid s)$, which can

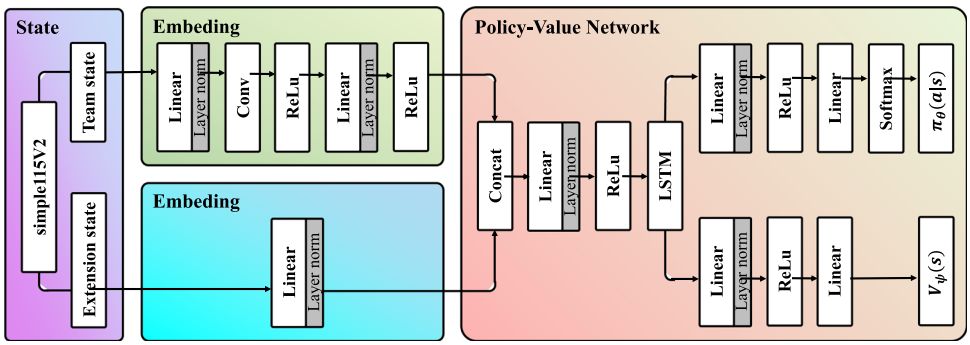

Fig. 2 Overview of the proposed model architecture

be used to expedite the training of sparring partners and imbue them with similar policy styles to those of the proposed opponents. These multi-style partners will be used to teach the primary agent, which will learn how to counter these opponents' policies as the training advances.

Actor–critic framework-based single-agent sparring partner training requires obtaining a parameterized policy $\pi^i_\theta(\cdot \mid s)$ and a parameterized value function $V^i_\phi(s)$. θ can be updated by the PPO-Clip loss:

$$\mathcal{L}_{\text{PPO-Clip}} = \frac{1}{|\mathcal{D}^i_k|T}\sum_{\tau\in\mathcal{D}^i_k}\sum_{t=0}^{T}\min\left(\frac{\pi^i_\theta(\cdot \mid s^i_t)}{\pi^i_{\theta_k}(\cdot \mid s^i_t)}A^{\pi^i_{\theta_k}}, \text{clip}\left(\frac{\pi^i_\theta(\cdot \mid s^i_t)}{\pi^i_{\theta_k}(\cdot \mid s^i_t)}, 1-\varepsilon, 1+\varepsilon\right)A^{\pi^i_{\theta_k}}\right) \tag{6}$$

where training sample $\mathcal{D}^i_k$ is generalized by running policy $\pi^i_{\theta_k}$ in the GRF environment. GAE was used to estimate the advantage function $A^{\pi^i_{\theta_k}}$, with the discount factor γ set to 0.993 and the parameter λ set to 0.96. We empirically show that the style of the policy $\pi^i_\theta(\cdot \mid s)$ updated by the PPO-Clip loss differs from the pre-trained policy $\mu^i_\psi(\cdot \mid s)$ contained in the pretrained model. This process influences the counter-policy effect of the primary agent.

In this work, we employ the Wasserstein distance, as opposed to the frequently utilized f-divergence, as a regularization term within this algorithm. Given that it is more customary to employ gradient descent instead of ascent in machine learning, by interchanging the variables $\boldsymbol{x}$ and $\boldsymbol{y}$ in Eq. (1), we derive the result denoted as follows:

$$\begin{aligned}\mathcal{L}_{\text{WD}} &= \inf_{\|f\|_L=1}\left\{E_{y\in\tau_\mu}[f(y)] - E_{x\in\tau_\pi}[f(x)]\right\}\\ &= \inf E\|d(x,y)\|_1\\ &= \inf_{\gamma\in\Pi[\tau_\pi,\tau_\mu]} E_{(\boldsymbol{x}-\boldsymbol{y})\in\gamma(\boldsymbol{x}-\boldsymbol{y})}\|d(\boldsymbol{x}-\boldsymbol{y})\|\end{aligned} \tag{7}$$

Furthermore, $\mathcal{L}_{\text{WD}}$ can be reformulated in the form of the cumulative distribution functions of probability distributions in the following equation:

$$\mathcal{L}_{\text{WD}} = \inf_{\gamma\in\Pi[\tau_\pi,\tau_\mu]}\int |\boldsymbol{x}-\boldsymbol{y}|d\gamma(\boldsymbol{x}-\boldsymbol{y}) \tag{8}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are random variables from policy distribution τ_π and expert distribution τ_μ, respectively, and γ denotes the joint distribution. We replace the integral $\inf_{\gamma\in\Pi[\tau_\pi,\tau_\mu]}\int |\boldsymbol{x}-\boldsymbol{y}|d\gamma(\boldsymbol{x}-\boldsymbol{y})$ with a sum over discrete points

$$\inf_{\gamma\in\Pi\left[\pi^i_\theta(\cdot|s^i_t),\mu^i_\psi(\cdot|s^i_t)\right]}\sum_{x,y}\gamma(\boldsymbol{x},\boldsymbol{y})\|\boldsymbol{x}-\boldsymbol{y}\|.$$

The Wasserstein distance $\mathcal{L}_{\text{WD}}$ of π^i_θ and μ^i_ψ can then be expressed as follows:

$$\mathcal{L}_{\text{WD}} = \frac{1}{|\mathcal{D}^i_k|T}\sum_{\tau\in\mathcal{D}^i_k}\sum_{t=0}^{T}\inf_{\gamma\in\Pi\left[\pi^i_\theta(\cdot|s^i_t),\mu^i_\psi(\cdot|s^i_t)\right]}\sum_{x,y}\gamma(\boldsymbol{x},\boldsymbol{y})\|\boldsymbol{x}-\boldsymbol{y}\|, \tag{9}$$

where $\pi^i_\theta(\cdot \mid s^i_t)$ and $\mu^i_\psi(\cdot \mid s^i_t)$ are policies denoting the conditional action probabilities for state s^i_t, with θ and ψ as their respective parameter sets. $\sum_{x,y}\gamma(\boldsymbol{x},\boldsymbol{y})\|\boldsymbol{x}-\boldsymbol{y}\|$ quantifies the Wasserstein distance between two policy action distributions in a trajectory τ, reflecting their dissimilarity in a latent space. $\inf_{\gamma\in\Pi\left[\pi^i_\theta(\cdot|s^i_t),\mu^i_\psi(\cdot|s^i_t)\right]}$ finds the distribution γ that minimizes the Wasserstein distance between policies $\pi^i_\theta(\cdot \mid s)$ and $\mu^i_\psi(\cdot \mid s)$ for each time step t, opponent policy style i, and trajectory τ.

The final actor training loss is the combination of the PPO-Clip loss and the type-1 Wasserstein distance of $\pi^i_\theta(\cdot \mid s)$ and $\mu^i_\psi(\cdot \mid s)$:

$$\mathcal{L}_{\text{actor}} = \mathcal{L}_{\text{PPO-Clip}} + \eta\mathcal{L}_{\text{WD}}, \tag{10}$$

where η is a distance balancing coefficient. $\mathcal{L}_{\text{actor}}$ forces the values of the other actions to be at least a margin lower than the value of the demonstrator's action. Adding $\mathcal{L}_{\text{WD}}$ grounds the values of the unseen actions to reasonable values and makes the greedy policy induced by the value function imitate the demonstrator. The parameter ϕ of $V^i_\phi(s)$ is updated by the mean squared error loss:

$$\mathcal{L}_{\text{critic}} = \frac{1}{|\mathcal{D}_k^i|T} \sum_{\tau \in \mathcal{D}_k^i} \sum_{t=0}^{T} \left(V_\phi^i(s_t^i) - R_t^i \right)^2. \tag{11}$$

Initialized with the parameters of the pre-trained model, the single-agent sparring partner was trained via a self-play approach, which started training through the experiences from the matches against itself. The models were saved every 15,000,000 time steps, and these saved models form a pool of opponents. The opponents are then sampled from the pool. Fifty percent of the opponents come from the 20 most recent models, and the rest come from a uniform random sampling of the entire pool. In the end, approximately 380 agents were in the pool. The training procedure of the MCRL is summarized in Algorithm 1. In the next section, the training method of our primary agent with the counter-policy effect will be introduced.

Algorithm 1 MCRL

Input: Distance balancing coefficient η, Initial pre-trained policy parameters ψ, Initial partner policy parameters θ, Initial partner value function parameters ϕ, pre-trained decision trajectory $\mathcal{D}_p$
Stage 1: Sparring Partner Training
for $i = 0, 1, 2, \ldots$ **do**
  while *not converged* **do**
    Update the pre-trained policy μ_ψ^i by minimizing the loss in Eq. (5) via stochastic gradient ascent with Adam;
    Initialize π_θ^i with μ_ψ^i;
  end
end
for $i = 0, 1, 2, \ldots$ **do**
  for $k = 0, 1, 2, \ldots$ **do**
    Collect set of trajectories $\mathcal{D}_k^i$ by running current policy $\pi_{\theta_k}^i$ in GRF;
    Compute rewards to go $\hat{R}_t^i$;
    Compute advantage estimates $\hat{A}_t^i$(using the GAE method of advantage estimation) based on the current value function V_ϕ^i;
    Update the policy π_θ^i by maximizing the loss in Eq. (10);
    Fit the value function V_ϕ^i by regression on the loss in Eq. (11);
  end
end
Input: Scalar weight λ, Initial mentor policy parameters ψ', Initial mentor value function parameters ω', Initial primary agent policy parameters θ', Initial primary agent value function parameters ϕ'
Stage 2: Primary Agent Training
for $i = 0, 1, 2, \ldots$ **do**
  Update the actor model $\mu_{\psi'}^i$ and critic model $V_{\omega'}^i$ of mentor;
  Generate training sample $\mathcal{D}_m^i$ by running $\mu_{\psi'}^i$ in GRF;
end
while *not converged* **do**
  Update the actor and critic models $\pi_{\theta'}$ and $V_{\phi'}$ of the primary agent by minimizing the distillation loss in Eq. (12);
end

3.3 Primary agent training

3.3.1 Primary agent training with league learning

One baseline method for training game AI is league learning [3], which was proposed by the Tencent AI Lab in 2021 in the form of WeKick. JueWu is a strategy cooperative AI developed jointly by the Tencent AI Lab and Honor of Kings, and the WeKick version was obtained through the transfer of the complete JueWu body and targeted adjustments for the football task. WeKick participated in the first Google Football Kaggle competition, and in this world-class AI football competition, WeKick defeated 1138 outstanding teams with an absolute advantage of 1785.8 points to win the championship. In this work, we developed a primary agent using this baseline algorithm and compared its performance to an agent trained by MCRL. We constructed a league (multiple

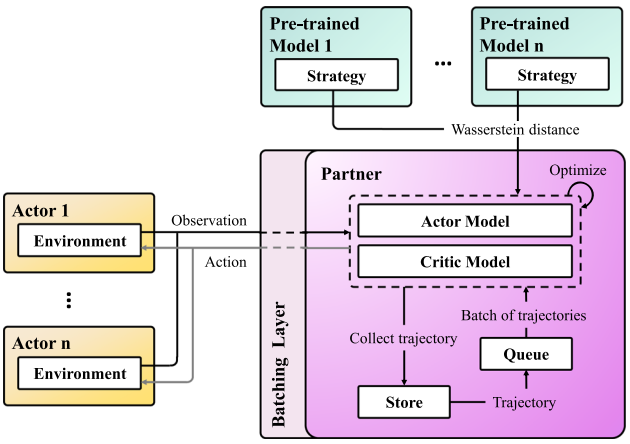

Fig. 3 The procedure of the first stage of MCRL. This procedure is derived from the SEED RL [72], with the weights of the pre-trained model utilized for the initialization of the actor model, and the Wasserstein distance between the action distributions output by the pre-trained model and the actor model is employed to ensure efficient strategy updates and a balance between policy iteration and style deviation. See Fig. 2 for a thorough architecture of the actor–critic model

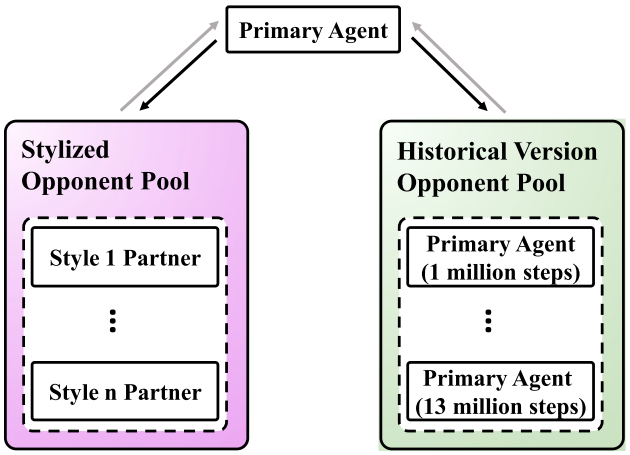

Fig. 4 Overview of the league-learning framework

policy pool) comprising sparring partners and the historical iterations of the primary agent and trained the primary agent based on this league. The process is shown in Fig. 4.

The stylized sparring partner agents focus on one specific playing style, while the primary agent not only faces its historical versions but also regularly includes stylized sparring partner agents as opponents to ensure that the primary agent can adapt to opponents with completely different styles and achieve good results in competitions.

3.3.2 Primary agent training using MCRL

The second stage of MCRL is training primary agent using policy distillation. To achieve the desired counter-policy effect, the primary agent's learning must be guided by a paradigm. The concept of policy distillation [61, 73] for multiple tasks motivated us to distill the knowledge of n single-task networks. The student network selects data in a different buffer in each episode for learning, which is more efficient than allowing the student network to directly learn in a multitasking environment. As shown in Fig. 5, we performed the policy distillation mentioned in Sect. 2.2 based on each partner's policy style. Firstly, the mentor agents are trained to counteract the sparring partners. We placed the partners into the opponent pool of these mentor agents. To prevent overspecialization and maintain fundamental abilities, we also regularly treat historical versions of the model as adversaries during the training process. After 13 million timesteps, mentors can effectively counteract opponents with specific policy styles, but they are unable to cope with variations in policies.

Then, we distilled n counter-policies acquired by mentors against opponents of a specific policy style. Training the primary agent with samples in a meaningful sequence instills the primary agent with the mentor's knowledge and adapts it to opponents with completely different styles, thus achieving the counter-policy effect. Distillation is a supervised process based on the loss function in Eq. (12):

$$\mathcal{L}_{\text{distill}} = \frac{1}{|\mathcal{D}_m^i|T} \sum_{\text{mentor}_i} \sum_{\tau \in \mathcal{D}_m^i} \sum_{t=0}^{T} \left[\mathcal{L}_{\pi_{\theta'}} + \lambda \mathcal{L}_{V_{\phi'}} \right]. \tag{12}$$

Equation (12) is another form of Eq. (3), utilized for discerning the nuances between the mentor and the student, where λ is a scalar weight adjudicating the trade-off between losses. Following Eq. (3), we can know that $\mathcal{L}_{V_{\phi'}} = \frac{1}{2}\left(V_{\omega'}^i(s_t) - V_{\phi'}(s_t)\right)^2$. And through algebraic manipulations analogous to those from Eq. (7) to Eq. (9), we can derive that $\mathcal{L}_{\pi_{\theta'}} = \inf_{\gamma \in \Pi \left[\mu_{\psi'}^i, \pi_{\theta'}\right]} \sum_{x,y} \gamma(\boldsymbol{x}, \boldsymbol{y}) \|\boldsymbol{x} - \boldsymbol{y}\|$.

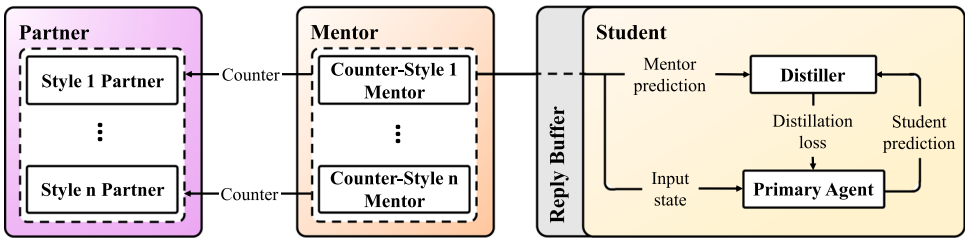

Fig. 5 Overview of the workflow for training the primary agent using MCRL

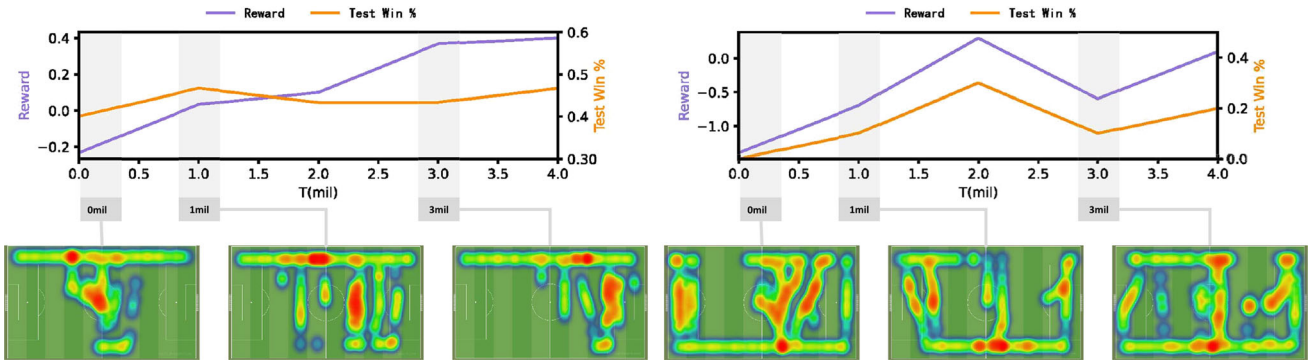

Fig. 6 Performance of the sparring partners in *the GRF_11v11_Stochastic* scenario. The weights of the pre-trained agent were utilized for the initialization (0 mil) of the sparring partners. The win rate and reward are evaluated via versus the built-in hard AI of GRF, while the heatmaps are generated via versus the built-in random AI of GRF

The training sample $\mathcal{D}_m^i$ is generalized by running the i-th mentor agents in the GRF environment, where $\mu_{\psi'}^i$ and $V_{\omega'}^i$ are the actor and critic networks of the i-th mentor, and $\pi_{\theta'}$ and $V_{\phi'}$ refer to the actor and critic networks of the primary agent, respectively. The training procedure of the MCRL is summarized in Algorithm 1.

4 Experiment

4.1 Evaluation of sparring partner

In the first stage of MCRL, we developed a number of multi-style single-agent football AIs (partners) that possess similar policy styles to the opponents, we encountered in the IEEE Conference on Games 2022 Football AI Competition warm-up session. We thus simulate opponents with various policies that we may face in real competition within the multi-agent game environment of GRF. To avoid stark differences in policy style between the partners and the pre-trained agents, the Wasserstein distance of $\pi_\theta(a_t \mid s_t)$ and $\mu_\psi(a_t \mid s_t)$ was introduced into the loss function. In this section, we demonstrate and explain the effectiveness of the sparring partners in maintaining policy style in the challenging scenario of the GRF 11v11 full game rather than example scenarios. In these experiments, partners were initialized with the pre-trained model parameters and iterated 4,201,530 timesteps using MCRL.

Figure 6 shows the heatmaps of the partner trained using MCRL at the initialization state and after 1 million and 3 million timesteps. After 4 million timesteps using MCRL, the partner's winning rate and reward against the hard built-in AI were markedly improved, and its heatmap was similar to that of the pre-trained agent. Introducing the Wasserstein distance allowed the partner to retain policy style similar to that of the pre-trained agent. Experimental results thus demonstrate that MCRL can efficiently search for valuable strategies with stable updates and balance the relationship between policy iteration and policy style deviation by introducing the Wasserstein distance into the distributed PPO algorithm.

4.2 Evaluation of primary agent

In this section, we consider the challenging GRF 11v11 full-game scenario (rather than toy scenarios) to validate the effectiveness of the proposed methods. We compare the proposed method against the self-play method, the league-learning method, and the method of playing against rule-based agents. The mean and variance of the performances of each method are presented using three random seeds.

4.2.1 RLChina AI ranking list

The official website of the RLChina community is an open platform that contains rich resources and content [74]. The website provides abundant literature about reinforcement learning, including the latest research progress, academic lectures, technical articles, code practices, and practical cases, which are suitable for both beginners to quickly obtain started and professionals to meet their in-depth academic needs.

On the website, users can read a lot of content about reinforcement learning, including research papers, technical articles, and industrial cases, to understand the latest research progress and application scenarios. Users can also participate in online activities, technical exchanges, express personal opinions and experiences, and communicate and discuss with other users.

In addition, the RLChina community provides open-source reinforcement learning code libraries and examples, as well as an open online algorithm competition platform named Jidi [75], which provides users with many choices of environments, high-quality competitions, real-time

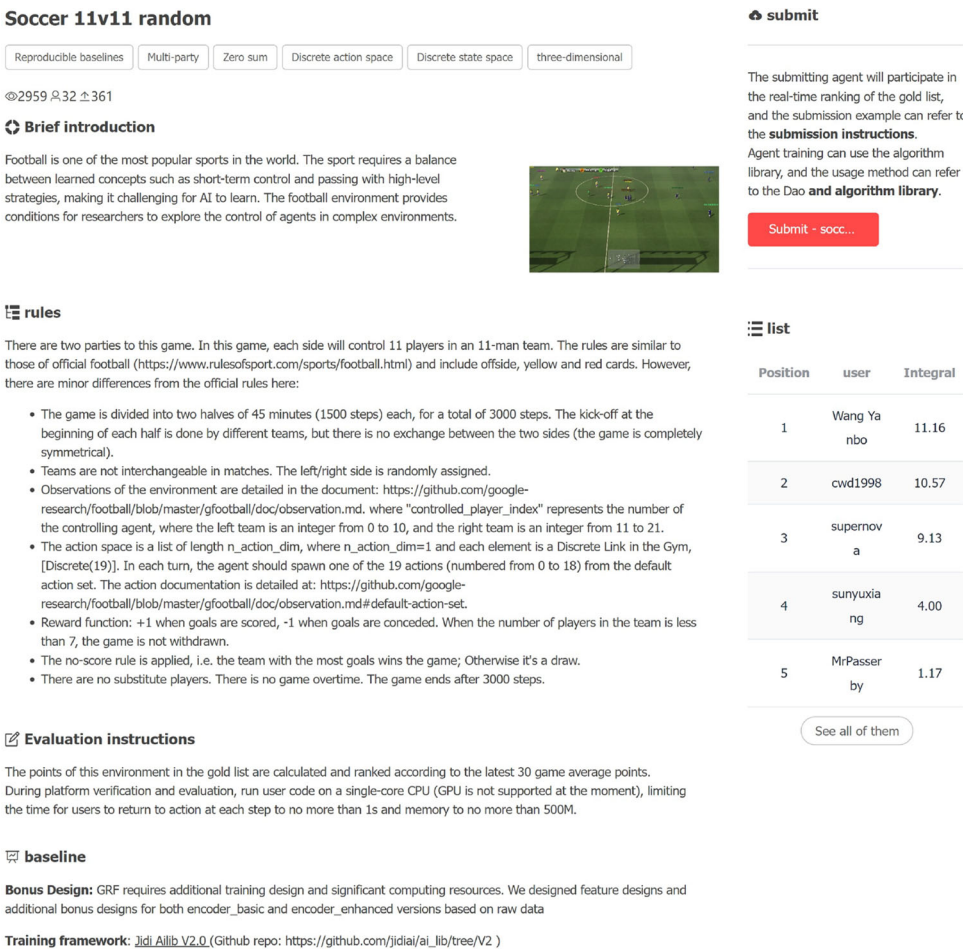

Fig. 7 *Football_11v11_Stochastic* subject on the *Jidi* platform by *RLChina*. This subject uses the same scenario as the IEEE Conference on Games 2022 Football AI Competition, which features a ranking list at the bottom right corner displaying the Elo scores and rankings of participating agents. The proposed agent achieved an Elo score of 4.00 and ranked 4/361 on the ranking list

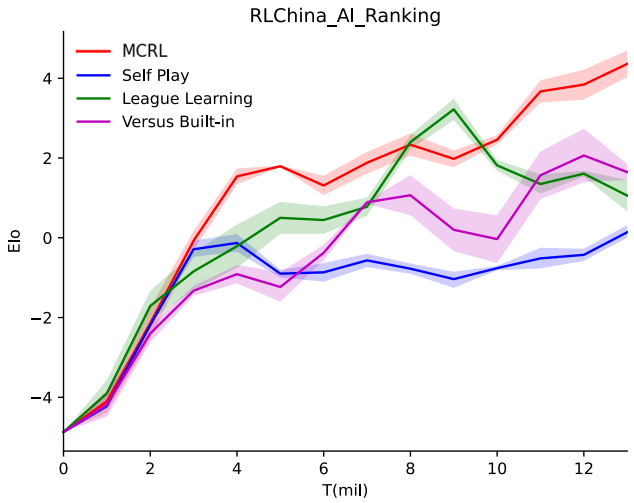

Fig. 8 Comparison of the primary agent against baseline methods on *RLChina_AI_Ranking*. We save the model every 1 million timesteps and submit it to the ranking list for evaluation, which involves a 72-h testing period consisting of 20 game matches, and record the final Elo rating

discussions, and fair algorithm rankings. The platform primarily offers five user functions:

- Jinbang (Ranking List) provides classification rankings of algorithms in different environments, as well as overall rankings, in real time. Users can view the dynamic ranking of their submitted algorithms, replay gameplays, and access detailed information in this study.
- Kemou (Subject) provides different intelligent agent environments for users to select and participate by submitting algorithms. Algorithm submissions in subjects are evaluated in real time and displayed in Jinbang (Ranking List).
- Miji (Algorithm) provides commonly used and popular intelligent agent algorithms, as well as detailed explanations of how each algorithm works and how it can be reproduced in applicable environments.

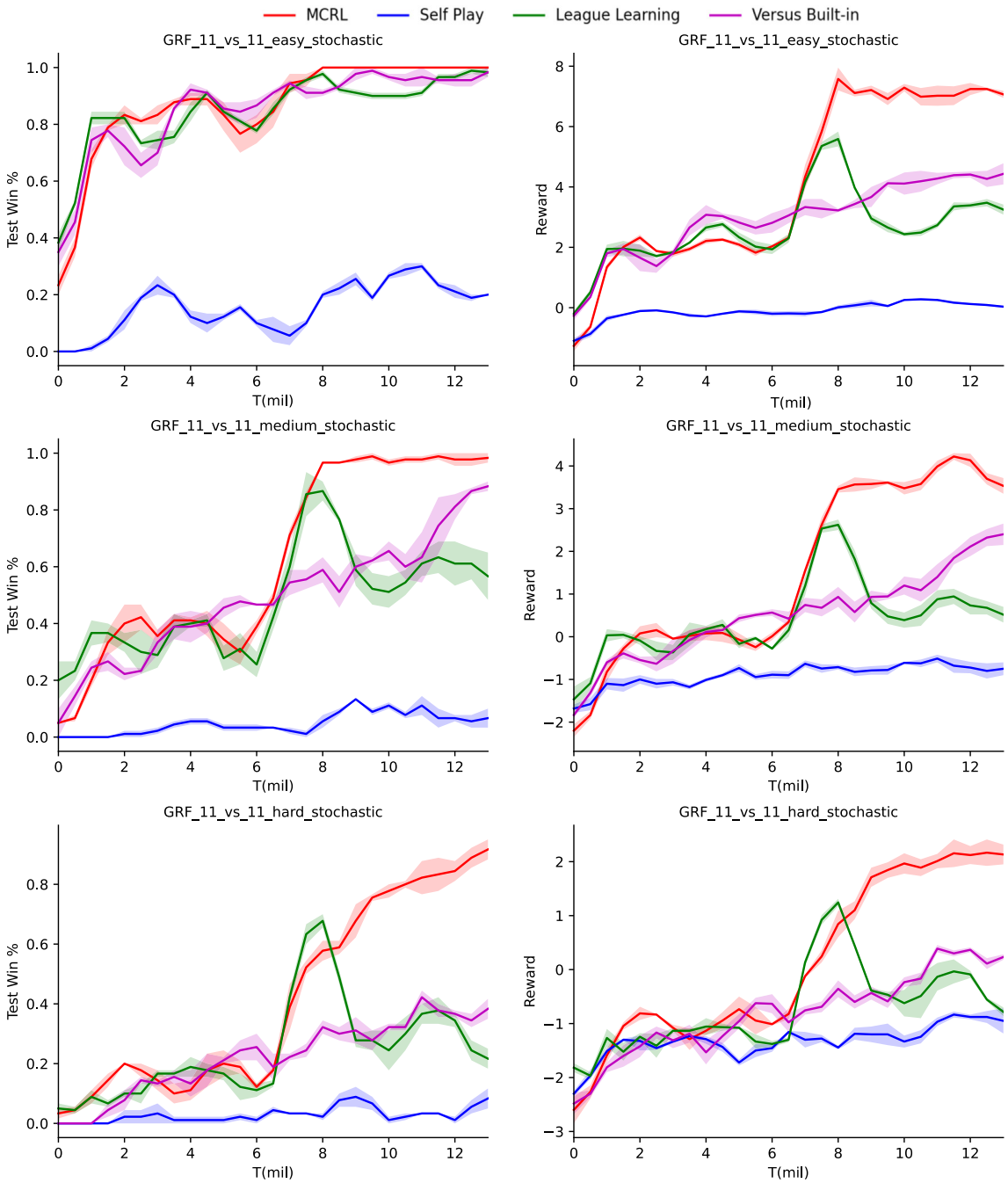

Fig. 9 Comparison of the performance of the primary agent against baseline methods on the *GRF_11v11_Stochastic* scenario against easy, medium, and hard built-in AI. *Left.* The winning rates. *Right.* The unshaped rewards

- Leitai (Arena) offers high-quality competitions that users can join based on their interests and obtain corresponding rewards.
- Lundao (Discussions) is a real-time communication platform where users can post topics for discussion, share experiences, and find like-minded individuals.

The Jidi platform by RLChina serves as the official platform for the IEEE Conference on Games 2022 Football AI Competition [75], showcasing an AI ranking list that aligns with the competition's scenario, as shown in Fig. 7.

The proposed method is evaluated using two benchmarks: the Football AI Ranking List by RLChina [74] shown in Fig. 8, and the challenging 11v11 stochastic

Table 2 Ablation studies of the MCRL in *RLChina_AI_Ranking*

Method	Rate			Score	
	Win	Draw	Fail	Goal	Elo
MCRL-npd	0.65	0.10	0.25	1.0	−0.70
MCRL-np	0.25	0.30	0.45	1.8	−0.65
MCRL-nd	0.40	0.30	0.30	2.1	1.45
MCRL	0.90	0.10	0.00	4.8	4.80

Ablation studies of MCRL which shows a good pertinence to *RLChina_AI_Ranking* and its three variants. Each evaluation involves a 72-h testing period consisting of 20 game matches, and the average winning rate, draw rate, failure rate, number of goals, and the final Elo are recorded

scenario of GRF shown in Fig. 9. The proposed method ranks fourth on the Football AI Ranking List by RLChina and advances to the top eight in the IEEE Conference on Games 2022 Football AI Competition. In the experimental scenario, the agents must coordinate their time and positioning to organize an attack, seize a brief opportunity, and only receive a reward when scoring a goal. In the experiments, we control all players except the goalkeeper of one side, and the other side player is controlled by the GRF game engine. The agent has a discrete action space of 19 actions, including moving in eight directions, sliding, shooting, and passing. The observation includes the positions and movement directions of the self-agent, other agents, and the ball. The z-coordinate of the ball is also included.

4.2.2 Results and analysis

This article uses MCRL in conjunction with three baseline methods (the self-play method, the league-learning method, and the method of playing against rule-based agents) to compare and assess the winning rate and unshaped rewards (i.e., number of goals) against the easy, medium, and hard built-in AI of the GRF 11v11 stochastic scenario. The four methods had each been trained for 13 million timesteps. Each method is tested three times, with each test consisting of 540 games.

In Figs. 8 and 9, we compare MCRL with the baseline methods. MCRL outperforms the three baseline methods, and the outcomes of the self-play method using the MAPPO algorithm are suboptimal. Even after 13 million iterations, there is no marked improvement in the agent's performance, which we believe is due to the sparsity of rewards. The league-learning method seems to successfully teach the agent a strategy against the entire league at the 8 millionth iteration but does not sustain it in subsequent iterations. The method of playing against rule-based agents requires more time to explore complex strategies. In contrast, the proposed method shows good pertinence to the Football AI Ranking List by RLChina. By distillation of

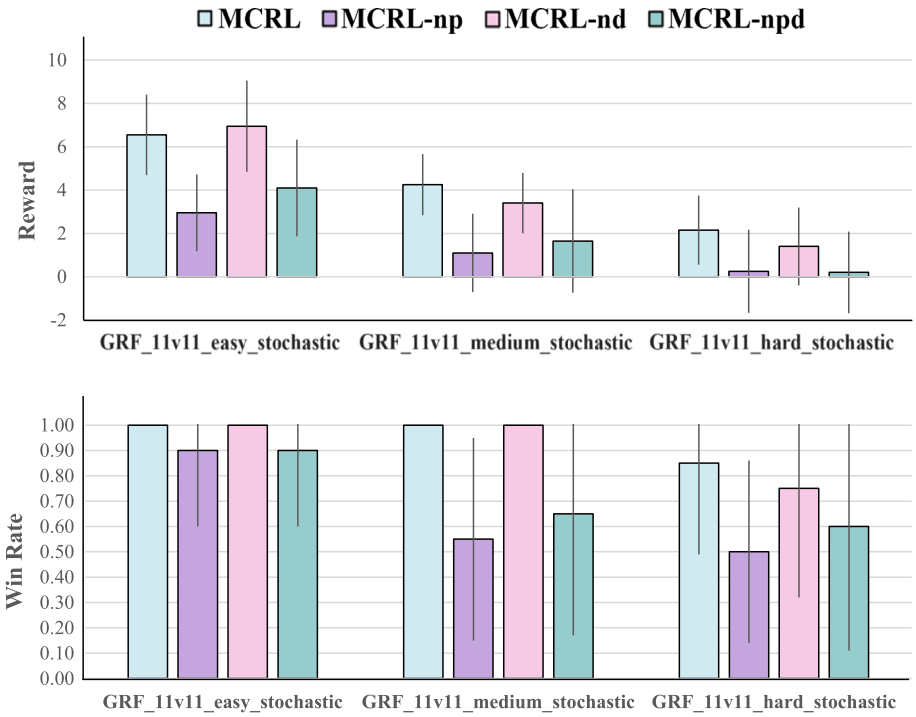

Fig. 10 Ablation studies of the MCRL and its three variants on challenging *GRF_11v11_Stochastic* scenarios against easy, medium, and hard built-in AI. Each evaluation involves a 72-h testing period consisting of 20 game matches, and the average winning rate and the unshaped reward are recorded

each mentors' strategy, the proposed method has learned diverse but coordinated counteracting strategies.

As depicted in Fig. 9, a discernible observation is made regarding the performance of various methods under the GRF 11v11 stochastic scenario. When competing against an easy built-in AI, the MCRL demonstrates a convergence rate analogous to the other three baseline methodologies. This parity in performance, however, deviates in a scenario against a medium built-in AI. Here, both MCRL and self-play achieve convergence at 8 million timesteps. Contrarily, league-learning is ensnared in a local optimum, and versus built-in fails to exhibit convergence. The scenario is further intensified against a hard built-in AI. Within the span of 13 million timesteps, all four methodologies remain non-convergent. Despite this, MCRL stands out, showcasing superior performance in both win rate and accrued rewards, underscoring its convergence efficacy in more challenging contexts.

4.3 Ablation study

MCRL trained diverse but coordinated counteracting strategies for partners trained via the PTL method. In this section, we analyze their effects through an ablation study. From MCRL, we derive four variants, MCRL-npd, MCRL-np, and MCRL-nd. MCRL-npd abandons partners and policy distillation and makes the primary agent directly compete with pre-trained agents of partners. MCRL-np does not employ the second stage, bearing similarity to league-learning, which makes the primary agent directly compete with partners trained via the first stage of MCRL. MCRL-nd abandons the first stage but retains the second stage, where the mentors are trained to counteract pre-trained partner agents. Then, using policy distillation with mentor models, it distills the counter-policies acquired by the mentors into the primary agent. We compare the performance of MCRL with these three variants on two benchmarks, the Football AI Ranking List by RLChina [74] and the challenging 11v11 stochastic scenario of GRF [4]. The evaluation results are reported in Table 2 and Fig. 10.

We first conduct ablation studies on the RLChina AI Ranking List to analyze which of the proposed novelties led to better performance, as shown in Table 1. Ablating each component of the MCRL results in a marked decrease in performance. Among them, the ablation of the second stage, which is policy distillation, has the least impact on performance. MCRL-np performs the worst, indicating that training the primary agent against weaker opponents can actually harm the coordination of the primary agent's own strategy. MCRL-npd shows a promising winning rate, but its average goal difference per game is poor, which affects the final Elo rating, which suggests that although training the primary agent to play against the coach agent can lead to learning coordinated strategies, it may not effectively counter the entire league.

We also conduct ablation studies on the challenging GRF 11v11 stochastic scenario. In the matches against all difficulty levels of built-in AI, MCRL generally outperforms these three variants. In matches against medium and hard difficulty levels of built-in AI, both MCRL-np and MCRL-npd achieve lower average winning rates and goal scores, with large standard deviations in both metrics, indicating that their competitiveness is highly unstable. These results suggest that applying the MCRL can more effectively improve the performance of the primary agent. MCRL-nd achieves promising results; however, these results are still inferior to those of MCRL, which demonstrates the effectiveness of the policy distillation method.

5 Conclusion

In this paper, we propose a novel two stage reinforcement learning algorithm named MCRL. This algorithm aims to counter participating agents in the competition and creates two distinctive agents: sparring partner and primary agent. We applied the MCRL based on the historical game logs of opponents, we encountered during the warm-up session and formulated sparring partner functions similarly to human sparring partners to simulate opponents with diverse styles of policy, enabling primary players to practice against a range of policies, they may encounter in real competitions and thus improve their skills more effectively. Also, we generated multiple counter-policies, distills them, and amalgamates them to form a potent primary agent. Empirical results show that the proposed MCRL algorithm can efficiently search for valuable strategies with stable updates and balance the relationship between policy iteration and policy style deviation by introducing the Wasserstein distance into the distributed PPO algorithm. The proposed primary agent can also learn diverse but coordinated counteracting strategies and ranks in the top eight in the competition.

The MCRL algorithm demonstrates excellent targeting for the league, which could be applied in the future to enhance the league-learning algorithm employed by AlphaStar [3]. During AlphaStar's training, there are three types of opponent pools (main agents, league exploiters, and primary exploiters), with league exploiters being used to compete with the league. We speculate that the MCRL algorithm can train better-performing league exploiters. Therefore, in the future work, we plan to use the proposed approach to improve the league-learning algorithm.

Acknowledgements This study was partially supported by the National Natural Science Foundation of China Youth Fund (62306135), the Ministry of Education Youth Fund (23YJC630156) and the Jiangsu Provincial Youth Fund (BK20230783).

Author contributions Research problem, method design, implementation, and experiments have been performed by JZ. Data analysis has been performed by JZ. JZ, JL, XZ, and YL have made results analysis. JZ, XZ, and YS read and approved the final manuscript.

Code availability The source code and data have been made available at https://github.com/zjj1224073665/football-distillation-SEED.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval This article does not involve human subject for data collection. There is no need for ethical approval.

References

1. Agarwal R, Schuurmans D, Norouzi M (2019) Striving for simplicity in off-policy deep reinforcement learning. CoRR. arXiv:1907.04543
2. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller MA (2013) Playing atari with deep reinforcement learning. CoRR. arXiv:1312.5602
3. Vinyals O, Babuschkin I, Czarnecki W, Mathieu M, Dudzik A, Chung J, Choi D, Powell R, Ewalds T, Georgiev P, Oh J, Horgan D, Kroiss M, Danihelka I, Huang A, Sifre L, Cai T, Agapiou J, Jaderberg M, Silver D (2019) Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature. https://doi.org/10.1038/s41586-019-1724-z
4. Kurach K, Raichuk A, Stanczyk P, Zajac M, Bachem O, Espeholt L, Riquelme C, Vincent D, Michalski M, Bousquet O, Gelly S (2019) Google research football: a novel reinforcement learning environment. CoRR. arXiv:1907.11180
5. Berner C, Brockman G, Chan B, Cheung V, Debiak P, Dennison C, Farhi D, Fischer Q, Hashme S, Hesse C, Józefowicz R, Gray S, Olsson C, Pachocki J, Petrov M, Oliveira Pinto HP, Raiman J, Salimans T, Schlatter J, Schneider J, Sidor S, Sutskever I, Tang J, Wolski F, Zhang S (2019) Dota 2 with large scale deep reinforcement learning. CoRR. arXiv:1912.06680
6. Rashid T, Samvelyan M, Witt CS, Farquhar G, Foerster JN, Whiteson S (2018) QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. CoRR. arXiv:1803.11485
7. Mahajan A, Rashid T, Samvelyan M, Whiteson S (2019) MAVEN: multi-agent variational exploration. CoRR. arXiv:1910.07483
8. Yu C, Velu A, Vinitsky E, Wang Y, Bayen AM, Wu Y (2021) The surprising effectiveness of MAPPO in cooperative, multi-agent games. CoRR. arXiv:2103.01955
9. Taïga AA, Fedus W, Machado MC, Courville AC, Bellemare MG (2021) On bonus-based exploration methods in the arcade learning environment. CoRR. arXiv:2109.11052
10. Zhang T, Xu H, Wang X, Wu Y, Keutzer K, Gonzalez JE. Tian Y (2020) Bebold: Exploration beyond the boundary of explored regions. CoRR. arXiv:2012.08621
11. Zhao R, Song J, Yuan Y, Haifeng H, Gao Y, Wu Y, Sun Z, Wei Y (2022) Maximum entropy population-based training for zero-shot human-AI coordination
12. Kapturowski S, Campos V, Jiang R, Rakićević N, Hasselt H, Blundell C, Badia AP (2022) Human-level Atari 200x faster
13. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap TP, Leach M, Kavukcuoglu K, Graepel T, Hassabis D (2016) Mastering the game of go with deep neural networks and tree search. Nature 529:484–489
14. Cobbe K, Hesse C, Hilton J, Schulman J (2020) Leveraging procedural generation to benchmark reinforcement learning
15. Ye D, Chen G, Zhang W, Chen S, Yuan B, Liu B, Chen J, Liu Z, Qiu F, Yu H, Yin Y, Shi B, Wang L, Shi T, Fu Q, Yang W, Huang L, Liu W (2020) Towards playing full MOBA games with deep reinforcement learning
16. Huang S, Chen W, Zhang L, Li Z, Zhu F, Ye D, Chen T, Zhu J (2021) Tikick: towards playing multi-agent football full games from single-agent demonstrations. CoRR. arXiv:2110.04507
17. Lin F, Huang S, Pearce T, Chen W, Tu W-W (2023) TiZero: mastering multi-agent football with curriculum learning and self-play
18. Liu X, Jia H, Wen Y, Yang Y, Hu Y, Chen Y, Fan C, Hu Z (2021) Unifying behavioral and response diversity for open-ended learning in zero-sum games. CoRR. arXiv:2106.04958
19. Li C, Wu C, Wang T, Yang J, Zhao Q, Zhang C (2021) Celebrating diversity in shared multi-agent reinforcement learning. CoRR. arXiv:2106.02195
20. Yang Y, Wang J (2021) An overview of multi-agent reinforcement learning from game theoretical perspective
21. Kajii Y, Yamada K (2017) Multi-agent reinforcement learning. In: The proceedings of JSME annual conference on robotics and mechatronics (Robomec), pp 2–109
22. Uddin Mondal W, Aggarwal V, Ukkusuri SV (2022) Mean-field approximation of cooperative constrained multi-agent reinforcement learning (CMARL)
23. Galliera R, Venable KB, Bassani M, Suri N (2023) Learning collaborative information dissemination with graph-based multi-agent reinforcement learning
24. Mishra S, Anand A, Hoffmann J, Heess N, Riedmiller M, Abdolmaleki A, Precup D (2023) Policy composition in reinforcement learning via multi-objective policy optimization
25. Maria, Grazia, Vigliotti: decentralized execution of constraint handling rules for ensembles. Comput Rev (2014)
26. Rashid T, Samvelyan M, De Witt CS, Farquhar G, Foerster J, Whiteson S (2018) Qmix: monotonic value function factorisation for deep multi-agent reinforcement learning
27. Schreuder N, Brunel V-E, Dalalyan A (2020) Statistical guarantees for generative models without domination
28. Jaderberg M, Dalibard V, Osindero S, Czarnecki WM, Donahue J, Razavi A, Vinyals O, Green T, Dunning I, Simonyan K, Fernando C, Kavukcuoglu K (2017) Population based training of neural networks
29. Yu H, Zhang X, Song L, Jiang L, Huang X, Chen W, Zhang C, Li J, Yang J, Hu Z, Duan Q, Chen W, He X, Fan J, Jiang W, Zhang L, Qiu C, Gu M, Sun W, Zhang Y, Peng G, Shen W, Fu G (2020) Large-scale gastric cancer screening and localization using multi-task deep neural network
30. Hüllermeier E, Waegeman W (2021) Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. Mach Learn 110(3):457–506. https://doi.org/10.1007/s10994-021-05946-3
31. Gehrig M, Shrestha SB, Mouritzen, D, Scaramuzza D (2020) Event-based angular velocity regression with spiking networks

32. Ge Y, Xu S, Liu S, Fu Z, Sun F, Zhang Y (2020) Learning personalized risk preferences for recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. SIGIR '20. Association for Computing Machinery, New York, pp. 409–418. https://doi.org/10.1145/3397271.3401056
33. Li Q, Huang J, Hu J, Gong S (2022) Feature-distribution perturbation and calibration for generalized person ReID
34. Gampe H, Griffin C (2023) Dynamics of a binary option market with exogenous information and price sensitivity. Commun Nonlinear Sci Numer Simul 118:106994. https://doi.org/10.1016/j.cnsns.2022.106994
35. Liu Z, Li X (2022) A novel Lagrange multiplier approach with relaxation for gradient flows
36. Hester T, Vecerik M, Pietquin O, Lanctot M, Schaul T, Piot B, Horgan D, Quan J, Sendonaris A, Dulac-Arnold G, Osband I, Agapiou J, Leibo JZ, Gruslys A (2017) Deep Q-learning from demonstrations
37. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap TP, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning
38. Wen G, Li B (2022) Optimized leader-follower consensus control using reinforcement learning for a class of second-order nonlinear multiagent systems. IEEE Trans Syst Man Cybern: Syst 52(9):5546–5555. https://doi.org/10.1109/TSMC.2021.3130070
39. Song Z, Ma C, Ding M, Yang HH, Qian Y, Zhou X (2023) Personalized federated deep reinforcement learning-based trajectory optimization for multi-UAV assisted edge computing
40. Cazenavette G, Wang T, Torralba A, Efros AA, Zhu J-Y (2022) Dataset distillation by matching training trajectories
41. Tu V, Pham TL, Dao PN (2022) Disturbance observer-based adaptive reinforcement learning for perturbed uncertain surface vessels. ISA Trans 130:277–292. https://doi.org/10.1016/j.isatra.2022.03.027
42. Du Y, Wu D (2022) Deep reinforcement learning from demonstrations to assist service restoration in islanded microgrids. IEEE Trans Sustain Energy 13:1062–1072
43. Tang Z, Shi Y, Xu X (2023) CSGP: closed-loop safe grasp planning via attention-based deep reinforcement learning from demonstrations. IEEE Robot Autom Lett 8:3158–3165
44. Martins FB, Machado MG, Bassani HF, Braga PHM, Barros ES (2021) rSoccer: A framework for studying reinforcement learning in small and very small size robot soccer
45. Stone P, Sutton RS, Kuhlmann G (2005) Reinforcement learning for robocup soccer keepaway. Adapt Behav 13(3):165–188. https://doi.org/10.1177/105971230501300301
46. Kitano H, Asada M, Kuniyoshi Y, Noda I, Osawa E (1997) Robocup: the robot world cup initiative. In: Proceedings of the first international conference on autonomous agents. AGENTS '97. Association for Computing Machinery, New York, pp 340–347. https://doi.org/10.1145/267658.267738
47. Liu S, Lever G, Wang Z, Merel J, Eslami SMA, Hennes D, Czarnecki WM, Tassa Y, Omidshafiei S, Abdolmaleki A, Siegel NY, Hasenclever L, Marris L, Tunyasuvunakool S, Song HF, Wulfmeier M, Muller P, Haarnoja T, Tracey BD, Tuyls K, Graepel T, Heess N (2021) From motor control to team play in simulated humanoid football
48. Fengming Zhu ZL, Zhu, K (2020) WeKick. https://www.kaggle.com/ c/google-football/discussion/202232
49. Oliehoek FA, Amato C (2016) A concise introduction to decentralized POMDPs, 1st edn. Springer, Berlin
50. Hester T, Vecerik M, Pietquin O, Lanctot M, Schaul T, Piot B, Horgan D, Quan J, Sendonaris A, Dulac-Arnold G, Osband I, Agapiou J, Leibo JZ, Gruslys A (2017) Deep Q-learning from demonstrations
51. Vecerik M, Hester T, Scholz J, Wang F, Pietquin O, Piot B, Heess N, Rothörl T, Lampe T, Riedmiller M (2018) Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards
52. Nair A, McGrew B, Andrychowicz M, Zaremba W, Abbeel P (2018) Overcoming exploration in reinforcement learning with demonstrations
53. Liang X, Wang T, Yang L, Xing E (2018) CIRL: controllable imitative reinforcement learning for vision-based self-driving
54. Fu J, Luo K, Levine S (2018) Learning robust rewards with adversarial inverse reinforcement learning
55. Hausman K, Chebotar Y, Schaal S, Sukhatme G, Lim J (2017) Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets
56. Zhang M, Wang Y, Ma X, Xia L, Yang J, Li Z, Li X (2020) Wasserstein distance guided adversarial imitation learning with reward shape exploration. CoRR. arXiv:2006.03503
57. Weng L (2019) From GAN to WGAN. CoRR. arXiv:1904.08994
58. Panaretos VM, Zemel Y (2019) Statistical aspects of Wasserstein distances. Annu Rev Stat Appl 6(1):405–431. https://doi.org/10.1146/annurev-statistics-030718-104938
59. Xing J, Nagata T, Zou X, Neftci E, Krichmar JL (2023) Achieving efficient interpretability of reinforcement learning via policy distillation and selective input gradient regularization. Neural Netw 161:228–241
60. Xing J, Nagata T, Zou X, Neftci E, Krichmar JL (2022) Policy distillation with selective input gradient regularization for efficient interpretability
61. Rusu AA, Colmenarejo SG, Gülçehre, Desjardins G, Kirkpatrick J, Pascanu R, Mnih V, Kavukcuoglu K, Hadsell R (2015) Policy distillation. CoRR. arXiv:1511.06295
62. Nowozin S, Cseke B, Tomioka R (2016) f-GAN: training generative neural samplers using variational divergence minimization
63. Yu C, Velu A, Vinitsky E, Wang Y, Bayen AM, Wu Y (2021) The surprising effectiveness of MAPPO in cooperative, multi-agent games. CoRR. arXiv:2103.01955
64. Lowe R, Wu Y, Tamar A, Harb J, Abbeel P, Mordatch I (2017) Multi-agent actor-critic for mixed cooperative-competitive environments. CoRR. arXiv:1706.02275
65. Sandler M, Howard AG, Zhu M, Zhmoginov A, Chen L (2018) Inverted residuals and linear bottlenecks: mobile networks for classification, detection and segmentation. CoRR. arXiv:1801.04381
66. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: efficient convolutional neural networks for mobile vision applications. CoRR. arXiv:1704.04861
67. Cho K, Merrienboer B, Gülçehre Ç, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder-decoder for statistical machine translation. CoRR. arXiv:1406.1078
68. Hochreiter S (1997) Long short-term memory. Neural Comput 9:1735–80. https://doi.org/10.1162/neco.1997.9.8.1735
69. Yu X, Li G, Chai C, Tang N (2020) Reinforcement learning with tree-LSTM for join order selection. In: 2020 IEEE 36th international conference on data engineering (ICDE), pp 1297–1308. https://doi.org/10.1109/ICDE48307.2020.00116
70. Saxe AM, McClelland JL, Ganguli S (2013) Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. https://doi.org/10.48550/ARXIV.1312.6120. arXiv:1312.6120
71. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. CoRR. arXiv:1412.6980

72. Espeholt L, Marinier R, Stanczyk P, Wang K, Michalski M (2020) SEED RL: scalable and efficient deep-RL with accelerated central inference
73. Czarnecki WM, Pascanu R, Osindero S, Jayakumar SM, Swirszcz G, Jaderberg M (2019) Distilling policy distillation. CoRR. arXiv:1902.02186
74. Automation CAoS RLChina Reinforcement Learning Community. Institute of Automation, Chinese Academy of Sciences. www.rlchina.org
75. Automation CAoS Jidi. Institute of Automation, Chinese Academy of Sciences. http://www.jidiai.cn/