Artificial Intelligence Review (2025) 58:165
https://doi.org/10.1007/s10462-025-11166-1

Intelligent games meeting with multi-agent deep reinforcement learning: a comprehensive review

Yiqin Wang[1] · Yufeng Wang[1] · Feng Tian[1] · Jianhua Ma[2] · Qun Jin[3]

Accepted: 28 February 2025 / Published online: 15 March 2025

Abstract
Recent years have witnessed the great achievement of the AI-driven intelligent games, such as AlphaStar defeating the human experts, and numerous intelligent games have come into the public view. Essentially, deep reinforcement learning (DRL), especially multiple-agent DRL (MADRL) has empowered a variety of artificial intelligence fields, including intelligent games. However, there is lack of systematical review on their correlations. This article provides a holistic picture on smoothly connecting intelligent games with MADRL from two perspectives: theoretical game concepts for MADRL, and MADRL for intelligent games. From the first perspective, information structure and game environmental features for MADRL algorithms are summarized; and from the second viewpoint, the challenges in intelligent games are investigated, and the existing MADRL solutions are correspondingly explored. Furthermore, the state-of-the-art (SOTA) MADRL algorithms for intelligent games are systematically categorized, especially from the perspective of credit assignment. Moreover, a comprehensively review on notorious benchmarks are conducted to facilitate the design and test of MADRL based intelligent games. Besides, a general procedure of MADRL simulations is offered. Finally, the key challenges in integrating intelligent games with MADRL, and potential future research directions are highlighted. This survey hopes to provide a thoughtful insight of developing intelligent games with the assistance of MADRL solutions and algorithms.

Keywords Intelligent game · Multi-agent deep reinforcement learning · Credit assignment · Communications structure · Game simulations

✉ Yufeng Wang
wfwang1974@gmail.com

Jianhua Ma
jianhua@hosei.ac.jp

Qun Jin
jin@waseda.jp

[1] Nanjing University of Posts and Telecommunications, Nanjing, China

[2] Hosei University, Chiyoda City, Japan

[3] Waseda University, Shinjuku City, Japan

1 Introduction

Recently, game industry has witnessed great deployment. In 2021, the global video game market generated over $175.8 billion in revenue (Chan et al. 2022). In 2024, the revenue from the worldwide gaming market was estimated at almost $455 billion[1]. The global revenue in the 'Games' segment of the media market was forecast to continuously increase between 2024 and 2029 by in total $216.3 billion (+45.53%). After the consecutively increasing for years, the revenue is estimated to reach $691.31 billion and therefore a new peak in 2029[2]. This remarkable expansion is fueled by the thriving e-economy, which has created substantial demand for advancements in game AI. AI technologies are now seen as key drivers for developing immersive and dynamic gaming experiences (Hu et al. 2024).

In response to these rapid advancements, the role of AI in gaming has shifted from merely enhancing game play mechanics to fundamentally redefining game design. This evolution has led to the rise of intelligent games. An intelligent game implies that the roles of game dynamically learn from interactions, respond to players' actions, and adaptively adjust behaviors to pursue game goal. Specially, instead of procedurally enforcing the static game behaviors, adaptive AI agents with human-like behaviors are essential (Zhao et al. 2020; Gero et al. 2020), which facilitate player-game interaction, significantly improve player experience and promote creative game play (Kumar et al. 2025). In literature, there exist numerous AI-enabled intelligent games. According to game characteristics, they can be broadly divided into genres, turn-based strategy games, e.g., *Go* (Silver et al. 2017), *chess* (Silver et al. 2018), *shogi* (Schrittwieser et al. 2020), *Stratego* (Perolat et al. 2022), *Xiangqi* (Li et al. 2023b), *poker* (Moravčík et al. 2017; Zhao et al. 2022a), *DouDiZhu* (Zha et al. 2021; Zhao et al. 2022c; Zhao et al. 2023), First-person shooting (FPS) games, e.g., *ViZDoom* (Li et al. 2023a), Real-time strategy (RTS) games, e.g., *StarCraft* (Vinyals et al. 2019), *Wargame* (Yao et al. 2023), and Multiplayer Online Battle Arena (MOBA) games, e.g. *League of Legends* (do Nascimento Silva and Chaimowicz 2015) and *DOTA2* (Berner et al. 2019), etc.

Among AI technologies for intelligent games, Reinforcement Learning (RL) has witnessed great application for its capability of directly learning appropriate game actions by interacting with the dynamic and uncertain game environment (Ferdous et al. 2022; Souchleris et al. 2023). Furthermore, Deep Reinforcement learning (DRL) combines the strong capability of Deep Neural Networks (DNNs) as functional approximators to deal with high-dimensional data, with the RL's ability of sequential decision-making under uncertainty, and can solve complicated game tasks. Many games typically feature multiple players engaging in cooperative or adversarial interactions. To effectively harness information within such complex environments and learn optimal strategies for each participant, Multi-agent Deep Reinforcement Learning (MADRL) emerges as a promising approach. For example, by employing DRL algorithms for each agent within a multi-agent training framework, AlphaStar (Vinyals et al. 2019) is rated at Grandmaster level in the full game *StarCraft II*, a RTS game renowned for its demand for high-level unit micro-management and strategic decision-making.

[1] https://www.statista.com/topics/868/video-games/#topicOverview.

[2] https://www.statista.com/forecasts/1344668/revenue-video-game-worldwide.

In MADRL, each agent's objective is to maximize its own long-term return or collaboratively optimize a shared goal. A naive way is to independently apply the single agent RL algorithms to multi-agent (MA) systems, totally ignoring coordination among multiple agents, like Independent Q-Learning (IQL) (Tampuu et al. 2017). However, IQL demonstrates poor performance, mainly due to the peculiar characteristics of MA environments. First, commonly, each agent can only observe the partial system state, so-called partial observability. Second, during the learning process, the polices of multiple agents are simultaneously updated, which leads to the non-stationary environment from the perspective of each agent. That is, the trajectory sampled for decision-making depends on all players' policies, instead of relying on each player's policy as in single agent DRL, so-called the non-stationary issue (Papoudakis et al. 2019). So, when each agent maximizes its return, the other agents' state and action should be incorporated as part of its optimization problem, which is fundamentally different from traditional single-agent DRL. In other words, the Markov assumption, which is essential to modeling the decision process in single-agent DRL, no longer holds in MADRL (Gronauer and Diepold 2022). Basically, the Markov property assumes that the future state of the system depends only on the current state and action, and not on previous states or actions, which always holds in single-agent systems where the environment remains constant except for the agent's actions. However, in MADRL settings, due to these two distinguished features mentioned above, i.e., partial observability of each agent, and non-stationary environment, the assumption of a fully observable and static environment is no longer valid in MADRL systems.

In literature, the intersection of MA systems and RL possesses a long record of active research. Several reviews on general MARL algorithms have already existed. For example, MADRL approaches are categorized from the following five aspects: non-stationary, partial observability, multi-agent training schemes, transfer learning in MA system, and continuous state and action spaces in MA learning (Nguyen et al. 2020). A review of cooperative MADRL is provided in (Oroojlooy and Hajinezhad 2023) under five categories: independent learners, fully observable Critic, value function factorization, consensus, and learn to communicate. The basic methods and application scenarios of MARL is summarized in (Zhou et al. 2023b), and their limitations are discussed, including safety, robustness, generalization and ethical constraints. Canese et al. (2021) preliminarily groups MARL algorithms according to their features. A monograph overviews MADRL from game theoretical perspective (Yang and Wang 2020). Gronauer and Diepold (2022) overviews of the current MADRL developments from the following three aspects: the MADRL training structure, the emergent patterns of agent behavior in various scenarios, and MA challenges and the corresponding methods to cope with these challenges. Hao et al. (2023) reviews the exploration methods in deep reinforcement learning from single-agent to multiagent domain. Although these works have well studied the challenges and state-of-the-art (SOTA) algorithms in MADRL, less sight is put on connecting intelligent games with MADRL.

Besides, there exist several surveys about intelligent games in literature. A systematic mapping is studied on various methods of game designing (Barambones et al. 2022), however it doesn't involve the MADRL methods for intelligent game. Lanctot et al. (2019) investigates the environments and games in reinforcement learning, including MA scenarios. However, this work lacks of systematic analysis of the relations between the MADRL and intelligent games. By exploring the intrinsic correlation between intelligent games and MADRL, not only can it fuel the development in intelligent game industry, but the new

emerging benchmarks can further be explored, which thus improve the existing MADRL algorithms.

To our best knowledge, till now, there exists no a comprehensive review smoothly connecting MADRL with intelligent games. To fill the gap, this article provides a holistic picture on MADRL and intelligent games through explicitly integrating MADRL and intelligent games from two perspectives: theoretical game concepts for MADRL, and MADRL for intelligent games, and correspondingly investigate their challenges and features respectively. In detail, the contributions of this article are given as follows.

- From viewpoint of theoretical game for MADRL, information structure and game environmental features are summarized; from the viewpoint of MADRL for intelligent games, typical challenges in intelligent games and corresponding MADRL solutions are systematically presented.
- The state-of-the-art MADRL algorithms for intelligent games are systematically categorized, especially from the viewpoint of credit assignment.
- The simulation environments and benchmarks for MADRL based intelligent games are summarized. Especially, the general procedure for simulating MADRL based intelligent games is presented.
- The open issues and potential research are highlighted for future research and applications.

Figure 1 illustrates the schematic structure of the main contents and their relations in this article. In detail, MADRL features from the viewpoint of theoretical games are discussed in Sect. 2, including information structure and game environmental features. Section 3 presents the challenges in intelligent games, i.e., heterogeneous agents, partial observability, non-stationary, scalability, etc., and their MADRL solutions, including credit assignment, communication structure, training paradigm, and scalability-enabling technologies. The detailed MADRL algorithms for intelligent games are categorized and summarized in Sect. 4. Section 5 presents the simulation environments and benchmarks in the literature. Section 6 describes several typical MADRL-based intelligent game apps. Some open issues are pointed out in Sect. 7. Finally, for the integrity of contents, typical DRL frameworks and algorithms are presented in the Appendix, based on which MADRL algorithms for intelligent games can be designed through adapting MADRL solutions presented in this article.

2 Viewpoint of theoretic game for MADRL

In this section, the concept and features of MADRL are summarized from two aspects: information structure and game environmental characteristics. Correspondingly, the theoretical framework of MADRL can be modeled as Markov/stochastic game (MG) and Partially-observable Markov game (POMG).

2.1 Information structure

From a game-theoretical perspective, depending on the integrity of the information acquired by agents from the environment, the information structure, i.e., who knows what, can be cat-

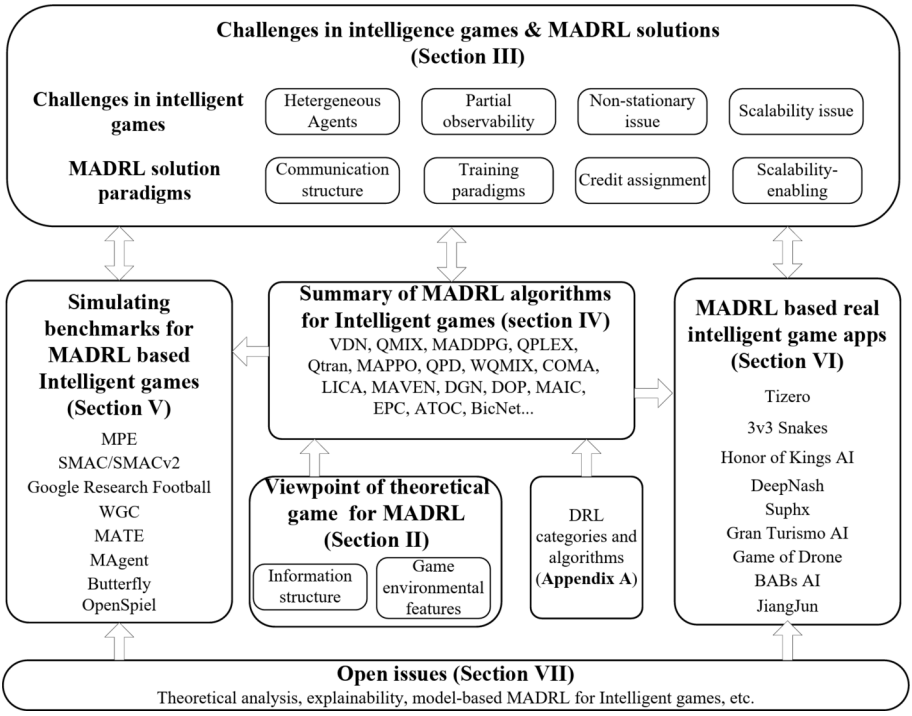

Fig. 1 Schematic structure of main contents and their relations in this review

egorized as perfect, imperfect, or incomplete. A perfect information game can be defined as one where there is only one history per state that can be fully observed by all agents. In such games, all players have complete knowledge of the current state, including all actions taken by others. In contrast, an imperfect information game implies that there generally exist multiple histories per state, which are observed differently by a variety of agents. While agents may share some knowledge of the state, certain elements (e.g., actions taken by other agents or hidden variables) are not observable, leading to uncertainty in the game. Additionally, incomplete information game refers to situations where players lack knowledge about the overall game structure, such as the payoffs, strategies, or preferences of other players. This represents a deeper level of uncertainty compared to imperfect information, involving not just missing details of the current state but also ignorance about the full range of possible strategies or game dynamics.

Note that, in the context of MADRL, especially in model-free MADRL, the 'state' should encompass all information available to agents used to make decisions (i.e., internal state of agents and external state of environment (Li et al. 2022), without assuming to know the intrinsic game structure/dynamics. To simplify the description of information structures in game-theoretical framework and align it more closely with model-free MADRL scenarios, it is reasonable to focus on perfect information games, where both environment states and actions of each agent are fully observable for all agents, and imperfect information games, where the necessary information for agents to make decisions—such as environment states

or other agents' strategies—is imperfect. Moreover, explicitly modeling the game structure is essentially treated in model-based MADRL for intelligent games in Sect. 7 "Open issues".

2.1.1 Perfect information structure and Markov game

In MA system, the perfect information structure intuitively means that agents can observe all information of the environment, including the historical selections of other agents, current states information, etc. (Mycielski 1992). A typical case is AlphaGo (Silver et al. 2016), in which the serial actions of each player can be fully observed. Games with perfect information structure usually are modeled as a stochastic game, also known as a Markov game (MG) (Shapley 1953).

MG can be defined by a tuple $\{N, \boldsymbol{S}, \boldsymbol{A}, P, r_i^t, \gamma\}$, in which N represents the number of agents (i.e., game players), when N=1, MG degenerates to a MDP. $\boldsymbol{S}$ represents the set of environment states shared by all agents. $\boldsymbol{A}_i$, $(i = 1, 2, 3, \ldots, N)$ denotes the action space of agent i, and thus joint action set of all agents is defined by $\boldsymbol{A} = \boldsymbol{A}_1 \times \boldsymbol{A}_2 \times \boldsymbol{A}_3 \times \ldots \times \boldsymbol{A}_N$. At each time step, all agents take actions and the states transfer from state s to s', and the state transition probability function is defined by $P : S \times A \times S \rightarrow [0,1]$. r_i^t represents the immediate reward received by agent i at timestep t, which maps state and joint actions to a real number. $\gamma \in [0,1]$ is the discount factor, which defines the present value of future rewards. The goal of MG is for agents either to learn an optimal policy maximizing the team's long-term return in cooperative game environment, or to maximize their own long-term return, irrespective of their opponents' return in a competitive game environment, i.e., $R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$.

2.1.2 Imperfect information structure and partially-observable Markov game (POMG)

Markov games can only handle the fully observed states. However, a plethora of MARL applications involve agents with only partial observability, in which agents neither observe the global state of the environment, nor do they have access to the internal knowledge of other agents, so-called imperfect information structure, as seen in games like *poker*, *Mahjong*, and *StarCraft*, etc.

Games with the imperfect information structure can be characterized by Partially-Observable Markov games (POMGs) (Liu et al. 2022; Hansen et al. 2004). A POMG is characterized by the set $\{N, \boldsymbol{S}, \boldsymbol{A}, P, r_i^t, \gamma, \boldsymbol{O}_i, \boldsymbol{O}\}$. Compared with MG, the newly added elements are $\boldsymbol{O}_i$ and $\boldsymbol{O}$, which respectively represent the observation of agent i, and the joint observations $\boldsymbol{O} = \boldsymbol{O}_1 \times \boldsymbol{O}_2 \times \boldsymbol{O}_3 \times \ldots \times \boldsymbol{O}_N$. Another theoretical framework for multi-agent decision-making under imperfect information is Extensive-Form Games (EFGs) (Roth and Erev 1995; Heinrich et al. 2015). Rooted in computational game theory, EFGs provide a flexible and rigorous framework to model sequential interactions and imperfect information. EFGs capture imperfect information by using information sets, where agents cannot distinguish between histories that belong to the same set. This representation explicitly encodes the strategic uncertainty arising from hidden states or unobservable actions of other agents. EFGs model decision-making processes as a tree, where nodes represent decision points, edges represent actions, and terminal nodes are associated with payoffs. The sequential structure allows EFGs to naturally describe scenarios where decision-making is

influenced by the order of actions and historical dependencies. Furthermore, the concept of information sets enables EFGs to accurately represent multi-stage interactions with hidden or private information, such as in poker or auctions.

Note that POMG and extensive-form game are essentially equivalent under slight constraints. Specifically, in simultaneous-move Markov games, an agent's actions are unknown to others, which thus leads to different histories that can be aggregated as one information state s. Histories in these games are sequences of joint actions, andthe discounted accumulated reward instantiates the utility at the end of the game. Conversely, by simply setting the possible actions that agents i can take as empty at the state s, when the agent i doesn't take action at each history at the state s, the extensive-form game reduces to a Markov game with state dependent action space (Yang and Wang 2020; Lanctot et al. 2019; Zhang et al. 2021a).

Note that in scenarios where only partial observations are available, a single observation fails to capture all pertinent information about the environment and its history. Consequently, the Markov property is not satisfied, so-called the non-Markovian environment. A natural way to deal with non-Markovian environments is through information exchange between the agents and using memory mechanism. For example, using deep recurrent neural network in MADRL framework can endow agents with a memory mechanism to store and embed multiple-step historical information.

2.2 Game environmental features

Orthogonal to the information structure of MADRL, from the perspective of game environmental features, MADRL algorithms should work with several environments: fully cooperative, fully competitive/non-cooperative, and mixture of both.

To have a panoramic view of these settings, a generalized form is provided to customize different specific game environments, shown as $R = f(R_1, R_2, R_3 \ldots, R_N)$, where R is the team's return, R_i represents the return of the agent *i*, and $f(\,)$ denotes that the relationship between individual return and team return. For cooperative setting, each agent works collaboratively to maximize the team return, for example, simply let $R = R_1 = \ldots = R_N$. While in competitive settings, it comes to constant-sum (k-sum) game, where the sum of each agents' return is restricted to a constant value, i.e., $\sum_{i \in N} R_i = k$. Specially, for number of agents N= 2, the zero-sum game happens, when $k = 0$, which explicitly describes the fully competitive relationship between the two agents. The mixture of the two can be denoted as general-sum game.

2.2.1 Cooperative settings

As opposed to competing with others, agents in a cooperative game aim to collaboratively solve a task or maximize the global payoff (also known as the global reward). It can be further modeled as fully cooperative games (team games), team-average reward games and Markov potential games (MPGs), etc. (Ding et al. 2022; Leonardos et al. 2022). Fully cooperative games mean that agents are assumed to be homogeneous and inter-changeable. That is, technically, they share the same reward function $r_1 = r_2 = \ldots = r_N$. In the team-average reward games, agents can have different reward functions, but they share the same objective, such as average return $R = \frac{1}{N}\sum_{i=1}^{N} R_i$. The average reward model allows

more heterogeneity among agents, can preserve privacy among agents, and facilitates the development of decentralized MADRL algorithms. MPG is a more general cooperative framework, where a certain potential function is shared by all agents, such that if any agent unilaterally changes its policy, the change in its reward equals (or is proportional to) that in the potential function.

Instead of the impractical centralized approaches, where agents determine actions based on global information, partial observation and communication constraints necessitate the learning of decentralized policies, which conditions only on the local action-observation history of each agent.

In a cooperative game, usually, players form a team/coalition to jointly seek a shared goal. The shared team reward is the common implementation. However, this approach may not accurately account for each agent's contribution (Wolpert and Tumer 2001), which motivates the study of the local reward approach that distributes the global reward to agents according to their contributions, so-called multi-agent credit assignment problem, i.e., the problem that works out each agent's contribution to the team's success, and accordingly assigns each agent the proportional reward generated by joint actions in cooperative settings.

Shapley value (Shapley 1953) is one of the most popular methods to fairly distribute the team's payoff to each agent through considering the extent to which the agent increases the marginal contributions of the coalitions in all possible permutations when it joins in.

Formally, for a game with N agents involved and the team return R, for any $C \subseteq N \setminus \{i\}$, the Shapley value of the i-th agent can be defined as Eq. (1).

$$Sh_i = \sum_{C \subseteq N \setminus \{i\}} \frac{|C|!\,(|N| - |C| - 1)!}{|N|!} \left(R\left(C \cup \{i\}\right) - R\left(C\right)\right) \tag{1}$$

Where $R\left(C \cup \{i\}\right) - R\left(C\right)$ denotes the marginal contribution of agent i to the specific coalition C. Theoretically, Shapley value uniquely provides an equitable assignment of values to individual agents with the following desirable properties: the null player, efficiency, symmetry, and linearity. However, computation of Shapley value of each agent requires computing all the possible marginal contributions, which grows factorially as the number of agents, and is not tractable for real world complex games. In literature, there exist many approximation methods to estimate the Shapley value, including truncated Monte Carlo and Gradient Shapley, among others (Wang et al. 2020a, 2022b; Chen et al. 2023).

2.2.2 Competitive/non-cooperative settings

Fully competitive setting in MARL is typically modeled as zero-sum Markov game. To ease algorithm analysis and computational tractability, most literature investigates the case of two fully competitive agents, where clearly the reward of one agent is exactly the loss of the other.

For MG in non-cooperative settings, the set of best-response policies for agent i is defined as Eq. (2), that is, the best response policy for player i is a policy π_i that maximizes the return of player i against the other players' policies (π_{-i}).

$$BR_i\left(\pi_{-i}\right) = \arg\max_{\pi_i} R_i\left(\pi_i, \pi_{-i}\right) \tag{2}$$

Based on the best-response, Nash equilibrium (NE) is one of the most common solutions. Since the optimal performance of each agent i depends on not only its own policy, but the choices of all other players of the game, ϵ-Nash equilibrium (ϵ-NE) of joint policy π^*, is defined as Eq. (3).

$$\forall \pi_i \in \Pi_i, R_i\left(\pi_i^*, \pi_{-i}^*\right) \geq R_i\left(\pi_i, \pi_{-i}^*\right) - \epsilon \tag{3}$$

π_i^* is the policy of agent i in joint policy π^*, $R_i(\cdot)$ is the expected long-term return of agent i and Π_i is the set of all possible strategies for agent i. When $\epsilon = 0$, π^*constitutes a Nash equilibrium.

For a MG, one stronger version of the NE is called the Markov perfect NE. By Markovian it implies the Nash policies are measurable with respect to a particular partition of possible histories (usually referring to the last state). The word "perfect" means that the equilibrium is also subgame-perfect regardless of the starting state.

A joint policy π is called Pareto-optimal, if no agent can increase its expected return without lessening the expected return of another agent. Although Pareto-optimal equilibrium yields the highest returns for all agents, practically, the risk-averse equilibrium is commonly chosen, since there is intrinsic uncertainty regarding the other agent's actions. By risk it refers to the possibility of inadvertently selecting a joint action that may incur penalties. Note that Pareto-optimal concept commonly works well in no-conflict games, in which all agents have the same set of most preferred outcomes, but it has limited significance in competitive tasks or zero-sum games (Christianos et al. 2023).

Learning in intelligent competitive and mixed games can be often characterized as training a team of agents to beat a fixed set of opponents. However, the dual task of intentionally generating useful opponents to train and evaluate against them, is under-studied. Counterfactual regret minimization (CFR) has a convergence guarantee to a NE in 2-player zero-sum games, but it usually needs domain-specific abstractions to deal with large-scale games (Fu et al. 2021). Another typical example is the self-play (Jaderberg et al. 2019; Hernandez et al. 2021; Bai and Jin 2020; Hernandez et al. 2019), which are able to solve a series of increasingly difficult problems by playing with models used before. Based on this idea, different forms of self-play are proposed (Balduzzi et al. 2019).

2.2.3 Mixed settings

In the cooperative setting, agents collaborate to optimize a long-term return, and in the competitive setting, the summed return of agents usually equals zero. While, the mixed setting involves both cooperative and competitive agents, with general-sum return, so-called general-sum game (Bai et al. 2021).

Such settings often entail a combination of cooperative and competitive tasks (Zhang et al. 2021b), such as simulation of a football game, where players in the same team collaborate with each other to compete with another team.

Equilibrium solution concepts in game theory, such as NE, can only be found for very simple settings. In more complex general-sum game situations, finding NE strategies is typically rather difficult. Zhang et al. (2023a, b) introduces the concept of Stackelberg equilibrium (SE) (Von Stackelberg 2010), in which agents make decisions in a leader-follower framework. Leaders prioritize decision-making and enforce their policies on followers who

then respond rationally. Research has shown that SE is a superior convergence objective for MARL compared to NE (Zhang et al. 2020a).

3 Challenges in intelligent games and corresponding MADRL solutions

As shown in Fig. 2, typical challenges in intelligent games are summarized, and the corresponding MADRL solutions are categorized. The **S1** to **S4** represents corresponding solutions enabled by MADRL, and **C1** to **C4** denotes challenges in intelligent games respectively. It is important to note that the solutions and challenges in multi-agent systems are not strictly one-to-one correspondences but rather complex and intertwined relationships. For example, an appropriate training paradigm (**S3**) and effective communication mechanisms (**S1**) can both help address the challenge of partial observability (**C2**). At the same time, the training paradigm (**S3**) can also mitigate issues related to non-stationarity (**C3**). To emphasize the connections between solutions and challenges, any potential correspondences are explicitly remarked where applicable.

3.1 Challenges in intelligent games

Intelligent games usually face several intrinsic challenges arising from the complexity of multi-agent interactions and the dynamic nature of game environments. These challenges stem from factors such as the diversity of agents, the limited observability of the environment, the dynamic adaptation of agents during training, and the exponential growth in computational complexity as the number of agents increases. This section provides an in-depth exploration of four major challenges: the heterogeneity of agents, partial observability, non-stationary dynamics, and scalability issues. Based on the four major challenges, intelligent games of different types typically exhibit their dominant and distinctive features. For instance, in turn-based games, core challenges like heterogeneous agents (e.g., competitive dynamics with asymmetric roles in Mahjong's multi-player setting) and scalability issues (e.g., vast state spaces in *Go*) dominate, while RTS games, such as *StarCraft*, prioritize partial observation (fog-of-war) and non-stationary issues (dynamic multi-agent interactions). Extending this analysis, FPS games emphasize partial observation (limited first-person perspective and sensory occlusion) and non-stationary issues (rapid adversarial tactics), whereas MOBA games like *DOTA2* predominantly face heterogeneous agents (diverse hero abilities requiring synergistic teamplay) and scalability issues (high-dimensional state-

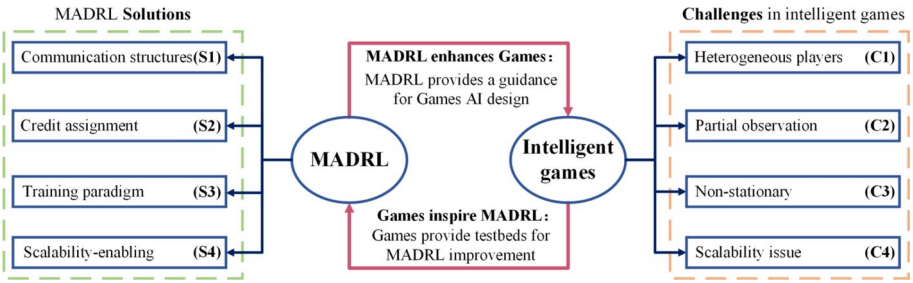

Fig. 2 Interactive relationships between MADRL solutions and intelligent games

action spaces with numerous entities and objectives). This genre-specific challenge decomposition highlights the need for tailored MADRL solutions.

3.1.1 Heterogeneous agents (C1)

Basically, the *Homogeneous agents* implies a group of agents sharing the same policy, that is $\pi_i = \pi, \boldsymbol{\pi} = (\pi, \ldots, \pi)$. Multiple homogeneous agents naturally enable a straightforward adoption of numerous single-agent DRL frameworks for game play decision, without introducing much computational and sample complexity as the increasing number of agents. Team homogeneity allows for parameter sharing among actors as well as simpler network architectures, which leads to faster and more stable training. However, sharing one policy across all agents prevents agents from learning different skills, and is harmful and dangerous in most intelligent games. A more ambitious approach to MADRL is to allow for heterogeneity of policies among agents, which means π_i may have different functions with π_j when $i \neq j$.In heterogeneous MA tasks, different types of agents vary in ability, quantity, and expected functionality, etc. (Fu et al. 2023).

It is unreasonable to assess the contribution of each heterogeneous agent using a uniform criterion (Jiang et al. 2023). Given a joint reward, an individual agent may not be able to distill its own contribution to it, a problem known as *credit assignment*. For example, in *StarCraft* (Whiteson et al. 2019), players can control a wide variety of unit types, each with distinct roles and capabilities. For instance, in the Terran faction, common units include Marines, which serve as basic infantry soldiers, and Medivacs, which function as aerial units capable of transporting troops and providing medical support. All marines may share the same policy while the Medivac's policy can be greatly different from that of marine. Besides, it is challenging to efficiently coordinate different types of agents to achieve a common goal. Communication mechanism (**S1**) may be a key to bridge various agents. A communication infrastructure, such as graph neural networks, can employ information regarding the different types of agents or environment entities to model specialized communication mechanisms. Besides, credit assignment (**S2**) that attributes credit to different agents according to their contributions, also has the potential to enable heterogeneous behaviors of agents.

3.1.2 Partial observability (C2)

In complex intelligent games, it is impractical to assume all agents can fully observe the environment and get complete information of states. For example, the units in *StarCraft* can only see the enemy in a limited observable range (Amos-Binks and Weber 2023; Huang 2023). As a result, the state information available to each agent when making decisions is incomplete, and has to rely on local observations. However, sharing states and actions among agents may mitigate the issue, but will result in significant communication overhead (Zhang and Zavlanos 2023). Therefore, the challenge lies in designing a reasonable and efficient architecture, including communication infrastructure (**S1**) or training and executing paradigms (**S3**), to acquire the necessary information that aids agents during training.

3.1.3 Non-stationary issue (C3)

In single-agent DRL paradigm, the agent is the only decision-maker that affects the environment's state, therefore state transitions can be clearly attributed to the agent. In other words, even though the environment may be stochastic, the learning problem remains stationary, since everything outside the agent's field of impact is viewed as the underlying system dynamic. On the contrary, one of the fundamental problems in the MA domain is that agents simultaneously update their policies during the learning process, such that the environment becomes non-stationary from the perspective of a single agent. Thus, naively applying independent DRL algorithms into multi-agent intelligent games always leads to poor performance. A key to address the issue is to reasonably leverage external state information of an agent, including other agent observation, interaction messages and so on, by well-designed communication mechanisms (**S1**).

Hernandez-Leal et al. (2017) summarizes five approach categories to cope with this non-stationary issue in MARL, i.e., ignore, forget, respond to target models, learn models, and theory of mind.

Orthogonally, from the viewpoint of training MADRL models, to tackle the non-stationary issue, the centralized critic training decentralized execution (CTDE) paradigm **(S3)** has great significance, where the training of critics is centralized through accessing to all agents' observations and actions, while the actors' training is decentralized. Under this architecture, since agents do not experience unexpected changes in the dynamics of the environment, the training procedure and obtained results can be stabilized. Besides, through communication, agents can exchange their observations, actions and intentions to further stabilize the model training (Papoudakis et al. 2019).

3.1.4 Scalability issue (C4)

In multiple agents based intelligent games, each agent needs to consider the joint action space. However, the scale of the joint action will increase exponentially with the increase of the agent, which causes the problem of scalability. Irrespective of either competitive or cooperative games, MADRL based systems suffers from the combinatorial nature (Cui et al. 2022). Additionally, some specific environments require agents to learn a policy to solve multiple tasks with arbitrary numbers of agents, which also incurs the scalability issue (Nayak et al. 2023). There exist scalable-enabling techniques **(S4)** to address this issue, which are briefly introduced in subsection 3.2.4.

3.2 MADRL solutions

This subsection explores key aspects of MADRL solutions, focusing on four core components: communication structure, training and execution paradigms, credit assignment methods and scalability-enabling technologies. Communication structures help alleviate partial observability by enabling information exchange between agents, while training and execution paradigms tackle the non-stationarity challenge, balancing adaptability and stability. Credit assignment methods ensure that each agent's contribution is appropriately evaluated, addressing the issue of differentiating agent behaviors in multi-agent systems. Finally, scalability-enabling technologies ensure that MADRL can manage the exponential growth

of complexity as the number of agents increases, making it feasible to handle large-scale, dynamic game environments. Together, these solutions provide comprehensive approaches to overcoming the challenges inherent in multi-agent interactions within complex games.

3.2.1 Communication structure (S1)

In MADRL tasks, partial observability is inherent as agents are distributed in the environments. Communication among agents is essential for better environmental understanding and coordinated behaviors. In Zhu et al. (2022), a comprehensive survey is conducted on MARL with communication (Comm-MARL) and considers various aspects of communication methods.

The research on learning to communicate have witnessed great development, since many MADRL issues can be alleviated by incorporating communication structure into agents, including partial observability (C2), non-stationary (C3), and coherent coordination among agents.

In intelligent games, agents should decide what, how, and with whom to communicate. As Fig. 3 shows, communication structure in MADRL can be described from the following four factors: what to communicate, with whom to communicate and how to communicate, and communication constraints. "*What to communicate*" includes local observations and historical information; "*with whom to communicate*" includes full coverage, namely all agents, and partial agents. Essentially, the communication object may influence the format of the communication message. Furthermore, different communication objects entail various communication methods, such as broadcast, centralized proxy, predefined structure, and

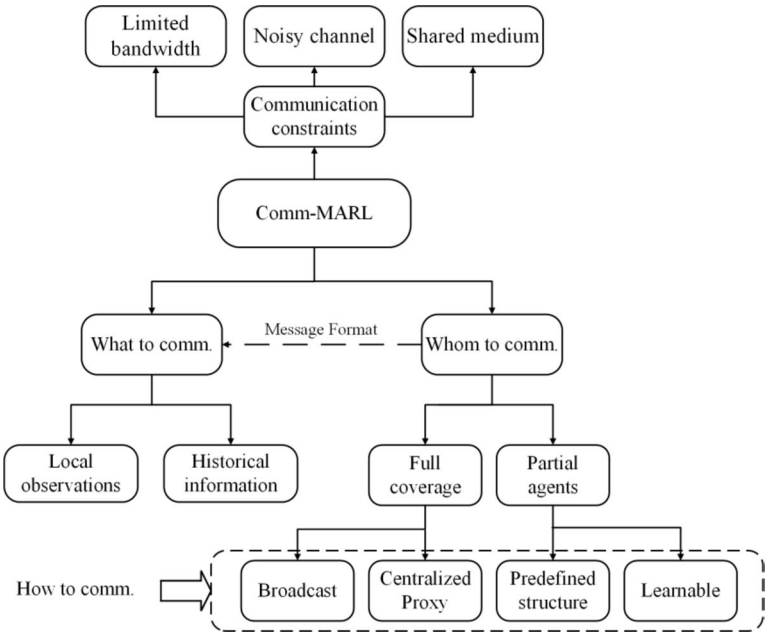

Fig. 3 Communication structure in MADRL

learnable methods, which are denoted as "*how to communicate*". The main factors "*Communication constraints*" include limited bandwidth, noisy channel, and shared medium, etc.

- What to communicate

Due to the fact that most MADRL scenarios are characterized as partial observability, that is, the agent can only receive the local observation. Besides, the historical information, such as the past actions, states, observations or trajectories, should be leveraged for optimal decision-making. Thus, the content of communication typically comprises local observations and historical information.

The local observation commonly denotes the encoding of information observed by existing agents (Kim et al. 2019; Das et al. 2019; Lin et al. 2021). If a central proxy exists, local observations are encoded or sent directly to the proxy, which then generates a single message for all agents, or an individual message for each agent (Niu et al. 2021; Wang et al. 2020c; Liu et al. 2020). Without a proxy, the information will be sent directly to each agent. Bidirectionally-Coordinated Nets (BiCNet) (Peng et al. 2017) leverages bi-directional recurrent structure to support that each agent can maintain its own states and share information with others as well. An attentional communication model called ATOC (Jiang and Lu 2018), takes the local observation as inputs of policy network and extracts thought, which is considered a communication message encoding observation and intend action. Transformer-based Email Mechanism (TEM) (Guo et al. 2023) takes the local observation of each agent, and merges them to facilitate communication.

The historical information incorporates the past local action-observation history, past memories or trajectories. Differentiable Inter-Agent Learning (DIAL) (Foerster et al. 2016) concatenates and encodes past local observations, actions, and local observations to as message. Multi-Agent Incentive Communication (MAIC) (Yuan et al. 2022) takes the local information history of each agent and team representations to generate tailored messages for different agents. Structured Attentive Reasoning Network (SARNet) (Rangwala and Williams 2020) formulates the memory unit, which incorporates local information history, or past memories, to predict actions. In Intention Sharing (IS) (Kim et al. 2020), each agent compresses the current trajectory with imagined trajectory emulating its future action plan, to generate the intention communication message, and based on the received message from other agents, utilizes an attention mechanism to learn the relative importance of the components in the imagined trajectory.

In environments where agents require real-time coordination, sharing immediate local observations is highly recommended, as seen in real-time games. Conversely, in environments with delayed rewards or tasks necessitating long-term planning, such as strategy games, leveraging historical information for communication is more suitable.

- With whom to communicate

Regarding with whom each agent to communicate, i.e., target of communication, full coverage and partial agents are two common choices.

Full coverage means each agent's communication message is received by all agents either through central proxy or diffusion. For example, DIAL (Foerster et al. 2016) and Communication Network (CommNet) (Sukhbaatar et al. 2016) learn a communication pro-

tocol which connects all agents together. Targeted Multi-Agent Communication architecture (TarMAC) (Das et al. 2019) uses a broadcast way to share messages. Heterogeneous Agents Mastering Messaging to Enhance Reinforcement learning (HAMMER) (Gupta et al. 2023) adopts a centralized proxy to receive information and transmit messages, connecting all agents in the MAS.

The way of partial agents means each agent can selectively communicate with partial agents instead of all agents. For example, Graph Attention Exchange Network (GAXNet) (Yun et al. 2021) and graph convolutional reinforcement learning (DGN) (Jiang et al. 2020) allow communication within a certain number of nearby agents. A neural communication protocol called NeurComm (Chu et al. 2020) is built for networked multi-agent systems, which keeps the number of communicating agents fixed during training. ATOC (Jiang and Lu 2018) allows the communication in a group way, which shares coordinated messages with all members. TEM (Guo et al. 2023) considers the agents in the observation range, such as, the nearest several fixed number of agents.

Full coverage method is suitable for highly cooperative tasks where all agents must coordinate closely, such as swarm control or robotic soccer. While partial agents are more effective in scenarios where localized coordination suffices, such as patrolling or decentralized tasks.

- How to communicate

As Fig. 3 shows, the communication way i.e., how to communicate is related to whom to communicate with. For full coverage situation, it may include broadcast and centralized proxy as discussed in the full coverage part. For communication with partial agents, it can also adopt a proxy to assist communication, either using a predefined structure, or a learnable structure. The former method is preferable in scenarios with stable relationships or known topologies, such as static formations or tasks with predefined roles, while the latter is ideal for dynamic environments where communication patterns cannot be predefined, such as unpredictable or evolving tasks.

Predefined structure

The relations between agents can also be captured by a predefined graph, such as Agent-Entity Graph (Agarwal et al. 2020). Additionally, DGN constructs a map including all agents and the set of neighbors, which only allows the communication with neighbor node. Deep Hierarchical Communication Graph (DHCG) (Liu et al. 2023) similarly predefines a hierarchical communication graph which indicates the communication among connected agents. GAXNet (Yun et al. 2021) labels agents who are observable and enables communication between them.

Learnable structure

Learnable structure dynamically learns a communication policy by individual agents or a central proxy, which offer more flexibility. In the learnable approaches, the way that each agent independently determines whom to communicate with, is known as individual control

method. For example, ATOC (Jiang and Lu 2018) introduces a learnable gate mechanism for agents to decide whether to broadcast their messages in a probabilistic way.

The global control used by proxy is to learn proper communication policy for all agents. For example, a deep MARL framework with scheduled communications called SchedNet (Kim et al. 2019), learns a global scheduler controlling which agents to broadcast their messages to reduce the cost. Individually Inferred Communication (I2C) (Ding et al. 2020) trains a prior network to quantify the causal effect between agents to determine agent-agent communication.

- Communication constraints

The communication structure, as an additional component in MADRL architecture, indispensably utilizes certain resources of agents. Therefore, consideration of communication constraints is essential in practical implementation. In general, the constraints exist in three parts: limited bandwidth, noisy channels and shared mediums. Communication messages should be processed into more concise to reduce communication overhead. For example, Variance Based Control (VBC) (Zhang et al. 2019) and Temporal Message Control (TMC) (Zhang et al. 2020b) adopt a predefined threshold to filter out unnecessary information. The noise in the environment or the communication channel (Tung et al. 2021) can affect message transmission. Thus, how to deal with the noise is an open problem. Finally, since messages may be propagated over a shared medium (channel), contention may arise. SchedNet (Kim et al. 2019) prioritizes those agents with high weights to broadcast messages first.

3.2.2 Training and executing paradigm (S2)

As Fig. 4 shows, based on the information used in training and execution, the training and execution paradigm of MADRL can be divided into three paradigms, Centralized Training Centralized Execution (CTCE), Decentralized Training Decentralized Execution (DTDE) and Centralized Training with Decentralized Execution (CTDE).

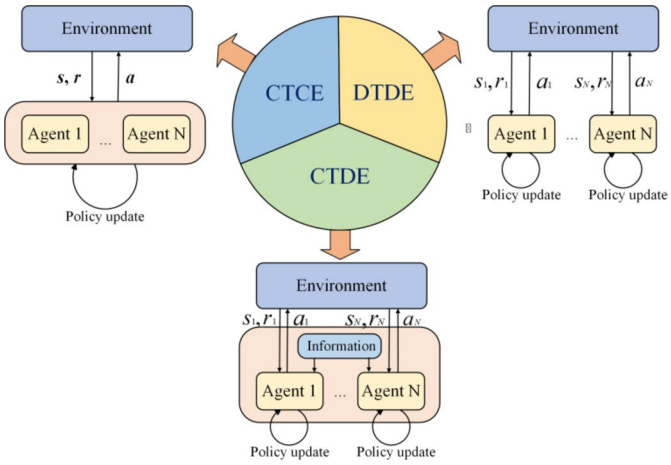

Fig. 4 MADRL training paradigms

- Centralized training entails policies being updated based on mutual information exchange during training. In contrast, decentralized training implies that each agent independently updates its policy without utilizing information from other agents.
- Centralized execution means that agents are controlled by a centralized unit who computes joint actions for all agents. On the contrary, decentralized execution means that agents determine actions according to their individual policy.

CTCE

CTCE uses as input a globally joint observation and outputs a joint action selection for all agents to execute, based on the assumption that unconstrained and instantaneous information exchange can be available among agents. CTCE can be considered as a simple extension of single-agent DRL methods to MA scenarios. However, the state-action space tends to grow exponentially, as the number of agents increases, that is, causing scalability issue. Some early work, such as BiCNet (Peng et al. 2017) and CommNet (Sukhbaatar et al. 2016) etc., adopt CTCE framework to train agents, demonstrating limitations when dealing with a large number of agents. Thus, CTCE is more suitable for small-scale scenarios, where the training and execution of all agents can be centrally managed with relatively low computational cost.

DTDE

DTDE features as each agent selecting actions individually and learning policy independently (Wen et al. 2021). During the training process, other agents are regarded as part of environment, which doesn't match with realistic scenarios, leading to the non-stationary issue. For instance, IQL (Tampuu et al. 2017) and Independent Proximal Policy Optimization (IPPO) (De Witt et al. 2020) directly apply single-agent DRL algorithms into multi-agent scenarios. In such case, the learned policy is more likely to be wrong due to the incomplete information acquired by each agent. Additionally, the insufficient exploration of state space caused by independent learning may also have an impact on the convergence rate. Therefore, the DTDE paradigm is better suited for tasks where agents operate independently with minimal coordination, such as resource collection or autonomous pathfinding. It is also appropriate for simpler environments with straightforward dynamics and limited interaction complexity, where the non-stationary issue is less significant.

CTDE

Naively applying independent DRL algorithms to MA problems always leads to poor performance due to non-stationary issue, while completely centralized often leads to scalability issue. To reconcile the two methods, a learning paradigm called CTDE was proposed (Lowe et al. 2017), which enables agents to exchange additional information during training, i.e., trains agents in a centralized style, and when executing, each agent takes actions independently based on its only local observations. Thus, CTDE is preferred in environments characterized by partial observability and high coordination demands. The centralized critic enables the learning of optimal cooperative policies, while agents retain the ability to make decentralized decisions based solely on their local observations.

CTDE presents the state-of-the-art practice for MADRL learning, and is used by enormous works. For example, multi-agent deep deterministic policy gradient (MADDPG) based on actor-critic structure, a widely used general learning framework, leverages a centralized critic network and decentralized actor networks to learn the optimum policy for all agents. However, recent works indicate that the CTDE framework still has some limitations.

Zhou et al. (2023a) argues that the centralized training in CTDE is not centralized enough. Specifically, agents' policies are assumed to be independent of each other. Centralized Teacher with Decentralized Student (CTDS) framework (Zhao et al. 2022b) is proposed to address the insufficient utility of global observation in CTDE framework, explicitly consisting of a teacher model and a student model. Specifically, the teacher model learns individual Q-values conditioned on global observation, that is, allocates the team reward, while the student model approximate the Q-values estimated by the teacher model based on the partial observations. As such, CTDS balances the full utilization of global observation during training and the feasibility of decentralized execution. Tests on *StarCraft II* micro-management tasks demonstrate CTDS-based algorithms outperforms some existing algorithms.

3.2.3 Credit assignment (S3)

Cooperative games are main scenarios that MADRL should deal with, in which cooperative agents form team, and receive a global reward (or infer the global return at each timestep). Credit assignment that aims at distributing the global reward to agents according to their contributions, is viewed as both a fundamental problem and important solution, and has been extensively studied in cooperative MADRL based intelligent games. In general, Credit assignment, by differentiating rewards based on agents' contributions, helps address heterogeneity of agents **(C1)** by fairly rewarding agents with varying roles and capabilities. Additionally, it mitigates the challenge of partial observability **(C2)** by ensuring that agents with limited state information are still fairly rewarded, promoting effective learning despite incomplete observations. The naive credit assignment is to distribute the global reward equally to individual agents, so-called shared reward game. However, it doesn't give each agent the accurate contribution, and can't perform well in complex games. Actually, due to the centralized training, it is the problem of credit assignment that mainly restricts the effectiveness of CTDE in MARL.

As shown in Fig. 5, credit assignment in MADRL can be divided into two categories: Implicit and Explicit.

The *implicit* method denotes that the credit assignment occurs implicitly during the training of the mixing network. Mixing network uses as input the local observations of agents and other global system information and state, and outputs the joint value function Q_{tot} . Conversely, the joint value function Q_{tot} is decomposed into individual value function $Q_i, i \in \{1, \ldots, N\}$ by the back propagation process of mixing network. The decomposition facilitated by the mixing network is referred to as the value decomposition network. The core idea in implicit methods lies in that the mixing network and agents' value functions as well as policies are simultaneously learned. That is, they are trained in an end-to-end manner using a unified loss. However, due to limitations imposed by the design of the decomposition function, implicit methods may suffer from inadequate decomposition. Moreover, they are lack of interpretability for the distributed credits.

The *explicit* method entails the explicit distribution of credits to individual agents according to specific rules. Technically, the central critic and the local actors of agents are trained separately. In each iteration, the central critic is first updated to generate the joint action-value function Q_{tot}, and then the reward or the value function of each agent is explicitly inferred. Such reward signals or value functions are used as target to guide the training of local agents.

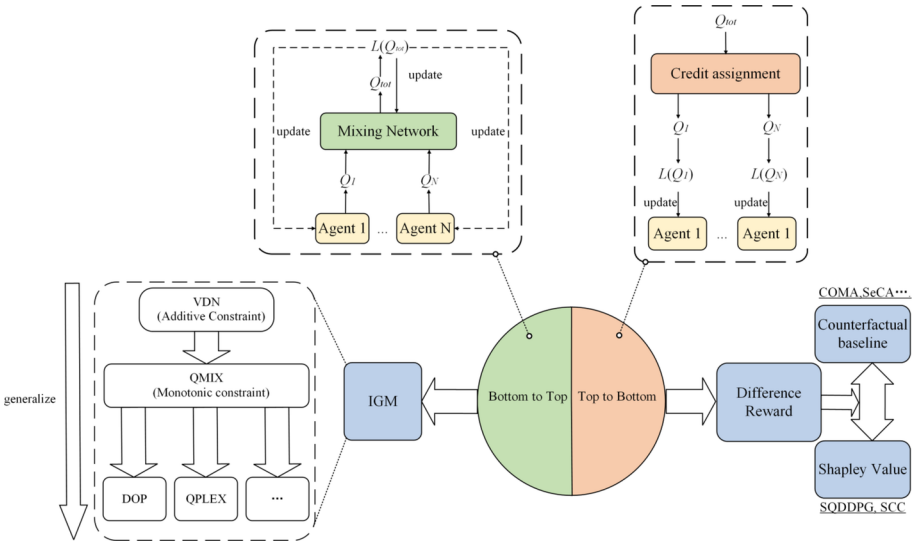

Fig. 5 Categories of credit assignment methods in MADRL

However, the *explicit* methods usually require extra computing resources to calculate the reshaped reward. Most of these algorithms are inspired by the difference reward, and two branches have emerged: counterfactual baseline and the Shapley value. The former is concerned with the estimate of the value of the current action relative to all other agents' actions, that is, the contribution of the current action to the grand coalition formed by all agents, while the latter focuses on the marginal contribution of all possible permutations of the agent set, i.e., coalitions.

- Implicit method

As shown in Fig. 5, the implicit credit assignment method usually holds Individual Global Max (IGM) (Son et al. 2019), a condition that describes the relation between individual Q value and joint Q, shown as Eq. (4).

$$\arg\max_{\boldsymbol{a}} Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}) = \begin{pmatrix} \arg\max_{a_1} Q_1(\tau_1, a_1) \\ \vdots \\ \arg\max_{a_N} Q_N(\tau_n, a_N) \end{pmatrix} \tag{4}$$

where $\boldsymbol{\tau}$ is a joint action-observation histories, $\boldsymbol{a}$ is joint action, and $[Q_i]_{i=1}^{N}$ are individual action-value functions.

Intuitively, IGM means that the optimal joint action is equivalent to the collection of greedy local actions of each agent. As the left part in Fig. 5 shows, the joint Q value can be obtained by taking individual Q values and additional information such as global state. Then the agent can leverage loss function of the joint Q value to update the policyif IGM is satisfied. For example, in QMIX, the loss function can be formulated as Eq. (5).

$$\mathcal{L}(\theta) = \sum_{batch} \left[\left(t_{target} - Q_{tot}(\tau, \boldsymbol{a}, s; \theta) \right)^2 \right], \; t_{target} = r + \gamma \, max_{\boldsymbol{a}'} Q_{tot}(\boldsymbol{\tau}', \boldsymbol{a}', s'; \theta_{target}) \quad (5)$$

where the t_{target} denotes the target value, θ and θ_{target} correspond to parameters of current and target network respectively. r represents instant reward received by agents, and γ is the discount factor. The IGM condition leads to end-to-end training for agents, which simplifies the learning process and improves adaptability.

Value Decomposition Network (VDN) is the earliest algorithm that uses IGM to address the credit assignment issue. VDN exerts additive constraint between global Q value and local Q values, that is, decomposing the joint Q value into the sum of individual Q values. However, the additive assumption is too rigorous to be satisfied by many applications, and fails to generalize to various scenarios.

Different from VDN that adopts a linear decomposition network, QMIX designs a hypernetwork as the mixing network, in which the joint-action value and individual-action values are strictly monotonic (Hong et al. 2022). Then, a series of works have been reported to improve the performance of QMIX through constructing more sophisticated mixing network structures or incorporating other factors.

Considering the limited expressive power of value functions in QMIX, duplex dueling multi agent Q-learning (QPLEX) (Wang et al. 2021a) transforms the IGM consistency into the constraints on the value range of the advantage functions with dueling networks. The scheme not only maintains consistency with IGM, but enhances the expressiveness of the action value function by incorporating the advantage function into the joint Q-value. Learning Implicit Credit Assignment (LICA) (Zhou et al. 2020) aims to releasing the monotonic constraint exerted on individual Q values and joint Q values in QMIX. Specifically, LICA introduces a framework composed of a centralized critic with multiple decentralized actors.

However, although the above value decomposition methods achieve great performance in certain cooperative tasks, they implicitly decompose the total reward signal to the individual value functions, and are lack of interpretability for the distributed credits. Moreover, the performance of these methods heavily relies on the decomposition function and the structure of mixing network.

- Explicit method

Explicit methods can provide interpretability and theoretical support for distributing credits, which are mainly based on the idea of difference reward.

Difference reward infers the contribution of an agent i, r_i^c when the actions of other agents are fixed. Formally, $r_i^c = r(s, \boldsymbol{a}) - r(s, (\boldsymbol{a}_{-i}, c_i))$, where c_i denotes default action of agent i; $\boldsymbol{a}_{-i}$ describes joint actions of all agents except that of agent i. Depending on how to form the set of agents, two different methods of explicit credit assignment based on different reward exist: Counterfactual baseline, and Shapley value.

Counterfactual baseline

For cooperative settings, the joint actions of all agents typically generate a global reward, and it is difficult for each agent to deduce its own contribution to the team's success. To address the issue, Counterfactual multi-agent policy gradients (COMA) (Foerster et al.

2018) uses a counterfactual baseline that marginalizes out a single agent's action, while keeping the other agents' actions fixed.

$$A_i(s, \boldsymbol{a}) = Q(s, \boldsymbol{a}) - \sum_{a_i'} \pi_i(a_i' | \tau_i) Q(s, (\boldsymbol{a}_{-i}, a_i')) \tag{6}$$

As shown in Eq. (6), the advantage function $A_i(s, \boldsymbol{a})$ reflects the contribution of a_i, the current action of agent i, compared to mean evaluation of all possible actions. The baseline is a conditional expectation of the prediction $Q(s, (\boldsymbol{a}_{-i}, a_i'))$, which takes into account all possible actions of agent i, a_i'.

Based on the idea in COMA, Sequential Credit Assignment (SeCA) (Zang et al. 2023) designs a sequential advantage function, where each agent's evaluation is based on its preceding agents' actions, and the actions of the subsequent agents do not influence the evaluation.

Though the counterfactual baseline can well assess the contribution of an agent's action, how to efficiently estimate or define the baseline remains a challenge. Besides, the performance of these methods mostly depends on the advantage function designs and complexity of scenarios.

Shapley value

Considering that Counterfactual baseline methods only consider the set composed of all agents when calculating the marginal contribution of the agent, namely the grand coalition. However, this estimation of contribution is not sufficient because the relationships between agents are very complex and there is no prior knowledge to indicate how they cooperate with each other. Therefore, in the field, the Shapley value in economics is widely used to estimate the contribution of each agent in MADRL, which assigns weights based on the contribution of an agent to various sets of agents (i.e. coalitions) (Wang et al. 2022b; Bian et al. 2022; Shapley et al. 1953; Heuillet et al. 2022; Ghorbani and Zou 2019; Rozemberczki et al. 2022).

Shapley Q-value deep deterministic policy gradient (SQDDPG) (Wang et al. 2020a) assumes that the actions of agents are taken sequentially, which can be leveraged by critic networks to output Shapley Q-values for each agent. Thus, each agent can learn from the reshaped reward, according to their contributions to a coalition. Shapley Counterfactual Credits (SCC) (Li et al. 2021) infers the marginal contributions of an agent through accumulating the change of central critic value caused by ignoring an agent's participation in different set unions, and finally calculates the Shapley value. SCC adopts Monte-Carlo sampling method to alleviate the computational burden of calculating Shapley value.

3.2.4 Scalability-enabling technologies (S4)

Gogineni and Wei (2023) conduct a detailed survey on the scalability issue (**C4**) of MADRL algorithms, and demonstrates that the attention mechanism, code optimization and multi-level compression can contribute to the improvement of scalability. For instance, transformer structure (Vaswani et al. 2017) is used in Multi-Agent Transformer (MAT) (Wen et al. 2022) convert the joint policy search problem into a sequential decision process, which renders only linear time complexity for multiagent problems and endows MAT with mono-

tonic performance improvement guarantee. Tested on *StarCraft* Multi-Agent Challenge (SMAC) and multi-agent MuJoCo tasks, MAT is capable to handle tasks regardless of number of agents.

The scalability issue of multi-agent learning can also be alleviated by directly applying the mean-field approximation to estimating each agent's Q-function (Mao et al. 2022). Instead of taking into account all other agents, Mean-field game (Tembine et al. 2013; Wang et al. 2020b)denotes that every single agent only considers an average effect of its neighborhoods. In game theory and MARL context, mean-field approximation essentially turns N-playergame ($N \rightarrow \infty$) into a "two"-player game (Yang and Wang 2020).

Solving tasks incrementally from simple to complex is also a commonly used scalability technology in MADRL field. Especially, Curriculum Learning (CL) (Wang et al. 2021b) first learns from simple sampled datasets (subtasks), and gradually extends to training datasets (target tasks). In EPC (Long et al. 2020), MADDPG is combined with CL and demonstrate superiority on some complex tasks.

4 Summary of MADRL algorithms for intelligent games

In this section, the state-of-art MADRL algorithms addressing the challenges in intelligent games are comprehensively summarized from the following perspectives: training paradigm, credit assignment and communication structure, as shown in Table 1. It is important to note that performance comparison across MADRL algorithms is highly dependent on the specific benchmarks and scenarios used, as well as the experimental setup (e.g., hyperparameter tuning, implementation details). Hence, this section focuses on summarizing algorithmic features and their suitability for various tasks rather than providing a direct performance ranking. For a detailed evaluation and systematic comparison of MADRL algorithms across various cooperative multi-agent learning benchmarks, interested readers are encouraged to consult the work of (Papoudakis et al. 2021).

The training paradigm includes CTCE, DTDE and CTDE, discussed at Sect. 3.2.2. The dimension of credit assignment is distinguished according to whether the credit assignment methodology is taken into account, either implicitly or explicitly. The dimension of communication corresponds to whether the extended communication structure among agents is designed. Lastly, the code links of algorithms are provided.

Alternatively, these MADRL algorithms are be categorized based on the used DRL paradigm (refer to the appendix), and environmental feature, as shown in Table 2.

In Table 2, the VB and PB respectively indicate value-based and policy-based DRL paradigm. The third column "Environmental feature" describes the applicable game environment for each algorithm, where the "Co" denotes the cooperative type and the "mixed" is both cooperative and competitive.

Selecting the appropriate algorithms in real scenarios and implementations depends heavily on the specific requirements of the target scenario, which are often defined by factors such as environmental settings, communication demands, and credit assignment strategies. For instance, in highly cooperative environments like robotic soccer or other multi-agent sports simulations, where effective coordination and synchronization are essential, algorithms incorporating communication mechanisms—whether through intrinsic frameworks or external strategies—can significantly enhance performance. Conversely, in relatively

Table 1 Summary of MADRL algorithms categorized by MADRL features

Algorithm name	Training paradigm	Credit assignment	Communication	CodeLinks (* denotes that it may not be the officially provided code)
IPPO (De Witt et al. 2020)	DTDE	✗	✗	*https://github.com/jianzhnie/deep-marl-toolkit
MAPPO (Yu et al. 2022)	CTDE	✗	✗	https://github.com/marlbenchmark/on-policy
HATRPO/ HAPPO (Kuba et al. 2022)	CTDE	✗	✗	https://github.com/cyanrain7/TRPO-in-MARL
NA-MAPPO, NVMAPPO (Hu et al. 2021)	CTDE	✗	✗	https://github.com/hijkzzz/noisy-mappo
MADDPG (Lowe et al. 2017)	CTDE	✗	✗	https://github.com/openai/maddpg
LICA (Zhou et al. 2020)	CTDE	✓	✗	https://github.com/mzho7212/LICA
EPC (Long et al. 2020)	CTDE	✗	✗	https://github.com/qian18long/epciclr2020
ATOC (Jiang and Lu 2018)	CTDE	✗	✗	Not found
BiCNet (Peng et al. 2017)	CTCE	✗	✗	*https://github.com/Coac/CommNet-BiCnet
COMA (Foerster et al. 2018)	CTDE	✓	✗	*https://github.com/starry-sky6688/MARL-Algorithms
VDN (Sunehag et al. 2018)	CTDE	✓	✗	*https://github.com/starry-sky6688/MARL-Algorithms
QMIX (Rashid et al. 2020b)	CTDE	✓	✗	*https://github.com/starry-sky6688/MARL-Algorithms
QPLEX (Wang et al. 2021a)	CTDE	✓	✗	*https://github.com/wjh720/QPLEX
QTRAN (Son et al. 2019)	CTDE	✓	✗	https://github.com/Sonkyunghwan/QTRAN
WQMIX (Rashid et al. 2020a)	CTDE	✓	✗	https://github.com/oxwhirl/wqmix
QPD (Yang et al. 2020a)	CTDE	✓	✗	https://github.com/QPD-NeurIPS2019/QPD
NDQ (Wang et al. 2020a, b, c, d, e)	CTDE	✓	✓	https://github.com/TonghanWang/NDQ
MAVEN (Mahajan et al. 2019)	CTDE	✓	✗	https://github.com/AnujMahajanOxf/MAVEN
DGN (Jiang et al. 2020)	CTCE	✗	✓	https://github.com/PKU-RL/DGN

Table 1 (continued)

Algorithm name	Training paradigm	Credit assignment	Communication	CodeLinks (* denotes that it may not be the officially provided code)
DOP (Wang et al. 2021c)	CTDE	✓	✗	https://github.com/TonghanWang/DOP
RIAL/DIAL (Foerster et al. 2016)	CTDE	✗	✓	https://github.com/iassael/learning-to-communicate
MAIC (Yuan et al. 2022)	CTDE	✓	✓	https://github.com/mansicer/MAIC
PAC (Zhou et al. 2022)	CTDE	✓	✗	https://github.com/hanhanAnderson/PAC-MARL

simple environments where agents are expected to act independently, algorithms based on the CTDE paradigm are often more suitable. The characteristics of various algorithms, such as their training paradigms, suitable environmental features, and other factors, as summarized in Tables 1 and 2, can be considered as references for practical applications.

4.1 Value-based MADRL algorithms

VDN (Sunehag et al. 2018) approximates the joint Q-value in a linear factorized manner, enabling each agent to select a greedy action based on its local observations while simultaneously enhancing the joint Q-value. Following VDN idea, QMIX (Rashid et al., 2020b) adopts a monotonic network structure for better representation. The core of QMIX is about the Q learning, and the CTDE framework enables agent to make decisions in a distributed way. Additionally, QMIX can address credit assignment issue implicitly, as discussed in Sect. 3. The above merits make QMIX a classical value-based MADRL algorithm. After that, many variants have emerged.

Considering the structural constraint in factorization of VDN and QMIX, i.e., additivity and monotonicity make them only work for factorizable MARL tasks, QTRAN (Son et al. 2019) is free from such structural constraints through transforming the original joint action-value function into an easily factorizable one, with the same optimal actions. Through providing more general factorization than VDN or QMIX, QTRAN covers a much wider class of MARL.

Targeting at solving the QMIX's representation restriction caused by the monotonic constraint, Q-value Path Decomposition (QPD) (Yang et al. 2020a) leverages the integrated gradient attribution technique into MADRL (Sundararajan et al. 2017) to directly decompose global Q-values along trajectory paths to assign credits for agents.

Weighted-QMIX (WQMIX) (Rashid et al. 2020a) focuses on addressing the limitation that QMIX underestimates the optimal joint action sometimes. The issue arises because the joint Q value is approximated with all joint actions considered equally important. WQMIX designs a weighting function to evaluate the importance of each joint action in QMIX's loss function, which leads to the better joint actions.

Given that QMIX and similar algorithms fail to explicitly consider the individual impact of agents on the overall system during decomposition, Qatten (Yang et al. 2020b) incorporates the multi-head attention mechanism (Vaswani et al. 2017) into mixing network to

Table 2 Summary of MADRL algorithms based on DRL paradigm and environmental feature

Algorithm name	MADRL paradigm	Environmental feature	Publication year
MADDPG (Lowe et al. 2017)	PB	Mixed	2017
COMA (Foerster et al. 2018)	PB	Co	2018
ATOC (Jiang and Lu 2018)	PB	Co	2018
IPPO (De Witt et al. 2020)	PB	Co	2020
MAPPO (Yu et al. 2022)	PB	Co	2020
LICA (Zhou et al. 2020)	PB	Mixed	2020
EPC (Long et al. 2020)	PB	Co	2020
HATRPO/ HAPPO (Kuba et al. 2022)	PB	Mixed	2022
NA-MAPPO/ NVMAPPO (Hu et al. 2021)	PB	Co	2023
RIAL/DIAL (Foerster et al. 2016)	VB	Co	2016
VDN (Sunehag et al. 2018)	VB	Co	2017
QMIX (Rashid et al. 2020b)	VB	Co	2018
QTRAN (Son et al. 2019)	VB	Co	2019
MAVEN (Mahajan et al. 2019)	VB	Co	2019
Qatten (Yang et al. 2020b)	VB	Co	2020
DOP (Wang et al. 2021c)	VB	Co	2020
QPLEX (Wang et al. 2021a)	VB	Co	2021
WQMIX (Rashid et al. 2020a)	VB	Co	2020
QPD (Yang et al. 2020a)	VB	Co	2020
NDQ (Wang et al. 2020a, b, c, d, e)	VB	Co	2020
DGN (Jiang et al. 2020)	VB	Co	2021
MAIC (Yuan et al. 2022)	VB	Co	2022
PAC (Zhou et al. 2022)	VB	Co	2022

address this issue. Each individual Q value will be calculated with a weight provided by attention mechanism, which takes global information and agent-relevant information as inputs. The weight somewhat demonstrates the influence of the corresponding agent to the whole system, thus the joint Q value can be more accurately estimated.

QPLEX (Wang et al. 2021a) as mentioned in Sect. 3, introduces dueling network into Q value, which reduces the error variance. Besides, QPLEX also adopts multi-head attention mechanism as Qatten in the mixing network, to efficiently learn weights of each advantage function. These modules enable QPLEX to realize the full expressiveness of value factorization.

The above MADRL algorithms follow the CTDE learning paradigm, in which a centralized critic collects all required information, that is, achieves an implicit communication. Another methodology to address issues of partial observability and non-stationary in MADRL scenarios is to explicitly design communication mechanisms for each agent.

Besides learning the action response to environment, Reinforced Inter-Agent Learning (RIAL) and DIAL meanwhile learn the communication actions among agents in multi-agent tasks with partial observability. Nearly Decomposable Q-functions (NDQ) (Wang et al. 2020a, b, c, d, e) introduces a communication method, in which agents occasionally send messages to other agents for effective coordination. Especially, regularizers are adopted to maximize mutual information between agents' action selection and communication messages while minimizing the entropy of messages between agents. Under this framework, each agent can learn to communicate with low communication overhead.

For multi-agent scenarios, the high dynamic changing environments usually lead to the fast changes of agents' neighbors. To understand the dynamic mutual interplay between agents, graph convolutional reinforcement learning (DGN) (Jiang et al. 2020) is proposed. In DGN, the multi-agent environment is modeled as a graph, where each node represents an agent characterized by its observed state, and an edge represents a connection between agents based on factors such as distance or other metrics. Graph convolution adapts to the dynamics of the underlying graph, and relation kernels capture the interplay between agents.

For traditional communication scheme, the agents just exchange their local observations or latent embeddings and use them to augment individual local policy input, which enlarges agents' local policy spaces and leads to poor coordination in complex scenarios. Towards realizing efficient teamwork, Multi-Agent Incentive Communication (MAIC) (Yuan et al. 2022) investigates a communication scheme that allows agents to exchange persuasive messages to explicitly coordinate their decisions. In MAIC, each agent learns to generate incentive messages, which bias the other agent's Q-values as an incentive. This approach would effectively promote the coordination and not explicitly enlarge the policy space of every agent.

Besides the value decomposition and explicit communication structure, several value-based MADRL algorithms are designed from the aspects of optimizing exploration strategy, enhancing learning efficiency, and increasing expressive capability. MAVEN (Mahajan et al. 2019) focuses on how to effectively explore in decentralized MADRL. In MAVEN, each agent will learn a shared potential variable controlled by hierarchical policies. The different value of the variable indicates different exploring strategies, instead of using ϵ -greedy policy unanimously. Such extended exploration increases the probability of finding an optimal strategy. In some complex tasks, agents' actions usually don't come from the same set. Thus, one shared policy network can't represent and learn all required skills, while using a distinct

policy network for each agent may incur the high learning complexity. To address the issue, Role-Oriented MARL (ROMA) (Wang et al. 2020d) introduces the concept of roles as an intermediary that enables agents with similar responsibilities to share their learning. A trainable neural network taking two trajectories as input is used to estimate dissimilarity between two agents' trajectories, and learns agents' roles. In such framework, the cooperation among agents is promoted and team efficiency is improved.

PAC (Zhou et al. 2022) demonstrates that an agent's ordering over its own actions could impose concurrent inequality constraints on the joint action-value Q function in different states. These constraints further lead to large estimation errors of the joint Q function. To address this issue, PAC optimizes mutual information and variational inference, introducing the generated auxiliary information into the framework of value decomposition. With the assisting information, PAC directly improves agents' value functions through factorization.

4.2 Policy-based algorithms

MADDPG (Lowe et al. 2017) uses actor-critic architecture, in which each agent has a critic neural network that takes global information as inputs, and outputs the evaluation of individual actions. With the framework, MADDPG is applicable not only to cooperative interaction but to competitive or mixed interactions.

However, in cooperative settings, joint actions typically generate only global rewards like MADDPG, making it difficult for each agent to deduce its own contribution to the team's success, so-called credit assignment problem. To address this issue, COMA (Foerster et al. 2018) follows the CTDE architecture and each agent has a specific advantage function that compares the agent's expected return with counterfactual baseline while the other agents' actions are fixed. In this case, each agent can learn with a reshaped value function according to its own contribution, which leads to better average performance for decentralized policy.

Similarly, LICA (Learning Implicit Credit Assignment) (Zhou et al. 2020) adopting actor-critic architecture under the CTDE, extend the value mixing to policy mixing, where the centralized critic is formulated as a hypernetwork that first maps current state information into a set of weights, concatenates them with action vectors, and finally formulates the critic value. Note that, different from MADDPG maintaining a separate centralized MLP critic network for each agent, LICA uses a single centralized critic (one for all agents) formulated as a hypernetwork.

To solve the centralized-decentralized mismatch that means that the suboptimality of one agent's policy can propagate through the centralized joint critic and negatively affect policy learning of other agents, multi-agent decomposed policy gradient method DOP (Wang et al. 2021c) introduces the VDN into the multi-agent actor-critic framework, which decomposes the centralized critic as a weighted linear summation of individual critics conditioned on local actions. Wang et al. (2021c) argued that, although a linearly decomposed critic has limited representational capacity, and may induce bias in value estimations, this bias does not violate the policy improvement guarantee of policy gradient methods.

How to establish effective communication mechanism in policy-based algorithm is also a challenging problem. Especially, to tackle the issue that agents may not distinguish valuable communication information from globally shared information, ATOC (Jiang and Lu 2018) combines actor-critic model with attention unit, in which an attention unit dynamically determines whether communication is needed for cooperation and a bi-directional

LSTM unit is used as a communication channel to interpret encoded information from other agents, which efficiently exploits communication for cooperation.

Trust region optimization represents another significant enhancement to the policy gradient method, e.g., Proximal Policy Optimization (PPO). It effectively limits the magnitude of policy updates, and thereby improves both convergence and training stability. IPPO (De Witt et al. 2020) and Multi-Agent PPO (MAPPO) (Yu et al. 2022) are two typical cases applying PPO into MA scenarios. IPPO applies an individual PPO algorithm to each agent, learning decentralized policies with policy clipping. The common issue for independent learning is the difficulty of convergence due to non-stationary. MAPPO follows the CTDE structure, using a centralized value function, i.e., critic network based on the observation of the global state.

NA-MAPPO/NV-MAPPO (Hu et al. 2021) points out that MAPPO may face the problem of the policy overfitting in multi-agent cooperation. Since it learns decentralized policies by the sampled centralized advantage values, also known as credit assignment problem, it may lead to learning in a suboptimal direction. To address the above issue, NA-MAPPO/NV-MAPPO adds noise to advantage function directly and state information respectively. The added noises can effectively improve the exploration capability of algorithms and prevent from trapping into suboptimal.

Considering that sharing the same action space and policy parameters significantly limits agents' applicability and potentially harms the performance, Heterogeneous-Agent Trust Region Policy Optimisation (HATPRO) and Heterogeneous-Agent Proximal Policy Optimisation (HAPPO) (Kuba et al. 2022) utilize multi-agent advantage decomposition lemma and the sequential policy update scheme to support the heterogeneous agents to learn, and provide essential theoretical property of the monotonic improvement guarantee.

4.3 Limitations of MADRL algorithms

Although the MADRL algorithms examined in preceding subsections have demonstrated efficacy across diverse scenarios, it is imperative to recognize their inherent limitations that constrain scalability and applicability. In this subsection, the key limitations of these MADRL algorithms are generalized as follows:

- **Computational overheads**

Computational overheads represent fundamental constraints in practical MADRL implementations. As the number of agents scales, the training process becomes increasingly intractable due to the exponential growth of the state-action space, particularly in real-time strategy games that require coordination among hundreds of units. Additionally, reward assignment mechanisms for agents with divergent objectives—such as the mixed competitive-cooperative scenarios in games like *StarCraft* or *DOTA*—necessitate computationally intensive optimization to balance local and global incentives.

- **Communication limitations**

Communication limitations impose additional challenges, particularly problematic in latency-sensitive game environments such as FPS games. While auxiliary communication

modules or mechanisms can improve collaboration, they introduce inference delays and extra communication overheads during decentralized execution. Furthermore, privacy-preserving communication protocols, though theoretically promising for scenarios involving confidential agent policies—such as fog-of-war exploration in real-time strategy games—remain a challenge in MADRL algorithm development.

- **Reinforcement learning paradigm**

The MADRL algorithms discussed in this section all belong to model-free paradigm, which imposes inherent limitations on sample efficiency. These algorithms rely heavily on direct interaction with the environment to collect data, which can be time-consuming. As a result, they often require a large number of samples to converge to an optimal solution. The high sample complexity becomes particularly problematic in scenarios where data collection is cost or difficult.

5 Benchmarks for MADRL based intelligent games

The proliferation of diverse MADRL benchmarks/simulation environments in recent years has facilitated the simulation of real-world MADRL tasks. On one hand, they offer valuable platforms for validating and refining various MADRL algorithms. While standard RL benchmarks are generally designed to evaluate a broad range of algorithms, including both single-agent and multi-agent methods, certain benchmarks in the MADRL domain are specifically tailored to assess particular aspects of multi-agent interactions, thus forming a distinct category of MADRL benchmarks. Beyond their utility for MADRL algorithm validation and improvement, on the other hand, these benchmarks also serve as instrumental resources, offering insights and guidance for the deployment of algorithms in customized environments. As summarized in Table 3, a set of MADRL benchmarks is provided, along with key features such as task type, typical tasks, and supported observations/actions. It is important to note that each benchmark typically involves multiple tasks with varying numbers of agents, and no maximum agent limit is usually imposed. The number of agents can therefore be customized for scalability testing purposes. A larger number of agents increases computational costs, so determining the number of agents should be guided by the specific objectives of the experiment and the computational capacity of the supporting hardware.

Table 3 summarizes the existing MADRL benchmarks from several dimensions, including type of game tasks, the space of observation/action, and the corresponding links are also presented. Note that, “Co” and “Cp” respectively denotes cooperative and competitive game environment, and “C”, “D” and “B” denotes that there exist tasks in environment with continuous, discrete, and both continuous and discrete observation/action space, respectively. Similarly, given that each MADRL benchmark contains a variety of tasks, it is difficult to cover all challenges within each benchmark in a limited scope. To illustrate the connection with intelligent games and their associated challenges, we list a typical task from each benchmark as an example. *Predator-Prey*, *Pursuit*, *4C2T*, *2R2P* are all focused on cooperative target tracking and can be classified as relatively simple RTS games. *Predator-Prey* involves a basic continuous observation space, where agents collaboratively chase a target, while *Pursuit*, set in a 2D grid environment, uses discrete observation and action spaces,

Table 3 Summary of MADRL benchmarks

Environments / benchmarks	Task type	Typical task	Observations /actions	Links
MPE (Lowe et al. 2017)	Mixed	Predator-Prey	C/B	https://github.com/openai/multiagent-particle-envs
MAgent (Zheng et al. 2018)	Mixed	Pursuit	D/D	https://github.com/geek-ai/MAgent
SMAC/SMACv2 (Whiteson et al. 2019; Ellis et al. 2024)	Co	3 Stalkers vs. 5 Zealots (3s5z)	C/D	https://github.com/oxwhirl/smacv2
Google Research Football (Kurach et al. 2020)	Co	3 vs. 1 with Keeper	C/D	https://github.com/google-research/football
WGC (Yao et al. 2023)	Co	POAC	C/D	*http://turingai.ia.ac.cn/datacenter/show/10
MATE (Pan et al. 2022)	Mixed	4 cameras vs. 2 targets (4C2T)	C/B	https://github.com/UnrealTracking/mate#mate-the-multi-agent-tracking-environment
Butterfly (Terry et al. 2021)	Co	Cooperative Pong	B/B	https://github.com/Farama-Foundation/PettingZoo
OpenSpiel (Lanctot et al. 2019)	Mixed	Tic-Tac-Toe	B/B	https://github.com/google-deepmind/open_spiel
APE (Maddila et al. 2024)	Mixed	2 rangers vs. 2 poachers (2R2P)	D/D	https://forgemia.inra.fr/chip-gt/antipoaching
POGEMA (Skrynnik et al. 2024)	Co	Mazes	D/D	https://github.com/AIRI-Institute/pogema

adding structural constraints. The complexity increases in *4C2T*, where agents' observation ranges are restricted to a cone-shaped field of view, requiring enhanced coordination. In *2R2P*, the task becomes even more intricate with agents gaining access to diverse actions, such as setting traps to hinder the target's movement.

SMAC and WarGame Challenge (WGC) represent classic RTS games with significantly higher demands for real-time decision-making and strategic planning. For instance, SMAC's *3s5z* scenario involves three Stalkers battling five Zealots, where agents must not only focus on achieving victory but also minimize costs through advanced tactics, such as focusing fire on low-health enemies and positioning healthier agents to absorb damage. Similarly, WCG features a partially observable asynchronous multi-agent cooperation environment (POAC), where each team comprises a chariot, a tank, and an infantry unit, necessitating strategic coordination under partial observability to secure victory. Meanwhile, Google Research Football's *3 vs. 1 with Keeper* task presents a soccer-based scenario where three offensive players must cooperate to outmaneuver a defensive player and goalkeeper to score. As this task emphasizes real-time decision-making and strategic planning rooted in realistic sports rules, it can be considered a specialized subcategory of RTS games, specifically a sports simulation game.

For *Mazes* in POGEMA and *Cooperative Pong* in Butterfly, *Mazes* are more aligned with strategy games, and more specifically, puzzle games, as they require agents to navigate a pre-defined partially observable grid map, reach their goals, and avoid collisions with both static obstacles and other agents. In contrast, *Cooperative Pong* falls under the category of arcade games, emphasizing real-time operations and cooperation. In this game, the objec-

tive is for both agents (paddles) to work together to keep the ball in play, with the game ending if the ball goes out of bounds on either the left or right edge of the screen. And *Tic-Tac-Toe*, a turn-based strategy game, is a perfect information and competitive task, where two players place marks in turn on a grid. The agents can observe full information about the game, including all marks' positions placed by opponents and themselves.

For all simulation benchmarks mentioned above, the general pipeline to implement MADRL based algorithms or game design are summarized as Fig. 6, which is composed of the following sequential procedures.

- Extract environment information. It always corresponds to the realistic scenario requirements and the computation capability of the hardware. Specifically, it is imperative to recognize the existing entities, dynamic behaviors and interactions of them, operational mechanisms and performance criteria of the entire game. Furthermore, the information structure and environmental feature should be identified.
- Formulate MADRL model. First, MADRL elements should be defined including mul-

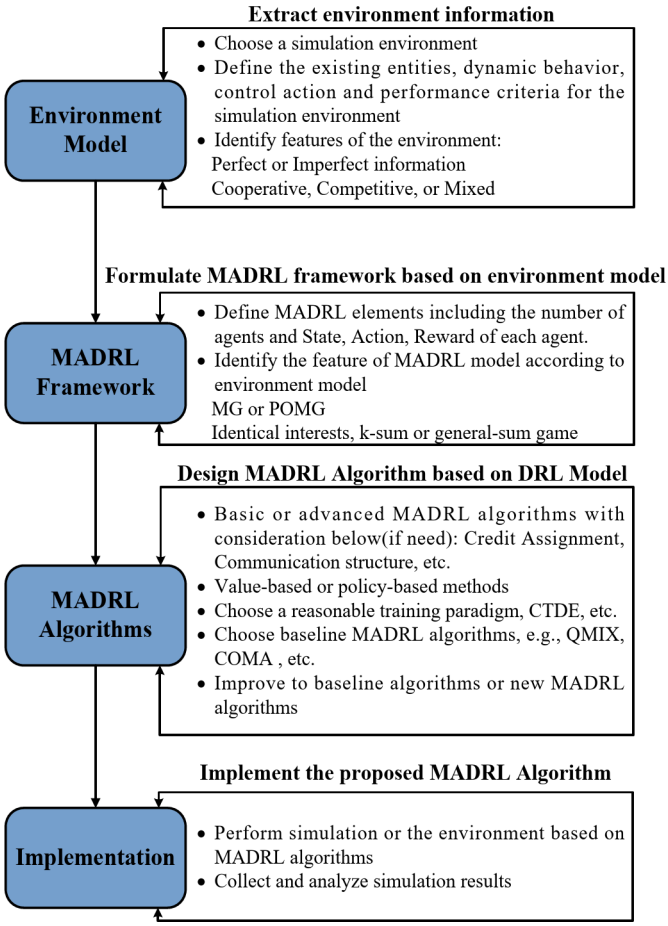

Fig. 6 General procedure for MADRL simulations

tiple agents, states (or the observations), actions and rewards should be defined. Then, based on the information and feature of the simulating environment, a general framework can be identified for selecting subsequent algorithms.

- Design MADRL algorithms. Various MADRL algorithms as summarized in Sect. 4, can be chosen or adapted to design MA based intelligent games to train the agents. Moreover, MADRL solution methodologies can be appropriately incorporated into the proposed algorithms, including communications structure that can effectively enhances the coordination between agents, and credit assignment that can fairly distribute value according to agents' contributions, and guide the global cooperative behavior.
- Implement/deploy the proposed MADRL algorithms in the simulating environment. Finally, the proposed framework and algorithms will be implemented, and the results will be analyzed to provide insights for the practical scenario applications.

As an example, Multiagent Particle Environment (MPE) is briefly introduced as follows. MPE (Lowe et al. 2017) is a set of time-discrete, spatially continuous two-dimensional multi-agent environment developed by OpenAI. In MPE, agents can take physical actions in the environment and communication actions that can be broadcasted to other agents. Some environments require explicit communication between agents in order to achieve the best reward, while in other environments agents can only perform physical actions. MPE implements various MADRL tasks. Following the procedure showed in Fig. 6, Predator-Prey is a typical task in MPE. For environment model, Predator-Prey is a cooperation task with imperfect information, where agents learn to cooperatively track and capture the designated targets while avoiding collisions. For MADRL, the observation of each agent includes relative position to other agents and its own position, velocity. And the action of each agent is predefined as move in the direction of top, bottom, left or right.

Note that, various MADRL algorithms used to control and optimize the MA tasks can be freely inserted into any MADRL benchmarks. The main benefit of MADRL benchmarks is to easily and flexibly designate the environment models and MADRL frameworks and facilitate the design and implementation of MADRL algorithms.

6 MADRL based real intelligent game applications

Barambones et al. (2022) systematically investigates the application of MAS into designing virtual games, including reinforcement learning and evolutionary techniques, and draws a conclusion that Real-Time-Strategy (RTS) and sports game are suitable platforms for MADRL implementations. In (Yin et al. 2023), recent studies of game AI are reviewed from imperfect information, long time horizon, in-transitive game and multi-agent cooperation. However, it is lack of analysis of the association between MADRL and intelligent game. In this section, several famous intelligent games are offered as examples to indicate the importance role that MADRL plays in them.

Some typical MADRL inspired intelligent game applications are listed in Table 4. The "Cp" and "Co" respectively denotes cooperative and competitive game environment, and "Mixed" represents that the environment includes both competitive and cooperative settings. Note that in the column of (MA)DRL, some single-agent DRL algorithms are included, due to the observation that many intelligent games adopt self-play to train models

Table 4 Summary of intelligent game applications

Game	Settings	(MA)DRL algorithms	Work
Football	Co	MAPPO	Tizero (Lin et al. 2023)
Snakes	Mixed	IPPO	3v3 Snake (Wang et al. 2022a)
Stratego	Cp	R-Nad	DeepNash (Perolat et al. 2022)
Majong	Cp	Policy gradient	Suphx (Li et al. 2020)
Honor of Kings	Co	PPO	(Ye et al. 2020a, b)
B&S Arena Battles	Cp	ACER	(Oh et al. 2021)
Gran Turismo racing	Cp	QR-SAC	Gran Turismo Sophy (Wurman et al. 2022)
Game of Drones	Co	MADDPG	(Zhang et al. 2022)
Xiangqi	Cp	MCTS	JiangJun (Li et al. 2023b)
Werewolf	Mixed	MAPPO	(Xu et al. 2024)
Chase tag	Cp	PPO	(Han et al. 2024)

in an adversarial way, which, in a sense, can be regarded as is a special kind of MADRL algorithms. The rightmost column represents the intelligent system for games after training with MADRL algorithms.

As an example, Tizero game is introduced. Tizero (Lin et al. 2023) is a self-evolving, multi-agent system based on MAPPO for Google Research Football (GFootball). Prior work on GFootball has mostly focused on simplified scenarios, for instance requiring agents to score in an empty goal or beat the goalkeeper in a 1-on-1. Tizero extends it to the full 11 vs. 11 game mode, which is a more challenging and complex task. As for MADRL algorithms in Tizero, a modified version of MAPPO, named Joint-ratio Policy Optimization (JRPO) is designed. Compared with MAPPO, where each agent's policy is optimized individually, JRPO optimize the joint-policy using a decentralized factorization, and all agents' policies are optimized jointly with a joint-policy objective. The method experimentally proves its advantage compared to vanilla MAPPO, which additionally reduces memory usage and improves training speed. Combining with the other techniques such as curriculum learning, Tizero is capable to firstly control all 10 outfield players in a decentralized fashion, and effectively trains agents in the GFootball 11 vs. 11 game mode with challenges of multi-agent coordination, sparse rewards, and non-transitivity.

In the *Game of Drones* (Zhang et al. 2022), quadcopters always cooperate with each other, to form an unmanned swarm system and confront with enemies. In, an intelligent system controlling drones swarm for pursuit-evasion game is proposed. As mentioned in Sect. 5, the game resembles the Predator-Prey subtask in the MPE benchmark. In the scenario, drones are required to learn to pursue targets while avoiding the obstacles. Given the imperfect information and cooperative nature of the simulation environment, effective communication is essential, and the problem can be modeled as a POMG. Consequently, in the design of MADRL algorithms, a bi-directional communication and target prediction network (CBC-TP), an improved version of MADDPG, is employed. This approach

enables communication between a variable number of agents and facilitates the learning of collaborative policies. Simulation results demonstrate that the well-trained CBC-TP agents are resilient to disruptions caused by non-functional agents. After training, the drones successfully learn to search for targets and avoid collisions, achieving robust performance in the pursuit-evasion game.

7 Open issues

This review highlights the broad applicability and importance of integrating MADRL into intelligent games. Specifically this section identifies the challenges and suggests some future directions.

- **Explainable MADRL for intelligent games**

Explainability has become an essential requirement in MADRL (Dazeley et al. 2023; Dwivedi et al. 2023). As DRL-based methods often generate innovative solutions that may not be immediately intuitive to human observers, it is essential to employ explainable techniques to verify the correctness of these solutions. Such explanations can also shed light on the novel strategies or approaches formulated by DRL models, for instance, understanding why a specific strategy was selected in a video game. In addition, MADRL models are challenging for developers to debug due to their reliance on various factors, including the environment (especially the design of the reward function), observation encoding, large deep learning models, and the training algorithm used for policy learning. Explainable MADRL models, particularly in the context of intelligent games, can significantly accelerate problem identification and resolution, thereby improving the efficiency of both MADRL methods and intelligent game development. More importantly, real-world applications often demand interpretable policies that stakeholders can inspect and understand prior to deployment. Examples include traffic management, autonomous driving, and other critical domains where transparency and trust are paramount. To enable explainable MADRL, one of intrinsic methods is to extract decision-tree policies from MARL-trained neural networks, which are able to interpret the underlying decisions made by agents (Zhang 2024). Additionally, numerical methods, such as assigning values to model inputs based on their contribution to outputs (e.g., using the Shapley value), offer another avenue for interpretability. However, existing explainability methods face significant limitations in more complex scenarios, stemming from factors such as the increasing complexity of action spaces and the heightened computational demands (Hickling et al. 2023).

- **Standardized performance evaluation**

Establishing standardized performance evaluation metrics for the diverse MADRL algorithms used in intelligent games is crucial. While some evaluation frameworks and criteria have been proposed independently (Gorsane et al. 2022), the increasing complexity of dynamic environments highlights the growing need for standardized performance evaluation. Such standardization is essential for driving the development of innovative MADRL algorithms and, in turn, advancing complex intelligent games. The literature contains a sub-

stantial body of work focused on building benchmarking tools, such as PyMARL2 (Hu et al. 2023b), TorchRL (Bou et al. 2024), MARLlib (Hu et al. 2023c), and BenchMARL (Bettini et al. 2024), which support standardization and reproducibility in evaluation. However, with the continuous emergence of new MADRL algorithms, it is equally important to update and refine performance evaluation platforms to keep pace with these developments.

- **Model-based MADRL for intelligent games**

Although significant advances have recently been achieved in MADRL based intelligent games, most works focus on the model-free (MF) based MADRL. A key limitation of MF-based MADRL is its high sample complexity, which can severely hinder effective training. In contrast, model-based (MB) DRL methods offer proven advantages in sample efficiency. Unlike MF-based MADRL, which relies on environment-provided rewards, MB methods estimate an empirical model using data and then derive equilibrium policies through planning within this model. Once the model is estimated, this approach can potentially address multiple MARL tasks with different reward functions but a common transition model, without the need for re-sampling. The ability to handle reward-agnostic cases significantly enhances the capability of model-based approaches (Zhang et al. 2023b). However, MB MADRL is just in its infancy. Well-designed MB MADRL frameworks have significant potential to improve sample efficiency in training and enable effective learning of optimal policies by multiple agents in both cooperative and competitive environments (Subramanian et al. 2023; Wang et al. 2022c).

- **Safe MADRL for intelligent games**

RL safety has garnered increasing attention in recent years, highlighting its importance as a critical practical concern (Ji et al. 2023; Gu et al. 2024). When extended to multi-agent scenarios, ensuring safety becomes even more challenging due to the inherent complexity and unpredictability of interactions among agents. In simulations or games, random exploration to discover optimal strategies is acceptable because the virtual environment allows for trial-and-error learning without significant consequences. However, in real-world applications, certain behaviors must be avoided or strictly prohibited. Neglecting safety considerations in RL systems could result in unacceptable catastrophes. For instance, in human-robot interaction environments, robots must not harm humans under any circumstances. Similarly, recommender systems should avoid recommending false or racially discriminatory information, and self-driving cars must prioritize safety while performing tasks in real-world environments. In the context of multi-agent systems, these safety challenges are amplified. Agents not only need to act safely themselves but also must account for the actions and potential failures of other agents. Existing methods tend to satisfy safety constraints in training progress to learn safe policy (Liu et al. 2021; Gu et al. 2023), or specify safe actions and correct unsafe actions that agents explored (ElSayed-Aly et al. 2021; Carr et al. 2023; Qiu et al. 2024). Nevertheless, achieving a zero-violation, robust Safe MADRL method remains a significant and unresolved challenge, requiring further innovation and exploration to ensure multi-agent systems can operate safely and reliably in real-world environments.

- **Language-conditioned MADRL for intelligent games**

Recent advances in Natural Language Processing (NLP) have showcased remarkable capabilities in multi-modal tasks, making language-conditioned MADRL an exciting and promising research direction. A notable example is the integration of Large Language Models (LLMs) with MADRL (Wang et al. 2024; Sun et al. 2024). Unlike traditional MADRL, LLM-based MADRL can leverages linguistic cues to enhance inter-agent communication and collaboration (Hu et al. 2023a, b, c), significantly improving system performance. For instance, agents can utilize a shared language to negotiate roles, coordinate actions, or exchange information about their environment and internal states. This capability enables them to align their objectives more effectively. Such language-driven coordination is particularly crucial in complex scenarios where agents must address ambiguous or dynamic tasks that demand continuous communication and mutual understanding. In addition, LLM-based MADRL can generate reward signals by employing an LLM as a proxy reward function (Kwon et al. 2023), thereby avoiding the challenges of designing complex reward functions or relying on expert demonstrations. Exploring these capabilities opens up new possibilities for developing more intelligent and adaptable multi-agent systems that can operate effectively in unpredictable real-world environments.

8 Conclusions

In literature, on one hand, AI-driven intelligent games have witnessed great development. On the other hand, MADRL has emerged as a powerful tool across various AI domains for sequentially determining the long-term optimal policies for all agents in cooperative and competitive environments, in which partial observability and non-stationary issues are typical challenges. Even though there naturally exist common points between multi-player AI driven intelligent games and MADRL, surprisingly, there is lack of systematical review on building the constitutional connections between them. This article aims to bridge this gap by offering a comprehensive overview through smoothly integrating intelligent games with MADRL from two key perspectives: games inspire MADRL, and MADRL enhances the development of intelligent games. Specifically, theoretical concepts and features of MADRL from game environments are firstly presented. Subsequently, the article delineates the challenges inherent in intelligent games explores existing MADRL methodologies tailored to address these challenges, and the state-of-the-art MADRL algorithms are comprehensively categorized. Then, various simulation environments/benchmarks are summarized, which can serve both as the utilities for evaluating MADRL algorithms and as the instruments for developing customerized MADRL-enabled intelligent games. Furthermore, the general procedures for MADRL simulations are presented. Besides, notable successes in MADRL-based intelligent game applications are listed to highlight the vast potential of MADRL. Finally, this article explicitly points out some key challenges in integrating intelligent games with MADRL.

Appendix

A. RL concepts and frameworks

The key concept of RL is to explore its environment via trial-and-error, that is, the agent tries out an action and observes the outcome to gain knowledge, and simultaneously exploit the gained knowledge to optimize its action/policy for long-term return. Technically, at each time step t, the agent in state s_t takes the action a_t and receives a reward r_{t+1} from the environment, and state (probabilistically) transfers into s_{t+1}, forming a transition $(s_t, a_t, r_{t+1}, s_{t+1})$. The agent interacts with the environment at discrete time steps, and generates a sequence of states (possibly infinite): $s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_T$ and collect a sequence of reward samples: $r_1 \rightarrow r_2 \rightarrow r_3 \rightarrow \ldots \rightarrow r_T$, jointly denoted as episode or trajectory $\tau = (s_0, a_0, r_1, s_1, a_1 \ldots, s_T)$, The above procedure is usually modeled as a Markov Reward Process (MRP): a Markov Chain where each transition is associated with a scalar reward r. The agent' goal is to find a policy to maximize the return R_t at each timestep t, i.e., the discounted sum of future rewards, shown as Eq. (7).

$$R_t = r_{t+1} + \gamma\, r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{7}$$

Where $\gamma < 1$ is the discount factor.

A core component in RL is to estimate the value function $V^{\pi}(s)$ of every state s and/or action-value function $Q^{\pi}(s,a)$ for each state-action pair (s,a) when agent follows the current policy π. The Q value, $Q^{\pi}(s,a)$ can be represented as Eq. (8).

$$Q^{\pi}(s,a) = \mathbb{E}_{\rho_{\pi}}(R_t | s_t = s, a_t = a) \tag{8}$$

The mathematical expectation operator $\mathbb{E}(\cdot)$ is indexed by ρ_{π}, the probability distribution of states achievable with π.

The state value $V^{\pi}(s)$ is given as Eq. (9).

$$V^{\pi}(s) = \mathbb{E}_{\rho_{\pi}}(R_t | s_t = s) = \mathbb{E}_{a \sim \pi(s,a)}[Q^{\pi}(s,a)] = \sum_{a \in \mathcal{A}(s)} \pi(s,a)\, Q^{\pi}(s,a) \tag{9}$$

For a fixed transition (s_t, a_t, s_{t+1}), the Q-values can be obtained when the state-values are known, given as Eq. (10).

$$Q^{\pi}(s_t, a_t) = r(s_t, a_t, s_{t+1}) + \gamma\, V^{\pi}(s_{t+1}) \tag{10}$$

If the transition probabilities $p(s_{t+1}|s_t, a_t)$ are stochastic, Eq. (10) simply takes expectation operator $\mathbb{E}(\cdot)$ indexed by the transition probabilities.

The recursive relationship $Q^{\pi}(s,a)$, i.e., the Bellman equation for Q^{π}, is shown as Eq. (11), which depends on the dynamics of the MRP, i.e., $(p(s'|s,a),\ r(s,a,s'))$ and the current policy π.

$$Q^{\pi}(s,a) = \sum_{s' \in \mathcal{S}} p(s'|s,a) \left(r(s,a,s') + \gamma \sum_{a' \in \mathcal{A}(s')} \pi(s',a')\, Q^{\pi}(s',a') \right) \tag{11}$$

Note that in our paper, the Q and V values with superscript π, denote the true Q value and state value under the current policy π, which are all unknown to the decision-maker, i.e., agent, and estimated through many ways.

Practically, for a given policy π, the value of all states $V^{\pi}(s)$ or all state-action pairs $Q^{\pi}(s,a)$ can be inferred based on sampled experiences, including Monte-Carlo (MC) and Temporal Difference (TD) methods.

As shown in Eq. (12), MC approximates the mathematical expectation by sampling M trajectories τ_i $(i = \{1,2,\ldots,M\}$ starting from the state s and computing the sampling average of the obtained returns $R(\tau_i)$.

$$V^{\pi}(s) = \mathbb{E}_{\rho_{\pi}}(R_t|s_t = s) \approx \frac{1}{M}\sum_{i=1}^{M} R(\tau_i) \tag{12}$$

The trajectories used to evaluate policy can be either generated by the current π, so-called on-policy RL, or sampled from another behavior $b(s,a)$ calibrated by the importance sampling, so-called off-policy RL. To ensure exploration, in on-policy control methods, the learned policy has to be ϵ-soft, which means all actions have a probability of at least $\epsilon/|\mathcal{A}|$ to be visited. The main advantage of off-policy strategies lies in that it can learn from other's experience, including expert's experiences, and greatly reduce the number of transitions needed to learn a policy.

The weak point of MC methods lies that the leaning is slow, and can't work for the continuing tasks, since they have to wait until the end of the episode to compute the obtained return.

TD methods simply replace the actual return by an estimation in the update rule, shown as Eq. (13).

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha \cdot \delta_t \tag{13}$$

where $\delta_t = [(r_{t+1} + Q(s_{t+1},a_{t+1})) - Q(s_t,a_t)]$ is the TD error; α is learning rate.

Equation (13) implies that, in estimating the Q value, the next action a_{t+1} should be determined. Similarly as MC methods, there are two categories of TD based RL algorithms: on-policy (e.g., state-action-reward-state-action, SARSA) and off-policy (i.e., Q-learning). The former selects the next action using the current policy π, that is, when arriving in s_{t+1} from (s_t,a_t), the next action a_{t+1} has been already sampled: $a_{t+1} \sim \pi(s_{t+1},a)$. Instead, Q-learning directly approximates the optimal action-value function through the greedy action in the next state, that is, independent of the current policy, shown in Eq. (14).

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha \cdot \left[\left(r_t + \gamma \max_a Q(s_{t+1},a)\right) - Q(s_t,a_t)\right] \tag{14}$$

In Q-learning, the next action a_{t+1} can be generated by a behavior policy different form the current policy, but the learned policy can be deterministic. Note that the requirement for the behavior policy is that it needs to visit all state-action pairs during learning to ensure optimality. Moreover, since only transitions not episodes, are sampled from behavior policy, it doesn't need to calibrate the returns using importance sampling. In summary, Q-learning allows learning Q-values from single transitions instead of complete episodes.

Compared between TD and MC based RL methods, the advantages and disadvantages can be summarized as follows. Instead of directly inferring the return as in MC through simulated trajectories, TD estimating the return R_t using single transitions increases the bias (estimations are always incorrect, especially at the beginning of learning), but reduces the variance: only $r(s, a, s')$ is stochastic, not the value function V^{π}. In brief, TD-based RL methods may have better sample efficiency than MC but worse convergence (suboptimal).

Since classical Q-learning stores one Q-value per state-action pair in a Q-table, so-called tabular RL, it only works for small discrete state and action spaces. To deal with continuous state and action spaces, discretization is needed, which leads to the problem of curse of dimensionality. Due to coupling the RL's sequential decision-making ability under uncertainty with the powerful function approximation capacity of deep neural networks (DNNs), DRL has a myriad of applications in various fields, including intelligent games. Note that, the specific DNN used in DRL is not the focus of DRL models, which can be any type of neural networks, including RNN (Recurrent Neural Networks), CNN (Convolutional Neural Networks), or GNN (Graph Neural Networks), etc. Instead, the core in DRL is how to define the loss function adequately to guide the training of DRL models.

B. DRL categories and algorithms

The existing numerous DRL algorithms in literature lay the basis for various MADRL algorithms, since they can be naturally extended into MADRL field through adapting some MADRL methodologies presented in this article into the existing DRL models. Therefore, the fundamental frameworks and insights developed in DRL provide a robust foundation for understanding and advancing MADRL.

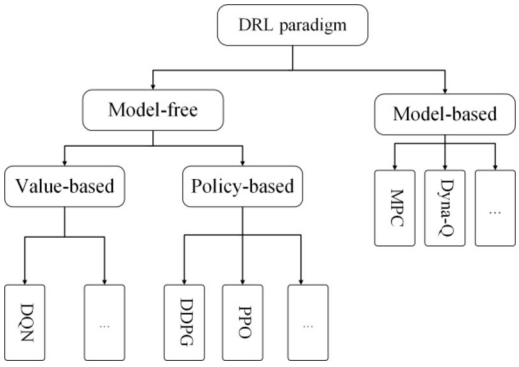

Fig. 7 The category of DRL algorithmsAs shown in Fig. 7, according to whether the dynamics of the environment is known to agent or not, when agent learns a policy, DRL paradigm can be distinguished from two aspects, model-free and model-based. The model-free type can be further categorized into value-based and policy-based methods, depending on whether to directly learn the policy or not. Among these algorithms, DQN is a typical value-based DRL algorithm, and DDPG and PPO are policy-based methods. The typical model-based DRL algorithms are Model Predictive Control (MPC) and Dyna-Q.

Model-free DRL methods

The model-free methods do not require information about the dynamics of the environment, but instead learn the policy from scratch through trial-and-error interactions with a black-box environment. Model-free algorithms can be further divided as value-based and policy-based. Their difference lies in that the former's goal is to approximate the Q-value $Q_\theta\ (s,a)$ for an action a in each possible state s, using DNN parameterized with θ; while the objective of the latter is to directly approximate the policy $\pi_\theta\ (s,a)$ with a DNN, called a parameterized policy.

- Value-based methods

DQN (Deep Q-learning Network) is the basis of many value based DRL models, which approximates every possible state-action (s,a) value function $Q^\pi\ (s,a)$ with a parameterized DNN $Q_\theta\ (s,a)$: $Q_\theta\ (s,a) \approx Q^\pi\ (s,a)$. The DNN in DRL aims to minimize the Mean Squared Error (MSE) between the predicted Q-values and the target value corresponding to the Bellman equation. DQN loss function can be represented as Eq. (15).

$$\mathcal{L}\left(\theta\right) = \mathbb{E}_{\mathcal{D}}\left[\left(r + \gamma \max_{a'} Q_{\theta'}\left(s', a'\right)\right) - Q_\theta\left(s,a\right)\right)^2\Big] \tag{15}$$

The loss function is estimated by sampling a mini-batch of $\mathcal{D}$ including K independent and identically distributed (IID) samples/transitions from the training set.

Technically, DQN uses two skills, i.e., experience replay memory (ERM), and target networks to solve two issues in training DQN model: not IID samples and non-stationary target. Specifically, to avoid correlation between samples, ERM is used to store the huge number of transitions (s,a,r,s'), and transitions used for training DRL models are randomly sampled from ERM. Note that, although, the samples of the mini-batch do not come from the same distribution: some samples may be generated by an old policy π_{θ_0}, while some other samples may be generated by the current policy π_θ, sampling transitions from ERM to train DQN still work, since DQN belongs to the off-policy algorithm. Moreover, to maintain a stationary target for the update, as shown in Eq. (15), instead of computing from the current DNN θ, the target, i.e., $r + \gamma \max_{a'} Q_{\theta'}\left(s', a'\right)$ is calculated from a target network θ' updated only every few thousands of iterations.

After the birth of DQN, numerous DQN variants have occurred. For example, considering DQNs tend to overestimate Q-values, which may lead to learning slower (sample complexity) and less optimal, double DQN (DDQN) (Van Hasselt et al. 2016) is proposed, in which the loss function is shown as Eq. (16).

$$\mathcal{L}\left(\theta\right) = \mathbb{E}_{\mathcal{D}}\left[\left(r + \gamma\, Q_{\theta'}\left(s', argmax_{a'}\, Q_\theta\left(s', a'\right) - Q_\theta\left(s,a\right)\right)^2\right] \tag{16}$$

In a nutshell, as shown in Eq. (16), in DDQN, the next greedy action a' is calculated by the value network θ, i.e., $argmax_{a'}\, Q_\theta\ (s', a')$, while the target Q-value is calculated with the target network θ', i.e., $(r + \gamma\, Q_{\theta'}\ (s', argmax_{a'}\, Q_\theta\ (s', a')))$.

Furthermore, to solve the issues that the predicted Q-values may have high variance, and only a single Q-value is updated, and the state value is not updated, dueling networks (Wang et al. 2016) is proposed, which forces the estimated Q-value to be decomposed into a state value $V_\alpha\ (s)$ and an advantage function $A_\beta\ (s,a)$, shown as Eq. (17).

$$Q_\theta\ (s,a) = V_\alpha\ (s) + A_\beta\ (s,a) \tag{17}$$

Where the parameters α and β are two shared subparts of the DNN. Note that the advantage function $A_\beta\ (s,a)$ has less variance than Q-values. Double dueling DQN (DDDQN) just utilizes the DDQN loss function given as Eq. (16) in dueling network.

Traditional DQN and variants attempt to learn the expectation of the returns, i.e., state value $V^\pi\ (s)$ or state-action value $Q^\pi\ (s,a)$ i.e., their mean values, and select actions with the highest expected return. However, it does not distinguish safe from risky actions. Considering fully characterizing the distribution of return can effectively deal with uncertainty, categorical DQN (Bellemare et al. 2017), attempts to learn the distribution of returns as a discrete probability distribution: make DNN parameterized with θ output Q-value as a discrete probability distribution $\mathcal{Z}_\theta\ (s,a)$ instead of a single Q-value $Q_\theta\ (s,a)$ using Distributional Bellman target and Kullback-Leibler (KL) divergence.

Usually, TD-based DRL can deal with MRP well, in which Markov property means that the current state representation s contains sufficient information to predict the probability of arriving in the next state s' given the chosen action a. But, in most complex decision-making scenarios, the Markovian property can't be met, for example, instead of accessing to the full state of environment, agent can only have partial observations to the environment, or the state transition and policy decision depends not only on the current state, but on the history of states. The Partially-Observable Markov Reward Process (POMRP) occurs. A common way to deal with POMRP in DRL is to utilize the output of the RNN to embed the complete history of state, which is then processed by DRL models, similar as the work, Deep Recurrent Q-Network (Hausknecht and Stone 2015).

DQN and its variants fall into the category of off-policy DRL, which have the benefit of learning from other's experience through ERM, and greatly reduce the number of transitions needed to learn a policy. However, Learning the Q-values in value-based methods (DQN) suffers from the following problems: Q-values are unbounded and high variability which makes learning unstable, and can work only for small discrete action spaces.

- Policy-based methods

Instead of implicitly deriving policy by means of estimating Q-function in value-based algorithms, the policy-based DRL algorithms aims at directly searching for policy π_θ to maximize the expected return over all possible trajectories (episodes), $\tau = (s_0, a_0, r_1, s_1, a_1 \ldots, s_T)$, shown as the Eq. (18).

$$\mathcal{J}(\theta) = \mathbb{E}_{\tau\sim\rho_\theta}\left[R(\tau)\right] \approx \int_\tau \rho_\theta\ (\tau)\, R(\tau)\, d\tau \tag{18}$$

Where ρ_θ is the space of trajectories possible under π_θ. Since directly computing the gradient of the expected return $\mathcal{J}(\theta)$, is intractable, policy Gradient (PG) methods adopt

the surrogate optimization to estimate the gradient: create a surrogate objective whose gradient is locally the same (or has the same direction) as $\mathcal{J}(\theta)$.

The REINFORCE algorithm (Williams 1992) estimates the policy gradient in an unbiased way, given as Eq. (19).

$$\nabla_\theta \mathcal{J}(\theta) = \nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{\tau \sim \rho_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta (s_t, a_t) R(\tau) \right] \tag{19}$$

Where $R(\tau)$is estimated with Monte-Carlo (MC) sampling. However, similar as MC based RL algorithms, REINFORCE algorithm has the following drawbacks: only for episodic tasks, high sample complexity, and high-variance gradient, etc.

Since $Q^{\pi_\theta}(s,a) = \mathbb{E}_{\pi_\theta}[R_t ? s_t = s; a_t = a]$, replacing R_t with $Q^{\pi_\theta}(s,a)$, the $\nabla_\theta \mathcal{J}(\theta)$ will not bring any bias, as shown in Eq. (20).

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{\tau \sim \rho_\theta} \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta (s_t, a_t) Q^{\pi_\theta}(s_t, a_t) \right] \tag{20}$$

In Eq. (14), the policy gradient is defined over complete trajectories τ, however, due to the Markov property of MRP, each step of the trajectory is independent from each other, thus, single transitions can be sampled to estimate the gradient, instead of complete trajectories. Then, Eq. (20) can be reformed as Eq. (21).

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{s \sim \rho_\theta, a \sim \pi_\theta} [\nabla_\theta \log \pi_\theta (s,a) Q^{\pi_\theta}(s_t, a_t)] \tag{21}$$

It is common to replace the true Q-value $Q^{\pi_\theta}(s,a)$ with an estimate $Q_\phi(s,a)$ parameterized with the parameter ϕ, if $Q_\phi(s,a) \approx Q^{\pi_\theta}(s,a), \forall s, a$, shown as Eq. (22).

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{s \sim \rho_\theta, a \sim \pi_\theta} [\nabla_\theta \log \pi_\theta (s,a) Q_\phi(s,a)] \tag{22}$$

The approximated Q-values can be inferred through minimizing the MSE with the true Q-values, shown as Eq. (23).

$$\mathcal{L}(\phi) = \mathbb{E}_{s \sim \rho_\theta, a \sim \pi_\theta} [\left(Q^{\pi_\theta}(s,a) - Q_\phi(s,a) \right]^2] \tag{23}$$

Unusually, the actor-critic architecture is used to approximate the Q-value and optimize the policy selection: the actor $\pi_\theta(s,a)$ implements the policy and selects an action in a state s, while the critic $Q_\phi(s,a)$ estimates the action's value and drives the learning of actor. Due to Critic uses value approximator to replace episode estimates, it has less sample complexity. Thus, actor-critic framework has been widely used in policy-based DRL algorithms, but at the cost of some bias.

Technically, to train the critic, various sampling methods introduced above can be used to estimate the true Q-value, including Monte-Carlo critic that samples the complete episode, SARSA critic that sample (s, a, r, s', a') transitions using the current points, and Q-learning critic that samples (s, a, r, s') transitions from ERM.

To alleviate the high variance of the estimated Q-values, the advantage actor-critic uses the value of a state as the baseline, in which the policy gradient can be represented as Eq. (24).

$$\begin{aligned}\nabla_\theta \mathcal{J}(\theta) &= \mathbb{E}_{\mathrm{s}\sim\rho_\theta,\mathrm{a}\sim\pi_\theta}\left[\nabla_\theta \log\pi_\theta(\mathrm{s},\mathrm{a})\left(\mathrm{Q}^{\pi_\theta}(\mathrm{s},\mathrm{a}) - \mathrm{V}^{\pi_\theta}(\mathrm{s})\right)\right] \\ &= \mathbb{E}_{\mathrm{s}\sim\rho_\theta,\mathrm{a}\sim\pi_\theta}\left[\nabla_\theta \log\pi_\theta(\mathrm{s},\mathrm{a})\,\mathrm{A}^{\pi_\theta}(\mathrm{s},\mathrm{a})\right]\end{aligned} \tag{24}$$

In summary, PG based methods can be represented as the general form shown as Eq. (25).

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{s_t\sim\rho_\theta, a_t\sim\pi_\theta}\left[\nabla_\theta \log\pi_\theta(s_t,a_t)\,\psi_t\right] \tag{25}$$

The different variants of PG varying with the different ψ_t formulations can balance the tradeoff between model's bias and variance.

Advantage actor-critic (A2C) employs an n-step advantage actor-critic architecture where the Q-value of the action (s_t, a_t) is approximated by the n-step return. Therefore, A2C is strictly on-policy: neither ERM can be used to deal with the correlated inputs, nor can an uncorrelated batch of transitions be obtained by acting sequentially with a single agent. To solve the issue, A2C enables multiple actors with the same weights θ given by the global network to interact in parallel with different copies of environment, so-called distributed learning where each worker starts from different state to gather uncorrelated transitions, and, sometimes all workers' policies are synchronously updated by the global network. That is, A2C synchronizes the workers, i.e., it waits for the multiple workers to finish their jobs: gather transitions, and train actor and critic of each work, then merge the gradients of all works, and update the global actor and critic networks.

Instead, A3C is the asynchronous version of A2C, in which, when available, the partial gradients calculated by workers are immediately applied to the global networks (Mnih et al. 2016).

The DRL algorithms mentioned above including Actor-critic methods belongs to stochastic policy gradients: The learned policy $\pi_\theta(s_t,a_t)$ is stochastic, which may generate high variance in the gradients and in the obtained returns. Furthermore, they are strictly on-policy: the critic must be trained with the transitions generated by the current version of the actor, that is, past transitions cannot be reused for training (no ERM). Supposing the policy is deterministic, i.e. it takes a single action in state s, deterministic policy $\mu_\theta(s)$, then, the deterministic policy gradient becomes Eq. (26).

$$\begin{aligned}\nabla_\theta \mathcal{J}(\theta) &= \mathbb{E}_{s\sim\rho_\theta}\left[\nabla_\theta Q^{\mu_\theta}(s,\mu_\theta(s))\right] \\ &= \mathbb{E}_{s\sim\rho_\theta}\left[\nabla_a Q^{\mu_\theta}(s,a)\big|_{a=\mu_\theta(s)} \times \nabla_\theta \mu_\theta(s)\right]\end{aligned} \tag{26}$$

Note that the second term in Eq. (21) comes from using the chain rule to decompose the gradient of $Q^{\mu_\theta}(s,\mu_\theta(s))$. As always, the true Q-value $Q^{\mu_\theta}(s,a)$ is estimated with an estimate $Q_\phi(s,a)$ using DNN with parameter ϕ, as work in DDPG (Deep Deterministic Policy Gradient) (Lillicrap et al. 2015). DDPG is an off-policy actor-critic method, in which the actor and critic are respectively trained with Eqs. (27) and (28).

$$\nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{s}\sim\boldsymbol{\rho}_{\boldsymbol{\theta}}}\left[\nabla_{\boldsymbol{\theta}} \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s}) \times \nabla_{\boldsymbol{a}} \boldsymbol{Q}_{\boldsymbol{\phi}}(\boldsymbol{s},\boldsymbol{a})\big|_{\boldsymbol{a}=\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s})}\right] \tag{27}$$

$$\mathcal{L}(\phi) = \mathbb{E}_{s\sim\rho_\theta}\left[\left(r(s,\mu_\theta(s)) + \gamma Q_\phi(s', \mu_\theta(s')) - Q_\phi(s,\mu_\theta(s))\right)^2\right] \quad (28)$$

In summary, the actor μ_θ learns using sampled transitions; the critic Q_ϕ can be implemented with DQN on sampled transitions. Note that since the policy in DDPG is deterministic, sampling states from other policy different from current policy won't affect the deterministic policy gradient, thus ERM can be used by DDPG. Due to the fact that DDPG only outputs the deterministic policy, to ensure exploration, the exploratory noise should be added to the deterministic action.

To address functional approximation error in DDPG, including overestimation of Q-values, TD3 (Twin Delayed Deep Deterministic policy gradient) (Fujimoto et al. 2018) is proposed.

In the above algorithms, DQN and DDPG are off-policy methods, in which ERM can be applied; A2C and A3C are on-policy, in which the distributed learning has to be employed. However, all these methods suffer from parameter brittleness, i.e., it is extremely difficult to infer the right hyperparameters θ in DNN. Usually, once the policy gradient $\nabla_\theta \mathcal{J}(\boldsymbol{\theta})$ is estimated, the weights θ can be updated in the direction of the inferred gradient: $\theta \leftarrow \theta + \eta \nabla_\theta \mathcal{J}(\boldsymbol{\theta})$. However, when in updating θ, two challenges exist. First, choosing an appropriate learning rate η is extremely difficult in DRL: too small learning rate makes DNN converge very slowly, requiring a lot of samples to converge (sample complexity); too large learning rate may totally destroy the policy (instability), even lead to policy collapse. Second, the policy gradient is unbiased only under the condition that the critic Q_ϕ can accurately approximate the true Q-values of the current policy.

Instead of the pre-defined learning rate, Trust Region Policy Optimization (TRPO) (Schulman et al. 2015) directly searches the DNN weights in the neighborhood of the current parameters θ, through modeling it as a constrained optimization problem: the agent still wants to maximize the return of the policy, but with a constraint that the new policy should be as close as possible from its previous value. Formally, besides maximizing the expected return that is further decomposed into two parts: the obtained return under current/old policy $\pi_{\theta_{old}}$, and the advantage of following new policy π_θ compared to following the current policy $\pi_{\theta_{old}}$, the KL divergence between the distributions $\pi_{\theta_{old}}$ and π_θ must be below a threshold. Different from using KL divergence as constraint, PPO (Schulman et al. 2017) simply clips objective to ensure that the importance sampling weight stays around one, so that the new policy is not very different from the old one.

In all of the algorithms mentioned, the objective is to search policy to maximize the return. In other words, they mainly care about exploitation, and only utilize the external exploration, for example, DQN and variants use ϵ-greedy or softmax on the Q-values, DDPG adds exploratory noise on policy, etc. However, intrinsic variety should be beneficial when facing POMRP scenarios like intelligent game: it can counteract the uncertainty about the complex environment.

The maximum entropy RL framework defines the new objective function as Eq. (29), which implies that policy is still sought to maximize the returns while being as stochastic as possible, depending on the parameter α.

$$\pi^* = \underset{\pi}{argmax}\,\mathbb{E}_\pi\left[\sum_t \gamma^t r(s_t, a_t, s_{t+1}) + \alpha H(\pi(s_t))\right] \quad (29)$$

The entropy of a policy in a state s_t is defined by the expected negative log-likelihood of the policy, shown as Eq. (30).

$$H\left(\pi_{\theta}\left(s_t\right)\right) = \mathbb{E}_{a\sim\pi_{\theta}\left(s_t\right)}\left[-log\pi_{\theta}\left(s_t, a\right)\right] \tag{30}$$

Essentially, the second term in Eq. (30) severs as the entropy regularization to encourage exploration and help prevent early convergence to sub-optimal policies.

Following the above maximum entropy RL framework, Soft Actor-Critic (SAC) is an off-policy actor-critic architecture. Specifically, the critic is trained with the following loss function, shown as Eq. (31).

$$\mathcal{L}\left(\phi\right) = \mathbb{E}_{s_t,a_t,s_{t+1}\sim\rho_{\theta}}\left[\left(r_{t+1} + \gamma\, Q_{\phi}\left(s_{t+1}, a_{t+1}\right) - \log\pi_{\theta}\left(s_{t+1}, a_{t+1}\right) - Q_{\phi}\left(s_t, a_t\right)\right)^2\right] \tag{31}$$

The actor learns a Gaussian policy that becomes close to a softmax over the soft Q-values, shown as Eq. (32).

$$\nabla_{\theta}\mathcal{L}\left(\theta\right) = \mathbb{E}_{s,a}\left[\alpha\nabla_{\theta}\log\pi_{\theta}\left(s, a\right) - Q_{\phi}\left(s, a\right)\right],\ \pi_{\theta}\left(s, a\right) \propto exp\frac{Q_{\phi}\left(s, a\right)}{\alpha} \tag{32}$$

The enhanced exploration strategy through maximum entropy DRL in SAC allows learning robust and varied policies that can cope with changes in the environment.

A key challenge is how to best combine the advantages of value and policy based DRL approaches, while mitigating their shortcomings. Nachum et al. (2017) establish a connection between value and policybased RLs, which shows that softmax consistent action values correspond to optimal entropy regularized policy probabilities along any action sequence. Then, a RL algorithm, Path Consistency Learning (PCL) is developed, which minimizes soft consistency error along multi-step action sequences extracted from both on- and off-policy traces. The work argues that PCL can be interpreted as generalizing both actor-critic and Q-learning algorithms.

Model-based DRL methods

The benefit of model-free RL, MFDRL methods lies in that they need not to know anything about the dynamics of the environment, i.e., $p\left(s_{t+1}|s_t, a_t\right)$ and $r\left(s_t, a_t, s_{t+1}\right)$: The agent just samples transitions $\left(s, a, r, s'\right)$ and update Q-values or a policy network. However, MF methods are very slow (sample complexity): as they make no assumption, and have to learn everything by trial-and-error from scratch.

If the agent can reasonably model the environment, she not only can speed up learning, but also can make more effective decision through planning ahead. That is the core idea in model-based DRL, i.e., MBDRL. For example, in strategy-like games, learning the environment (world model) should be the part of the strategy of players. The formulated model about environment is called the dynamics model, or the transition model, which aims to answer what would happen if the agent does that action? Theoretically, learning the world model by agent is straightforward: collect enough transitions $\left(s_t, a_t, s_{t+1}, r_{t+1}\right)$ using a random agent (or during learning) and train a model to predict the next state and reward. Generally, there are two paradigms to learn the world model: Model-based Augmented

Model-Free (MBMF) generates imaginary transitions/rollouts used to train a model-free algorithm; Model-based planning uses the learned world model to plan actions that maximize the DRL objective.

The first work of MBMF is Dyna-Q (Sutton 1991), in which the MF algorithm (e.g. Q-learning) learns from transitions (s, a, r, s') sampled both with real experience through interaction with the environment, and simulated experience by the world model. If the simulated transitions are good enough, the MF algorithm can converge using much less real transitions, thereby reducing its sample complexity.

The pioneering work of model based planning is Model predictive control (MPC) (Nagabandi et al. 2018), which re-plans at each time step and executes only the first planned action, to combat against the issue that, when facing long horizon, imperfect world model will accumulate (drift) and lead to completely wrong trajectories.

Based on the core ideas of Dyna-Q and MPC, various MBDRL algorithms have been proposed. Moerland et al. (2023) provide a comprehensive survey on model-based RL methods including the world model learning methods and integration of planning-learning integration. In summary, MBDRL can only work when the model is close to real model of environments, especially for long trajectories or probabilistic MDPs.

Author contributions Yiqin Wang is responsible for Conceptualization; Writing—original draft; Writing—review & Editing; Formal analysis; Yufeng Wang is responsible for Conceptualization; Methodology; Writing—review & Editing; Formal analysis; Feng Tian is responsible for Conceptualization; Supervision; Jianhua Ma is responsible for Supervision; Investigation; Validation; Qun Jin is is responsible for Investigation; Review & Editing; Validation;

Data availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

Agarwal A, Kumar S, Sycara K et al (2020) Learning transferable cooperative behavior in multi-agent teams. In: Proceedings of the 19th international conference on autonomous agents and multiagent systems. International Foundation for Autonomous Agents and Multiagent Systems, AAMAS '20, pp 1741–1743

Amos-Binks A, Weber BS (2023) Risk management: anticipating and reacting in starcraft. In: Proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment, pp 13–22

Bai Y, Jin C (2020) Provable self-play algorithms for competitive reinforcement learning. In: International conference on machine learning, PMLR, pp 551–560

Bai Y, Jin C, Wang H et al (2021) Sample-efficient learning of Stackelberg equilibria in general-sum games. Adv Neural Inf Process Syst 34:25799–25811

Balduzzi D, Garnelo M, Bachrach Y et al (2019) Open-ended learning in symmetric zero-sum games. In: International conference on machine learning, PMLR, pp 434–443
Barambones J, Cano-Benito J, S, ́anchez-Rivero I et al (2022) Multiagent systems on virtual games: a systematic mapping study. IEEE Trans Games 15(2):134–147
Bellemare MG, Dabney W, Munos R (2017) A distributional perspective on reinforcement learning. In: International conference on machine learning, PMLR, pp 449–458
Berner C, Brockman G, Chan B et al (2019) Dota 2 with large scale deep reinforcement learning. arXiv preprint arXiv:191206680
Bettini M, Prorok A, Moens V (2024) Benchmarl: benchmarking multi-agent reinforcement learning. J Mach Learn Res 25(217):1–10
Bian Y, Rong Y, Xu T et al (2022) Energy-based learning for cooperative games, with applications to valuation problems in machine learning. In: International conference on learning representation
Bou A, Bettini M, Dittert S et al (2024) TorchRL: a data-driven decision-making library for pytorch. In: The Twelfth international conference on learning representations
Canese L, Cardarilli GC, Di Nunzio L et al (2021) Multi-agent reinforcement learning: a review of challenges and applications. Appl Sci 11(11):4948
Carr S, Jansen N, Junges S et al (2023) Safe reinforcement learning via shielding under partial observability. In: Proceedings of the AAAI conference on artificial intelligence, pp 14748–14756
Chan L, Hogaboam L, Cao R (2022) Artificial intelligence in video games and esports. Applied artificial intelligence in business: concepts and cases. Springer, pp 335–352
Chen H, Covert IC, Lundberg SM et al (2023) Algorithms to estimate Shapley value feature attributions. Nat Mach Intell 5(6):590–601
Christianos F, Papoudakis G, Albrecht SV (2023) Pareto actor-critic for equilibrium selection in multi-agent reinforcement learning. arXiv preprint arXiv:2209.14344
Chu T, Chinchali S, Katti S (2020) Multi-agent reinforcement learning for networked system control. In: International conference on learning representation
Cui K, Tahir A, Ekinci G et al (2022) A survey on large-population systems and scalable multi-agent reinforcement learning. arXiv preprint arXiv:220903859
Das A, Gervet T, Romoff J et al (2019) Tarmac: targeted multi-agent communication. In: International conference on machine learning, PMLR, pp 1538–1546
Dazeley R, Vamplew P, Cruz F (2023) Explainable reinforcement learning for broad-xai: a conceptual framework and survey. Neural Comput Appl 35(23):16893–16916
De Witt CS, Gupta T, Makoviichuk D et al (2020) Is independent learning all you need in the Starcraft multi-agent challenge? ArXiv Preprint ArXiv:201109533
Ding Z, Huang T, Lu Z (2020) Learning individually inferred communication for multi-agent Cooperation. Adv Neural Inf Process Syst 33:22069–22079
Ding D, Wei CY, Zhang K et al (2022) Independent policy gradient for large-scale markov potential games: sharper rates, function approximation, and game-agnostic convergence. In: International Conference on Machine Learning, PMLR, pp 5166–5220
do Nascimento Silva V, Chaimowicz L (2015) On the development of intelligent agents for moba games. In: 2015 14th Brazilian symposium on computer games and digital entertainment (SBGames), IEEE, pp 142–151
Dwivedi R, Dave D, Naik H et al (2023) Explainable Ai (xai): core ideas, techniques, and solutions. ACM-CSUR 55(9):1–33
Ellis B, Cook J, Moalla S et al (2024) Smacv2: an improved benchmark for cooperative multi-agent reinforcement learning. Adv Neural Inf Process Syst 36:37567–37593
ElSayed-Aly I, Bharadwaj S, Amato C et al (2021) Safe multi-agent reinforcement learning via shielding. In: Proceedings of the 20th international conference on autonomous agents and multiagent systems
Ferdous R, Kifetew F, Prandi D et al (2022) Towards agent-based testing of 3d games using reinforcement learning. In: Proceedings of the 37th IEEE/ACM international conference on automated software engineering, pp 1–8
Foerster J, Assael IA, De Freitas N et al (2016) Learning to communicate with deep multi-agent reinforcement learning. Adv Neural Inf Process Syst 29
Foerster J, Farquhar G, Afouras T et al (2018) Counterfactual multi-agent policy gradients. In: Proceedings of the AAAI conference on artificial intelligence
Fu H, Liu W, Wu S et al (2021) Actor-critic policy optimization in a large-scale imperfect-information game. In: International conference on learning representations
Fu Q, Ai X, Yi J et al (2023) Learning heterogeneous agent cooperation via multiagent league training. IFAC-PapersOnLine 56(2):3033–3040
Fujimoto S, Hoof H, Meger D (2018) Addressing function approximation error in actor-critic methods. In: International conference on machine learning, PMLR, pp 1587–1596

Gero KI, Ashktorab Z, Dugan C et al (2020) Mental models of ai agents in a cooperative game setting. In: Proceedings of the 2020 chi conference on human factors in computing systems, pp 1–12
Ghorbani A, Zou J (2019) Data shapley: equitable valuation of data for machine learning. In: International conference on machine learning, PMLR, pp 2242–2251
Gogineni K, Wei P (2023) Scalability bottlenecks in multi-agent reinforcement learning systems. In: FastPath 2023: International workshop on performance analysis of machine learning systems
Gorsane R, Mahjoub O, de Kock RJ et al (2022) Towards a standardised performance evaluation protocol for cooperative marl. Adv Neural Inf Process Syst 35:5510–5521
Gronauer S, Diepold K (2022) Multi-agent deep reinforcement learning: a survey. Artif Intell Rev 55(2):895–943
Gu S, Kuba JG, Chen Y et al (2023) Safe multi-agent reinforcement learning for multi-robot control. Artif Intell 319:103905
Gu S, Yang L, Du Y et al (2024) A review of safe reinforcement learning: methods, theories, and applications. IEEE Trans Pattern Anal Mach Intell 46(12):11216–11235
Guo X, Shi D, Fan W (2023) Scalable communication for multi-agent reinforcement learning via transformer-based email mechanism. In: Proceedings of the thirty-second international joint conference on artificial intelligence, IJCAI '23
Gupta N, Srinivasaraghavan G, Mohalik S et al (2023) Hammer: Multi-level coordination of reinforcement learning agents via learned messaging. Neural Comput Appl 1–16
Han L, Zhu Q, Sheng J et al (2024) Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models. Nat Mach Intell 6(7):787–798
Hansen EA, Bernstein DS, Zilberstein S (2004) Dynamic programming for partially observable stochastic games. In: AAAI, pp 709–715
Hao J, Yang T, Tang H et al (2023) Exploration in deep reinforcement learning: from single-agent to multiagent domain. IEEE Trans Neural Networks Learn Syst 35(7):8762–8782
Hausknecht M, Stone P (2015) Deep recurrent q-learning for partially observable mdps. In: 2015 AAAI fall symposium series
Heinrich J, Lanctot M, Silver D (2015) Fictitious self-play in extensive-form games. In: International conference on machine learning, PMLR, pp 805–813
Hernandez D, Denamganaï K, Gao Y et al (2019) A generalized framework for self-play training. In: 2019 IEEE Conference on Games (CoG), IEEE, pp 1–8
Hernandez D, Denamganai K, Devlin S et al (2021) A comparison of self-play algorithms under a generalized framework. IEEE Trans Games 14(2):221–231
Hernandez-Leal P, Kaisers M, Baarslag T et al (2017) A survey of learning in multiagent environments: dealing with non-stationarity. arXiv preprint arXiv:170709183
Heuillet A, Couthouis F, Díaz-Rodríguez D N (2022) Collective explainable Ai: explaining cooperative strategies and agent contribution in multiagent reinforcement learning with Shapley values. IEEE Comput Intell Mag 17(1):59–71
Hickling T, Zenati A, Aouf N et al (2023) Explainability in deep reinforcement learning: a review into current methods and applications. ACM-CSUR 56(5):1–35
Hong Y, Jin Y, Tang Y (2022) Rethinking individual global max in cooperative multi-agent reinforcement learning. Adv Neural Inf Process Syst 35:32438–32449
Hu J, Hu S, Liao S (2021) Policy regularization via noisy advantage values for cooperative multi-agent actor-critic methods. arXiv preprint arXiv:210614334
Hu B, Zhao C, Zhang P et al (2023a) Enabling intelligent interactions between an agent and an llm: a reinforcement learning approach. arXiv preprint arXiv:230603604
Hu J, Wang S, Jiang S et al (2023b) Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning. In: The Second Blogpost Track at ICLR 2023
Hu S, Zhong Y, Gao M et al (2023c) Marllib: a scalable and efficient multi-agent reinforcement learning library. J Mach Learn Res 24(315):1–23
Hu Z, Liu H, Xiong Y et al (2024) Promoting human-ai interaction makes a better adoption of deep reinforcement learning: a real-world application in game industry. Multimed Tools Appl 83(2):6161–6182
Huang X (2023) Starcraft adversary-agent challenge for pursuit–evasion game. J Franklin Inst 360(15):10893–10916
Jaderberg M, Czarnecki WM, Dunning I et al (2019) Human-level performance in 3d multiplayer games with population-based reinforcement learning. Science 364(6443):859–865
Ji J, Zhang B, Zhou J et al (2023) Safety gymnasium: a unified safe reinforcement learning benchmark. Adv Neural Inf Process Syst 36:18964–18993
Jiang J, Lu Z (2018) Learning attentional communication for multi-agent cooperation. Adv Neural Inf Process Syst 31

Jiang J, Dun C, Huang T et al (2020) Graph convolutional reinforcement learning. In: International conference on learning representation
Jiang K, Liu W, Wang Y et al (2023) Credit assignment in heterogeneous multi-agent reinforcement learning for fully cooperative tasks. Appl Intell 53(23):29205–29222
Kim D, Moon S, Hostallero D et al (2019) Learning to schedule communication in multi-agent reinforcement learning. In: International conference on learning representation
Kim W, Park J, Sung Y (2020) Communication in multi-agent reinforcement learning: Intention sharing. In: International conference on learning representations
Kuba JG, Chen R, Wen M et al (2022) Trust region policy optimisation in multi-agent reinforcement learning. In: International conference on learning representations
Kumar K, Veena N, Aravind T et al (2025) Game-changing intelligence: unveiling the societal impact of artificial intelligence in game software. Entertain Comput 52:100862
Kurach K, Raichuk A, Sta´nczyk P et al (2020) Google research football: A novel reinforcement learning environment. In: Proceedings of the AAAI conference on artificial intelligence, pp 4501–4510
Kwon M, Xie SM, Bullard K et al (2023) Reward design with language models. In:The eleventh international conference on learning representations
Lanctot M, Lockhart E, Lespiau JB et al (2019) Openspiel: a framework for reinforcement learning in games. arXiv preprint arXiv:190809453
Leonardos S, Overman W, Panageas I et al (2022) Global convergence of multi-agent policy gradient in markov potential games. In: ICLR 2022 Workshop on gamification and multiagent solutions
Li J, Koyamada S, Ye Q et al (2020) Suphx: Mastering mahjong with deep reinforcement learning. arXiv preprint arXiv:200313590
Li J, Kuang K, Wang B et al (2021) Shapley counterfactual credits for multi-agent reinforcement learning. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp 934–942
Li T, Zhao Y, Zhu Q (2022) The role of information structures in game-theoretic multi-agent learning. Annu Rev Control 53:296–314
Li S, Xu J, Dong H et al (2023a) The fittest wins: a multistage framework achieving new Sota in Vizdoom competition. IEEE Trans Games 16(1):225–234
Li Y, Xiong K, Zhang Y et al (2023) Jiangjun: mastering Xiangqi by tackling non-transitivity in two-player zero-sum games. arXiv preprint arXiv:2308.04719
Lillicrap TP, Hunt JJ, Pritzel A et al (2015) Continuous control with deep reinforcement learning. arXiv preprint arXiv:150902971
Lin T, Huh J, Stauffer C et al (2021) Learning to ground multi-agent communication with autoencoders. Adv Neural Inf Process Syst 34:15230–15242
Lin F, Huang S, Pearce T et al (2023) Tizero: Mastering multi-agent football with curriculum learning and self-play. In: Proceedings of the 2023 International conference on autonomous agents and multiagent systems. International Foundation for Autonomous Agents and Multiagent Systems, AAMAS '23, p 67–76
Liu Y, Wang W, Hu Y et al (2020) Multi-agent game abstraction via graph attention neural network. In: Proceedings of the AAAI conference on artificial intelligence, pp 7211–7218
Liu C, Geng N, Aggarwal V et al (2021) Cmix: Deep multi-agent reinforcement learning with peak and average constraints. In: Machine learning and knowledge discovery in databases. Research Track: European Conference, ECML PKDD 2021, pp 157–173
Liu Q, Szepesv´ari C, Jin C (2022) Sample-efficient reinforcement learning of partially observable Markov games. Adv Neural Inf Process Syst 35:18296–18308
Liu Z, Wan L, Sui X et al (2023) Deep hierarchical communication graph in multi-agent reinforcement learning. In: IJCAI, pp 208–216
Long Q, Zhou Z, Gupta A et al (2020) Evolutionary population curriculum for scaling multi-agent reinforcement learning. In: International conference on learning representation
Lowe R, Wu YI, Tamar A et al (2017) Multi-agent actor-critic for mixed cooperative-competitive environments. Adv Neural Inf Process Syst 30
Maddila P, Eric C, Chabrier P et al (2024) APE: an anti-poaching multi-agent reinforcement learning benchmark. In: Seventeenth European workshop on reinforcement learning
Mahajan A, Rashid T, Samvelyan M et al (2019) Maven: multi-agent variational exploration. Adv Neural Inf Process Syst 32
Mao W, Qiu H, Wang C et al (2022) A mean-field game approach to cloud resource management with function approximation. Adv Neural Inf Process Syst 35:36243–36258
Mnih V, Badia AP, Mirza M et al (2016) Asynchronous methods for deep reinforcement learning. In: International conference on machine learning, PMLR, pp 1928–1937

Moerland TM, Broekens J, Plaat A et al (2023) Model-based reinforcement learning: a survey. Found Trends Mach Learn 16(1):1–118
Moravčík M, Schmid M, Burch N et al (2017) Deepstack: expert-level artificial intelligence in heads-up no-limit poker. Science 356(6337):508–513. https://doi.org/10.1126/science.aam6960
Mycielski J (1992) Games with perfect information. Handb Game Theory Econ Appl 1:41–70
Nachum O, Norouzi M, Xu K et al (2017) Bridging the gap between value and policy based reinforcement learning. Adv Neural Inf Process Syst 30
Nagabandi A, Kahn G, Fearing RS et al (2018) Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In: 2018 IEEE international conference on robotics and automation (ICRA), IEEE, pp 7559–7566
Nayak S, Choi K, Ding W et al (2023) Scalable multi-agent reinforcement learning through intelligent information aggregation. In: International conference on machine learning, PMLR, pp 25817–25833
Nguyen TT, Nguyen ND, Nahavandi S (2020) Deep reinforcement learning for multiagent systems: a review of challenges, solutions, and applications. IEEE Trans Cybern 50(9):3826–3839
Niu Y, Paleja RR, Gombolay MC (2021) Multi-agent graph-attention communication and teaming. In: AAMAS
Oh I, Rho S, Moon S et al (2021) Creating pro-level Ai for a real-time fighting game using deep reinforcement learning. IEEE Trans Games 14(2):212–220
Oroojlooy A, Hajinezhad D (2023) A review of cooperative multi-agent deep reinforcement learning. Appl Intell 53(11):13677–13722
Pan X, Liu M, Zhong F et al (2022) Mate: benchmarking multi-agent reinforcement learning in distributed target coverage control. Adv Neural Inf Process Syst 35:27862–27879
Papoudakis G, Christianos F, Rahman A et al (2019) Dealing with non-stationarity in multi-agent deep reinforcement learning. arXiv preprint arXiv:190604737
Papoudakis G, Christianos F, Schafer L et al (2021) Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In: Thirty-fifth conferenceon neural information processing systems datasets and benchmarks track (Round1)
Peng P, Wen Y, Yang Y et al (2017) Multiagent bidirectionally-coordinated nets: emergence of human-level coordination in learning to play starcraft combat games. arXiv preprint arXiv:170310069
Perolat J, De Vylder B, Hennes D et al (2022) Mastering the game of stratego with model-free multiagent reinforcement learning. Science 378(6623):990–996
Qiu Y, Jin Y, Yu L Safe multi-agent reinforcement learning via dynamic shielding. In: 2024 IEEE Conference on Artificial, Intelligence (CAI), IEEE, pp 1254–1257
Rangwala M, Williams R (2020) Learning multi-agent communication through structured attentive reasoning. Adv Neural Inf Process Syst 33:10088–10098
Rashid T, Samvelyan M, De Witt CS et al (2020b) Monotonic value function factorisation for deep multi-agent reinforcement learning. J Mach Learn Res 21(178):1–51
Rashid T, Farquhar G, Peng B et al (2020a) Weighted Qmix: expanding monotonic value function factorisation for deep multi-agent reinforcement learning. Adv Neural Inf Process Syst 33:10199–10210
Roth AE, Erev I (1995) Learning in extensive-form games: experimental data and simple dynamic models in the intermediate term. Games Econ Behav 8(1):164–212
Rozemberczki B, Watson L, Bayer P et al (2022) The shapley value in machine learning. In: Proceedings of the 31st international joint conference on artifical intelligence, IJCAI-ECAI 2022. International joint conferences on Artificial Intelligence Organization, pp 5572–5579. https://doi.org/10.24963/ijcai.2022/778
Schrittwieser J, Antonoglou I, Hubert T et al (2020) Mastering Atari, go, chess and Shogi by planning with a learned model. Nature 588(7839):604–609
Schulman J, Levine S, Abbeel P et al (2015) Trust region policy optimization. In: International conference on machine learning, PMLR, pp 1889–1897
Schulman J, Chen X, Abbeel P (2017) Equivalence between policy gradients and soft q-learning. arXiv preprint arXiv:170406440
Shapley LS (1953) Stochastic games. Proc Natl Acad Sci 39(10):1095–1100
Shapley LS et al (1953) A value for n-person games
Silver D, Huang A, Maddison CJ et al (2016) Mastering the game of go with deep neural networks and tree search. Nature 529(7587):484–489
Silver D, Schrittwieser J, Simonyan K et al (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359
Silver D, Hubert T, Schrittwieser J et al (2018) A general reinforcement learning algorithm that masters chess, Shogi, and go through self-play. Science 362(6419):1140–1144
Skrynnik A, Andreychuk A, Borzilov A et al (2024) Pogema: a benchmark platform for cooperative multi-agent navigation. arXiv preprint arXiv:240714931

Son K, Kim D, Kang WJ et al (2019) Qtran: learning to factorize with transformation for cooperative multi-agent reinforcement learning. In: International conference on machine learning, PMLR, pp 5887–5896
Souchleris K, Sidiropoulos GK, Papakostas GA (2023) Reinforcement learning in game industry—review, prospects and challenges. Appl Sci 13(4):2443
Subramanian J, Sinha A, Mahajan A (2023) Robustness and sample complexity of model-based marl for general-sum Markov games. Dyn Games Appl 13(1):56–88
Sukhbaatar S, Fergus R et al (2016) Learning multiagent communication with backpropagation. Adv Neural Inf Process Syst 29
Sun C, Huang S, Pompili D (2024) Llm-based multi-agent reinforcement learning: current and future directions. arXiv preprint arXiv:240511106
Sundararajan M, Taly A, Yan Q (2017) Axiomatic attribution for deep networks. In: International conference on machine learning, PMLR, pp 3319–3328
Sunehag P, Lever G, Gruslys A et al (2018) Value-decomposition networks for cooperative multi-agent learning based on team reward. International Foundation for Autonomous Agents and Multiagent Systems, AAMAS '18, pp 2085–2087
Sutton RS (1991) Dyna, an integrated architecture for learning, planning, and reacting. ACM Sigart Bull 2(4):160–163
Tampuu A, Matiisen T, Kodelja D et al (2017) Multiagent Cooperation and competition with deep reinforcement learning. PLoS ONE 12(4):e0172395
Tembine H, Zhu Q, Başar T (2013) Risk-sensitive mean-field games. IEEE Trans Autom Control 59(4):835–850
Terry J, Black B, Grammel N et al (2021) Pettingzoo: gym for multi-agent reinforcement learning. Adv Neural Inf Process Syst 34:15032–15043
Tung TY, Kobus S, Roig JP et al (2021) Effective communications: a joint learning and communication framework for multi-agent reinforcement learning over noisy channels. IEEE J Sel Areas Commun 39(8):2590–2603
Van Hasselt H, Guez A, Silver D (2016) Deep reinforcement learning with double q-learning. In: Proceedings of the AAAI conference on artificial intelligence
Vaswani A, Shazeer N, Parmar N et al (2017) Attention is all you need. Adv Neural Inf Process Syst 30
Vinyals O, Babuschkin I, Czarnecki WM et al (2019) Grandmaster level in Starcraft Ii using multi-agent reinforcement learning. Nature 575(7782):350–354
Von Stackelberg H (2010) Market structure and equilibrium. Springer Science & Business Media
Wang Z, Schaul T, Hessel M et al (2016) Dueling network architectures for deep reinforcement learning. In: International conference on machine learning, PMLR, pp 1995–2003
Wang J, Zhang Y, Kim TK et al (2020a) Shapley q-value: a local reward approach to solve global reward games. In: Proceedings of the AAAI conference on artificial intelligence, pp 7285–7292
Wang L, Yang Z, Wang Z (2020b) Breaking the curse of many agents: provable mean embedding q-iteration for mean-field reinforcement learning. In: International conference on machine learning, PMLR, pp 10092–10103
Wang R, He X, Yu R et al (2020c) Learning efficient multi-agent communication: an information bottleneck approach. In: International conference on machine learning, PMLR, pp 9908–9918
Wang T, Dong H, Lesser V et al (2020d) Roma: multi-agent reinforcement learning with emergent roles. In: Proceedings of the 37th international conference on machine learning, ICML'20
Wang T, Wang J, Zheng C et al (2020e) Learning nearly decomposable value functions via communication minimization. In: International conference on learning representation
Wang J, Ren Z, Liu T et al (2021a) QPLEX: Duplex dueling multi-agent q-learning. In: International conference on learning representations
Wang X, Chen Y, Zhu W (2021b) A survey on curriculum learning. IEEE Trans Pattern Anal Mach Intell 44(9):4555–4576
Wang Y, Han B, Wang T et al (2021c) DOP: Off-policy multi-agent decomposed policy gradients. In: International conference on learning representations
Wang J, Xue D, Zhao J Mastering the game of 3v3 snakes with rule-enhanced multi-agent reinforcement learning. In: 2022 IEEE Conference on Games (CoG), IEEE, pp 229–236
Wang J, Zhang Y, Gu Y et al (2022b) Shaq: incorporating Shapley value theory into multi-agent q-learning. Adv Neural Inf Process Syst 35:5941–5954
Wang X, Zhang Z, Zhang W (2022c) Model-based multi-agent reinforcement learning: Recent progress and prospects. arXiv preprint arXiv:220310603
Wang L, Ma C, Feng X et al (2024) A survey on large Language model based autonomous agents. Front Comput Sci 18(6):186345
Wen G, Fu J, Dai P et al (2021) Dtde: a new cooperative multi-agent reinforcement learning framework. Innovation 2(4)

Wen M, Kuba J, Lin R et al (2022) Multi-agent reinforcement learning is a sequence modeling problem. Adv Neural Inf Process Syst 35:16509–16521
Whiteson S, Samvelyan M, Rashid T et al (2019) The starcraft multi-agent challenge. In: Proceedings of the international joint conference on autonomous agents and multiagent systems, AAMAS, pp 2186–2188
Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach Learn 8:229–256
Wolpert DH, Tumer K (2001) Optimal payoff functions for members of collectives. Adv Complex Syst 4(02n03):265–279
Wurman PR, Barrett S, Kawamoto K et al (2022) Outracing champion Gran turismo drivers with deep reinforcement learning. Nature 602(7896):223–228
Xu Z, Yu C, Fang F et al (2024) Language agents with reinforcement learning for strategic play in the werewolf game. In: Forty-first international conference on machine learning
Yang Y, Wang J (2020) An overview of multi-agent reinforcement learning from game theoretical perspective. arXiv preprint arXiv:201100583
Yang Y, Hao J, Chen G et al (2020a) Q-value path decomposition for deep multiagent reinforcement learning. In: International conference on machine learning, PMLR, pp 10706–10715
Yang Y, Hao J, Liao B et al (2020b) Qatten: a general framework for cooperative multiagent reinforcement learning. arXiv preprint arXiv:200203939
Yao M, Feng X, Yin Q (2023) More like real world game challenge for partially observable multi-agent cooperation. arXiv preprint arXiv:230508394
Ye D, Chen G, Zhang W et al (2020a) Towards playing full Moba games with deep reinforcement learning. Adv Neural Inf Process Syst 33:621–632
Ye D, Liu Z, Sun M et al (2020b) Mastering complex control in moba games with deep reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence, pp 6672–6679
Yin QY, Yang J, Huang KQ et al (2023) Ai in human-computer gaming: techniques, challenges and opportunities. Mach Intell Res 20(3):299–317
Yu C, Velu A, Vinitsky E et al (2022) The surprising effectiveness of Ppo in cooperative multi-agent games. Adv Neural Inf Process Syst 35:24611–24624
Yuan L, Wang J, Zhang F et al (2022) Multi-agent incentive communication via decentralized teammate modeling. In: Proceedings of the AAAI conference on artificial intelligence, pp 9466–9474
Yun WJ, Lim B, Jung S et al (2021) Attention-based reinforcement learning for real-time uav semantic communication. In: 2021 17th International symposium on wireless communication systems (ISWCS), IEEE, pp 1–6
Zang Y, He J, Li K et al (2023) Sequential cooperative multi-agent reinforcement learning. In: Proceedings of the 2023 international conference on autonomous agents and multiagent systems. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, AAMAS '23, p 485–493
Zha D, Xie J, Ma W et al (2021) Douzero: mastering doudizhu with self-play deep reinforcement learning. In: international conference on machine learning, PMLR, pp 12333–12344
Zhang Z (2024) Advancing sample efficiency and explainability in multi-agent reinforcement learning. In: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, pp 2791–2793
Zhang Y, Zavlanos MM (2023) Cooperative multiagent reinforcement learning with partial observations. IEEE Trans Autom Control 69(2):968–981
Zhang SQ, Zhang Q, Lin J (2019) Efficient communication in multi-agent reinforcement learning via variance based control. Adv Neural Inf Process Syst 32
Zhang H, Chen W, Huang Z et al (2020a) Bi-level actor-critic for multi-agent coordination. In: Proceedings of the AAAI conference on artificial intelligence, pp 7325–7332
Zhang SQ, Zhang Q, Lin J (2020b) Succinct and robust multi-agent communication with temporal message control. Adv Neural Inf Process Syst 33:17271–17282
Zhang K, Yang Z, Ba ̧sar T (2021a) Multi-agent reinforcement learning: a selective overview of theories and algorithms. Handbook of reinforcement learning and control pp 321–384
Zhang K, Yang Z, Liu H et al (2021b) Finite-sample analysis for decentralized batch multiagent reinforcement learning with networked agents. IEEE Trans Autom Control 66(12):5925–5940
Zhang R, Zong Q, Zhang X et al (2022) Game of drones: Multi-uav pursuit-evasion game with online motion planning by deep reinforcement learning. IEEE Trans Neural Networks Learn Syst 34(10):7900–7909
Zhang B, Li L, Xu Z et al (2023a) Inducing stackelberg equilibrium through spatio-temporal sequential decision-making in multi-agent reinforcement learning. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI '23. https://doi.org/10.24963/ijcai.2023/40
Zhang K, Kakade SM, Basar T et al (2023b) Model-based multi-agent Rl in zero-summarkov games with near-optimal sample complexity. J Mach Learn Res 24(175):1–53

Zhao Y, Borovikov I, de Mesentier Silva F et al (2020) Winning is not everything: enhancing game development with intelligent agents. IEEE Trans Games 12(2):199–212
Zhao E, Yan R, Li J et al (2022a) Alphaholdem: high-performance artificial intelligence for heads-up no-limit poker via end-to-end reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence, pp 4689–469
Zhao J, Hu X, Yang M et al (2022b) Ctds: centralized teacher with decentralized student for multiagent reinforcement learning. IEEE Trans Games 16(1):140–150
Zhao Y, Zhao J, Hu X et al (2022c) Douzero+: Improving doudizhu ai by opponent modeling and coach-guided learning. In: 2022 IEEE conference on games (CoG), IEEE, pp 127–134
Zhao Y, Zhao J, Hu X et al (2023) Full douzero+: improving Doudizhu Ai by opponent modeling, coach-guided training and bidding learning. IEEE Trans Games 16(3):518–529
Zheng L, Yang J, Cai H et al (2018) Magent: A many-agent reinforcement learning platform for artificial collective intelligence. In: Proceedings of the AAAI conference on artificial intelligence
Zhou M, Liu Z, Sui P et al (2020) Learning implicit credit assignment for cooperative multi-agent reinforcement learning. Adv Neural Inf Process Syst 33:11853–11864
Zhou H, Lan T, Aggarwal V (2022) Pac: assisted value factorization with counterfactual predictions in multi-agent reinforcement learning. Adv Neural Inf Process Syst 35:15757–15769
Zhou Z, Liu G, Tang Y (2023b) Multi-agent reinforcement learning: methods, applications, visionary prospects, and challenges. arXiv preprint arXiv:230510091
Zhu C, Dastani M, Wang S (2022) A survey of multi-agent reinforcement learning with communication. arXiv preprint arXiv:220308975