arXiv:2106.15691v1 [cs.LG] 29 Jun 2021

Multiagent Deep Reinforcement Learning: Challenges and Directions Towards Human-Like Approaches

Annie Wong · Thomas Bäck · Anna V. Kononova · Aske Plaat

July 1, 2021

Abstract This paper surveys the field of multiagent deep reinforcement learning. The combination of deep neural networks with reinforcement learning has gained increased traction in recent years and is slowly shifting the focus from single-agent to multiagent environments. Dealing with multiple agents is inherently more complex as (a) the future rewards depend on multiple players' joint actions and (b) the computational complexity of functions increases. We present the most common multiagent problem representations and their main challenges, and identify five research areas that address one or more of these challenges: centralised training and decentralised execution, opponent modelling, communication, efficient coordination, and reward shaping. We find that many computational studies rely on unrealistic assumptions or are not generalisable to other settings; they struggle to overcome the curse of dimensionality or nonstationarity. Approaches from psychology and sociology capture promising relevant behaviours such as communication and coordination. We suggest that, for multiagent reinforcement learning to be successful, future research addresses these challenges with an interdisciplinary approach to open up new possibilities for more human-oriented solutions in multiagent reinforcement learning.

Keywords Reinforcement learning · Deep learning · Multiagent systems · Evolutionary Algorithms · Psychology · Survey

1 Introduction

Reinforcement learning (RL) is a machine-learning method in which an autonomous agent maximises its long term reward through repeated interaction with its environment. The agent is not told what actions to take and must learn its optimal behaviour via trial-and-error. Since rewards may be delayed, an agent has to make

Leiden Institute of Advanced Computer Science
Leiden University, Leiden
The Netherlands

E-mail: a.s.w.wong@liacs.leidenuniv.nl

a trade-off between exploiting states with the current highest reward and exploring states that may potentially yield higher rewards (Bellman, 1957). As agents learn by a continuous process of receiving rewards for every action taken, RL can automate learning and decision-making without supervision or having complete models of the environment. Yet, one drawback of RL methods is that they suffer from the *curse of dimensionality* (Bellman, 1957): algorithms become less efficient as the dimensions of the state-action space increases (Sutton et al., 1998). In recent years the rise of deep reinforcement learning (DRL), a combination of RL and deep learning, has enabled artificial agents to reach human-level performance in a wide range of complex decision-making tasks that were not possible before. While prior RL applications required carefully handcrafted features based on human knowledge and experience (Sutton et al., 1998), deep neural networks can automatically find low-dimensional representations (features) of high-dimensional data (LeCun et al., 2015). This development has led to enormous growth in the application of RL to more complicated problems. First in single-agent settings such as playing Atari (Mnih et al., 2015), resource management (Mao et al., 2016) and indoor robot navigation (Zhu et al., 2017), and more recently in multiagent settings such as bidding optimization (Jin et al., 2018), traffic-light control (Chu et al., 2020), financial market trading (Bao and Liu, 2019) and strategy games like Go (Silver et al., 2016), Dota 2 (Berner et al., 2019) and Starcraft (Vinyals et al., 2019).

It is challenging to translate the successes of DRL in single-agent settings to a multiagent setting. Multiagent reinforcement learning (MARL) differs from single-agent systems foremost in that the environment's dynamics are determined by the joint actions of all agents in the environment, in addition to the uncertainty already inherent in the environment. As the environment becomes non-stationary, each agent faces the moving-target problem: the best policy changes as the other agents' policies change (Busoniu et al., 2008; Papoudakis et al., 2019). The violation of the stationarity assumption required in most single-agent RL algorithms poses a challenge in solving multiagent learning problems. The curse of dimensionality is worse in a multiagent setting as every additional agent exponentially increases the state-action space. At the same time, MARL introduces a new set of opportunities as agents may share knowledge and imitate or directly learn from other learning agents (Da Silva and Costa, 2019; Ilhan et al., 2019), which may accelerate the learning process and subsequently result in more efficient ways of arriving at a goal.

Multiagent deep reinforcement learning (MADRL) constitutes a young field that is rapidly expanding. Many real-world problems can be modeled as an MARL problem, and the emergence of DRL has enabled researchers to move from simple representations to more realistic and complex environments. This survey examines current research areas within MADRL, addresses critical challenges, and presents future research directions. Earlier surveys were driven by the theoretical difficulties in multiagent systems, including non-stationarity (Hernandez-Leal et al., 2019a; Papoudakis et al., 2019), partial observability, continuous state and action spaces (Nguyen et al., 2020). Others focus on how agents learn, such as transfer learning (Da Silva and Costa, 2019), on modelling other agents (Albrecht and Stone, 2018), or on a theoretical domain such as game theory (Yang and Wang, 2021) and evolutionary algorithms (Bloembergen et al., 2015). This paper complements a group of surveys that provide a general framework to classify the deep learning

algorithms used in recent multiagent DRL studies (Hernandez-Leal et al., 2019b; Gronauer and Diepold, 2021).

In contrast to prior work, we propose a taxonomy based on the challenges inherent in multiagent problem formalisations and their solutions. Modelling a multiagent problem differs from the single-agent setting due to the violation of the stationarity assumption and the difference in learning objectives. Hence, alternative problem formalisations and solutions have been introduced. While other taxonomies also start from multiagent problem representations (Yang and Wang, 2021; Zhang et al., 2021), their focus is on Markov games and extensive-form games—formalisations that originated from game theory. We extend these with other common frameworks used in recent MADRL research, such as the decentralised partially observable Markov game and the partially observable Markov game.

The remainder of this paper is organised as follows. In section 2 the preliminaries of single-agent RL are discussed. In section 3 we present the most common MADRL problem frameworks. The taxonomy is introduced in section 4. The discussion and recommendations for future research are given in section 5. We end with the conclusion in section 6.

2 Single-agent Reinforcement Learning

2.1 Markov Decision Process

Most RL problems can be framed as a Markov Decision Process (MDP) (Bellman, 1957): a model for sequential decision making under uncertainty that defines the interaction between a learning agent and its environment. Formally, it can be defined as a tuple $\langle S, A, P, R, \gamma \rangle$ where S is the set of states, A is the set of actions, P is the transition probability function, R is the reward function and $\gamma \in [0, 1]$ is the discount factor for future rewards. The learning agent interacts with the environment in discrete time-steps. At each time step t, the agent is in some state $s_t \in S$ and selects an action $a_t \in A$. At time-step t_{t+1} the agent receives a reward $R_{t+1} \in R$ and moves into a new state S_{t+1}. Specifically, it is given by the state transition function $P(s', r|s, a) = Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$ that defines the dynamics of the model. Each state in a MDP has the Markov property, which means that the future only depends on the current state and not on the history of earlier states and actions. MDPs further assume that the agent has full observability of the state and that the environment is stationary: the transition probabilities and rewards remain constant over time.

A setting where the agent does not have full observability of the state is called a partially observable Markov decision process (POMDP) (Åström, 1965). In addition to a state set S, transition probabilities P and reward function R, there is a finite set Ω of observations where $O(o|a, s')$ is the observation function O that denotes the probability that the agent observed o given that action a was taken which led to state s'.

A policy π is a mapping from states to probabilities of selecting each action and can be deterministic or stochastic. The goal of the agent is to learn a policy that maximizes its performance and is typically defined as the expected return, i.e. the

expected discounted sum of rewards $\mathbb{E}_\tau\Big[\sum_{t=0}^{T}\gamma^t r_t\Big]$ in a trajectory τ. The discount factor $\gamma \in [0,1]$ describes how rewards are valued. A γ closer to 0 means that the agent places more value on immediate rewards, while a γ closer to 1 indicates that the agent favours future rewards. A policy that maximizes the function above is called an optimal policy and is denoted as π^*.

The majority of MDP solving algorithms can be divided into one of the three groups: value-based methods, policy-based methods and model-based methods. This distinction is based on the three primary functions to learn in RL (Graesser and Keng, 2019). Hybrid forms of the three primary functions also exist. We present a brief overview of each of the three classes.

2.2 Value-based Methods

Value-based methods learn the value function and derive the optimal policy from the optimal value function. There are two kinds of value functions. The state-function describes how good it is to be in a state. It is the expected return from being in state s and then following policy π and is denoted as:

$$v_\pi(s) = \mathbb{E}_{s_0=s,\tau\sim\pi}\Big[\sum_{t=0}^{T}\gamma^t r_t\Big] \tag{1}$$

The action-value function describes how good it is to perform action a in state s and is denoted as:

$$q_\pi(s,a) = \mathbb{E}_{s_0=s,a_0=a,\tau\sim\pi}\Big[\sum_{t=0}^{T}\gamma^t r_t\Big] \tag{2}$$

The optimal policy π^* maximizes the state-value function such that $v_{\pi_*}(s) > v_{\pi(s)}$ for all $s \in S$ and all policies π. If we have the optimal state-value function, the optimal policy can be extracted by choosing the action that gives the maximum action-value for state s. This relationship is given by $\pi^* = \max_\pi v_\pi(s) = \max_\pi q_\pi(s,a)$.

Q-learning is a value-based method that has become increasingly popular due to remarkable results combined with deep learning. It is a form of temporal difference (TD) learning, which uses bootstrapping to approximate the value function. It is suitable for non-episodic tasks since it allows agents to learn from incomplete trajectories. The basic version of Q-learning keeps a lookup table for all state-action pairs, which is problematic in high-dimensional environments as it is infeasible for the agent to visit all states.

Deep Q-networks (DQN) (Mnih et al., 2015) is a parameterized approach that can be applied to more complex environments. It mitigates the instabilities of parameterized Q-learning using experience replay: a random sample of previous actions to break the correlations in the sequence of observations. Additionally, a separate target network is used towards which the Q-network is updated. The weights of the target network are frozen and periodically updated to limit the moving target problem.

Value-based methods are also called indirect methods as they do not optimize the policy function directly. Recent developments in RL research show a preference for policy-based strategies, even though value-based methods can capture the underlying structure of the environment (Arulkumaran et al., 2017).

2.3 Policy-based and Combined Methods

In contrast to value-based methods, policy-based methods search directly for the optimal policy and the output is represented as a probability distribution over actions. The optimal policy is found by optimizing a θ-parameterized policy with respect to the objective via gradient ascent. The policy network weights (i.e. the sum of rewards) are updated iteratively so that state-action pairs that result in higher returns are more likely to be selected. The objective is the expected return over all complete trajectories and is defined as follows:

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\Big[\sum_{t=0}^{T}\gamma^t r_t\Big] \tag{3}$$

Gradient ascent is then performed with respect to the policy parameters θ to improve on the objective: $\theta \leftarrow \theta + \alpha\nabla_\theta J(\pi_\theta)$ where a is the learning rate and $\nabla_\theta J(\pi_\theta)$ is the policy gradient that follows from the policy gradient theorem (Sutton et al., 1998):

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau\sim\pi\theta}\big[\sum_{t=0}^{T} R_t(\tau)\nabla_\theta \log \pi_\theta(a_t|s_t)\big] \tag{4}$$

The objective gradient can be computed as the expected sum of the gradients of the log probabilities of the actions and is multiplied by the corresponding return and forms the theoretical foundation for all policy gradient methods. REINFORCE (Williams, 1992) was one of the first and simplest policy gradient implementations which uses Monte Carlo sampling to estimate the policy gradient. The agent collects a trajectory τ of one full episode using its current policy to update the policy parameters.

$$\nabla_\theta J(\pi_\theta) \approx \sum_{t=0}^{T}\big(R_t(\tau)\big)\nabla_\theta \log \pi_\theta(a_t|s_t) \tag{5}$$

Policy gradient methods have several advantages. They perform better in continuous and stochastic environments, learn specific probabilities for each action, and they learn the appropriate level of exploration (Sutton and Barto, 2018). The main limitation of policy gradient methods is the large variance in the gradient estimators (Greensmith et al., 2004) due to sparse rewards and the fact that only a finite set of states and actions are tried, rather than all. In addition, policy gradient methods are not sample-efficient since new estimates of the gradients are learned independently from past estimates (Konda and Tsitsiklis, 2003; Peters and Schaal, 2008).

To address these limitations, Actor-Critic methods (Konda and Tsitsiklis, 2003; Grondman et al., 2012; Bahdanau et al., 2017) combine policy-based and value-based methods to get the best of both: Actor-Critic methods preserve the desirable convergence properties while reducing oscillation during learning. Actor-Critic methods consist of an actor, which learns a parameterized policy and a critic, which learns a value function (parameterized separately) to evaluate the state-action pair. The critic approximates and updates the value function parameters w for either the state-value $V(s;w)$ or the action-value $Q(a|s;w)$, and the actor updates the policy parameters θ for $\pi_\theta(a|s)$ in the direction suggested by the critic. While REINFORCE uses a simple sample return $R_t(\tau)$ as the reinforcing signal for computing the policy gradient, a new value, the advantage A, is commonly used in actor-critic methods (Sutton et al., 1998):

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_t\Big[A_t^\pi \nabla_\theta \log \pi_\theta(a_t|s_t)\Big] \tag{6}$$

where the advantage is the difference between the action-value and state-value $A^\pi = Q^\pi(s_t, a_t) - V^\pi(s_t)$. In contrast to Q-values that are absolute, the advantage is a relative measure that shows how much better the action is compared to the average state-value. Popular actor-critic methods include Advantage Actor-Critic (A2C) (Sutton et al., 1998) and Asynchronous Advantage Actor-Critic (A3C) (Mnih et al., 2016).

2.4 Model-based Methods

Model-based approaches learn a model of the environment in finding the optimal policy. This model captures the transition and reward function of the environment and can be given to or learned by the agent. The agent can then use planning, the construction of trajectories or experiences using the model (Hamrick et al., 2021), to find the optimal policy. While model-free methods focus on learning, where the agent improves a policy or value function from direct experiences generated by the environment, model-based methods focus on planning. The latter is therefore also known as indirect reinforcement learning (Sutton and Barto, 2018).

The transition function is a mapping $P(s_t, a_t) \rightarrow s_{t+1}$. It can be a forward function $(s_t, a_t) \rightarrow s_{t+1}$ which predicts the next state given the current state and action, a backward function $(s_t, a_t) \rightarrow s_{t-1}$ which predicts the preceding state of the current state, or an inverse function $(s_t, s_{t+1}) \rightarrow a_t$ which predicts the action to take to get from one state to another (Moerland et al., 2021). The forward function has attracted the most scientific interest in model-based learning. In deterministic settings the function $P(s, a)$ predicts the next state, while in stochastic settings the dynamics are represented by a probability distribution over the future state (Polydoros and Nalpantidis, 2017).

The environment model can either be given or learned. Games such as chess and Go belong to the first category. When there is no given model, the agent must learn it. To understand the dynamics of the environment, the agent repeatedly interacts with the environment according to a base policy $\pi_0(a_t|s_t)$. The experiences are stored in $\mathcal{D} = \{(s_t, a_t, s_{t+1})_i\}$, which is then used to learn the dynamics model $P(s, a)$ by minimizing $\sum_i ||P(s_i, a_i) - s_{t+1_i})||^2$. Given the current state s and action a, the next state s_{t+1} is then given by $s_{t+1} = P(s_t, a_t)$. Planning is then

performed through $P(s, a)$ (Levine, 2017; Chua et al., 2018). Planning methods generally compute value functions, via updates or backup operations to simulated experiences, to find the optimal policy (Sutton and Barto, 2018).

Examples of model-based algorithms include AlphaZero (Silver et al., 2017) and MuZero (Schrittwieser et al., 2020) that achieved state-of-the-art performance in Atari, Go, chess and Shogi. For a recent overview of deep reinforcement learning in model-based games, see (Plaat, 2020).

The main advantage of model-based approaches is better sample efficiency. Agents may use the model to simulate experiences to have fewer actual interactions with the environment, which results in faster convergence. However, it is difficult to accurately represent the model, especially in real-world scenarios where the environment is stochastic and the transition dynamics are not available. This problem becomes worse in large state-space environments. In addition, when bias and inaccuracies are present in the model, errors may accumulate for each step (Graesser and Keng, 2019).

3 Multiagent Problem Representations

In multiagent RL, a set of autonomous agents interact within the environment to learn how to achieve their objectives. While MDPs have proven helpful to model optimal decision-making in single-agent stochastic environments, multiagent environments require a different representation. The state dynamics and expected rewards change upon all agents' joint action, violating the core stationarity assumption of an MDP.

MDPs can be fully or partly visible to the agent. In a multiagent setting, the problem representation is also dependent on the nature of the interaction between agents, which can be cooperative, competitive or mixed, and whether agents take actions sequentially or simultaneously. Figure 1 shows an overview of the most common theoretical frameworks used in the MADRL literature. When agents have full observability of the state, the problem is usually represented by a Markov game. A particular type is the team Markov game, where agents collaborate to maximise a common reward. If agents collaborate but execute actions decentrally, it is represented by a decentralised partially observable Markov decision process. The partially observable variant for the mixed and competitive setting is known as the partially observable Markov game. The extensive-form game representation is used when agents take turns sequentially instead of simultaneously. The following sections outline the theoretical frameworks pertinent to the MADRL literature, which are visually depicted in Figure 2.

3.1 Markov Games

Markov games (Littman, 1994), or Stochastic games (Shapley, 1953)[2], provide a theoretical framework to study multiple interacting agents in a fully observable environment and can be applied to cooperative, collaborative and mixed settings.

[1] Illustrations are created with BioRender.com

[2] The terms Markov game and stochastic game are used interchangeably in the literature. For consistency, we will continue with the term Markov game throughout the paper.

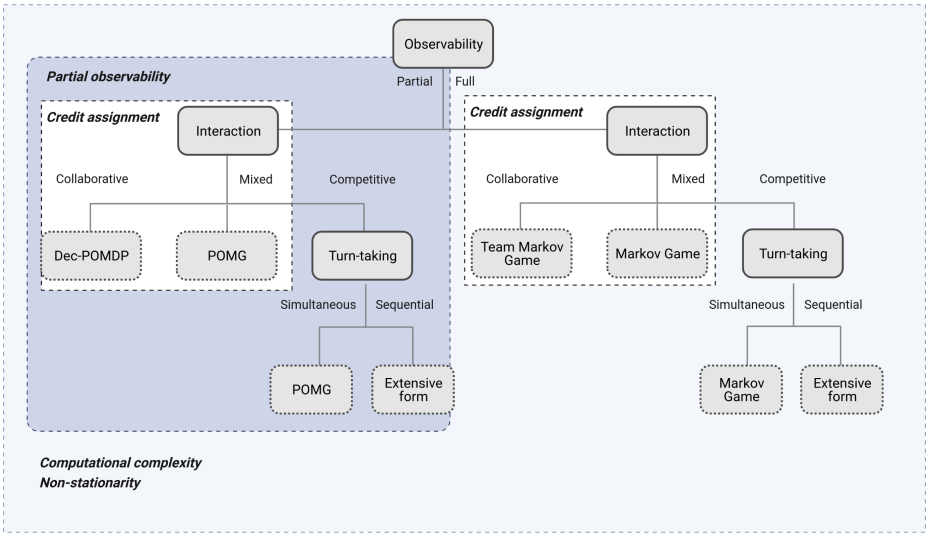

Fig. 1 Diagram of problem representations and their main challenges Multiagent problem representations can be categorised along a number of axes. First, whether the environment is fully or partially observable. Second, whether the nature of the interaction is collaborative, mixed or competitive. Third, whether turns are taken sequentially or simultaneously. Different problem representations come with different challenges. The four main challenges include computational complexity, non-stationarity, partial observability and credit assignment.[1]

A Markov game is a collection of normal-form games (or matrix games) that the agents play repeatedly. Each state of the game can be viewed as a matrix representation with the payoffs for each joint action determined by the matrices.

In its general form, a Markov game is a tuple $\langle I, S, A, R, T, \gamma \rangle$ where I is the set of N agents, S is a finite state space, $A = A_1 \times A_2 \times ... \times A_N$ is the joint action space of N agents, $R = (r_1, r_2, ..., r_N)$ where $R_i : S \times A \to \mathbb{R}$ is each agent's reward function, $T : S \times A \times S \to [0, 1]$ is the transition function and γ is the discount factor. In a team Markov game, agents work together to achieve a goal and share the rewards function $r_1 = r_2 = \ldots = R_N$. A competitive Markov game is represented by a zero-sum game: the gains for one party automatically result in equal losses for the other. A Markov game is a normal form game, which means that the game is represented in a tabular form, and all agents take their actions simultaneously.

One way to solve Markov games is to learn equilibria by optimising over an agent's reward function and ignore others in the environment (Tan, 1993; Littman, 1994). Another approach involves best response learners. Agents optimise their reward function while accounting for other agents' changing policies. If these algorithms converge during the play, then it must be an equilibrium (Bowling and Veloso, 2001, 2002). However, equilibrium concepts either assume infinite computational resources or have been applied to smaller grid-word environments, as they do not scale well with the number of agents.

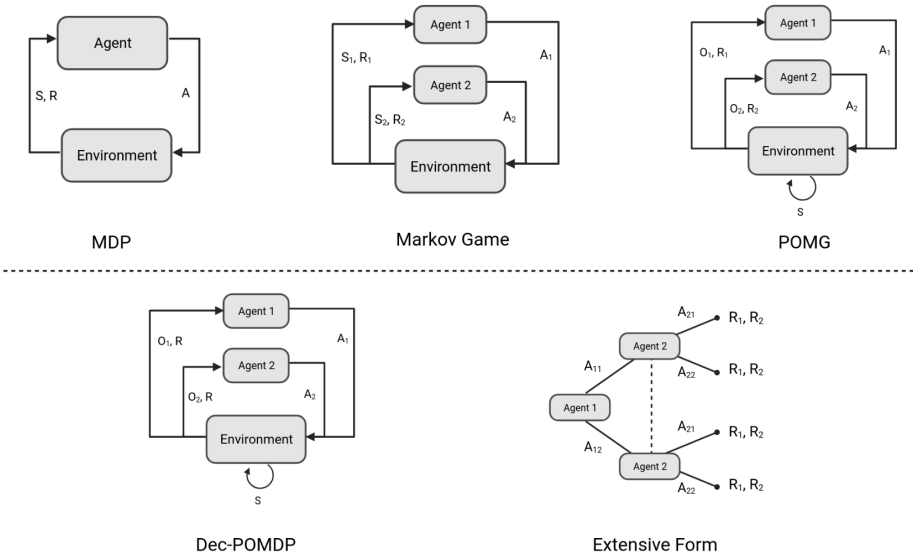

Fig. 2 Visual depiction of the main problem representations in multiagent reinforcement learning The MDP is the primary framework used in the single-agent setting. An agent is in some state S, performs action A, and receives a reward R from the environment. In partially observable environments, the agent cannot view the true state S and receives an observation O instead. For simplicity, all figures display the interaction between two agents $i = 1, 2$ but can be extended to more agents.

The majority of studies in MADRL focuses on Markov Games, such as Pong (Diallo et al., 2017), predator games (Zheng et al., 2018a) and the iterated prisoner's dilemma (Foerster et al., 2018b).

3.2 Extensive Form Games

When agents take turns sequentially, this is modelled as an extensive-form game (Kuhn and Tucker, 1953). An extensive-form game specifies the sequential interaction between agents in the form of a game tree. The game tree shows the order of the agents' moves and the possible actions at each point in time. Formally, an extensive game with finite and perfect-information is given by the tuple $\langle N, A, H, Z, \chi, \rho, \sigma, u \rangle$ where N is a set of agents, A is a single set of actions, H is a set of non-terminal choice nodes, Z is a set of terminal outcome nodes, $\chi : H \to 2^A$ is an action function, representing the set of possible actions at each node, $\rho : H \to N$ is the player function, which assigns at each choice node a player $i \in N$ who is to take an action at a given non-terminal node, $\sigma : H \times A \to H \cup Z$ is the successor function, that maps a choice node and an action to a new choice node or terminal node and u is a set of utility functions (Shoham and Leyton-Brown, 2008).

When agents have incomplete information or a partial view of the global state, this can be formalised as an imperfect information extensive-form game in which decision nodes are portioned into information sets. When the game

reaches the information set, the agent whose turn it is cannot distinguish between nodes within the information set and cannot tell which node in the tree has been reached. Formally, an imperfect information extensive-form game is a tuple $\langle N, A, H, Z, \chi, \rho, \sigma, u, I \rangle$ where $\langle N, A, H, Z, \chi, \rho, \sigma, u \rangle$ is a perfect information extensive-form game and $I = \{I_1, ..., I_n\}$ is the set of information partitions of all players.

A strategy maps each agent's information sets to a probability distribution over possible actions. The exploitability is a mean score over all positions against a worst-case adversary who uses at each turn a best-response. In a Nash equilibrium, the exploitability is equal to 0, and no agents have an incentive to change their strategies (Johanson et al., 2013). Studies try to solve extensive-form games by approximating a Nash equilibrium, predominantly in the poker domain (Heinrich et al., 2015; Moravčík et al., 2017; Heinrich and Silver, 2016; Brown and Sandholm, 2018, 2019) and board games (Silver et al., 2016, 2017).

3.3 Decentralized Partially Observable Markov Decision Process

In a decentralized partially observable Markov decision process (Dec-POMDP) all agents attempt to maximize the joint reward function, while having different individual objectives (Bernstein et al., 2002). A Dec-POMDP consists of a state space S, the transition probabilities $P(s'|s, a_1, ..., a_N)$ and expected rewards $R(s, a_1, ..., a_N)$. Ω_i is a finite set of observations for agent i, and $O(o_1, ..., o_N|a_1, ..., a_N, s')$ are observed by agents $1, ..., N$, respectively, given that each action tuple $\langle a_1, ..., a_N \rangle$ was taken and led to state s'. Each agent i has a set of actions A_o^i for each observation $o_i \in \Omega_i$. At every time step, each agent takes an action, receives a local observation that is correlated with the state and a joint immediate reward. A local policy is a mapping from local histories of observations to actions, and a joint policy is a tuple of local policies.

Dec-POMDPs are very hard to solve and are known to be NEXP-complete. These problems are not solvable with polynomial-time algorithms and searching directly for an optimal solution in the policy space is intractable (Bernstein et al., 2002). One approach is to transform the Dec-POMDP into a simpler model and solve it with planning algorithms (Amato and Oliehoek, 2015; Ye et al., 2017). For instance, using a centralised controller that receives all agents' private information converts the model into a POMDP and allowing communication that is free of costs and noise reduces it to a multiagent POMDP (MPOMDP) (Amato and Oliehoek, 2015; Gupta et al., 2017). Recent solutions also take advantage of the key assumption that planning can be centralised as long as execution is decentralised.

The Dec-POMDP has been used to represent riddles (Foerster et al., 2016), coordination of bipedal walkers (Gupta et al., 2017) and real-time strategy games such as Starcraft (Vinyals et al., 2019; Schroeder de Witt et al., 2019; Du et al., 2019), Dota 2 (Berner et al., 2019), and Capture the Flag (Jaderberg et al., 2019).

3.4 Partially Observable Markov Game

The partially observable Markov game (POMG) (Hansen et al., 2004), also known as the partially observable stochastic game (POSG), is the counterpart of the

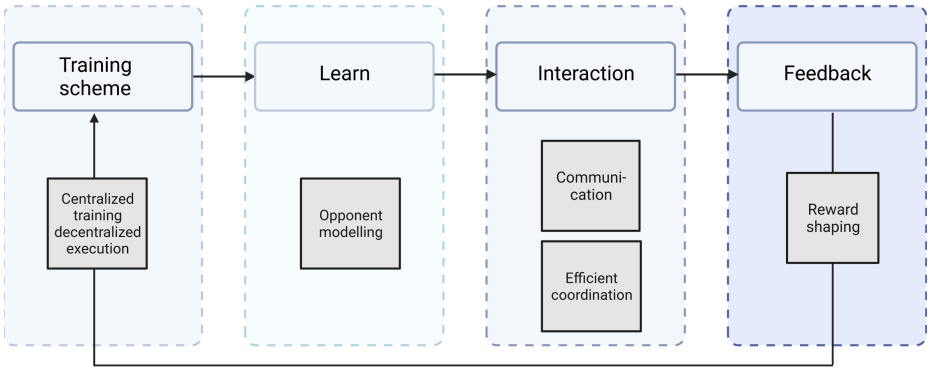

Fig. 3 Overview of taxonomy This figure shows how the paper is organized. We start with discussing the main training scheme in MADRL: centralized training and decentralized execution. We then move to how agents learn through opponent modelling, and interact with other agents via communication and coordination. Finally, we discuss how different reward shaping methods act as a feedback mechanism.

Dec-POMDP. Instead of a joint reward function, agents optimise their individual reward functions in a partially observable environment. The POMG implicitly models a distribution over other agents' belief states. Formally, a POMG is a tuple $\langle I, S, A, O, b^0, P, R \rangle$ where I is the set of N agents, S is the set of states, A_i is the action set of agent i and $A = A_1 \times A_2 \times ... \times A_N$ is the joint action set, O_i is a set of observations for agent i and $O = O_1 \times O_2 \times ... \times O_N$ is the joint observation set. The initial state of the game, also called the initial belief, is drawn from a probability distribution b^0 over the states. P is a set of state transitions and observation probabilities, where $P(s', o|s, a)$ is the probability of moving into state s' and joint observation o when taking joint action a in state s. $R_i : S \times A \to \mathbb{R}$ is the reward function for agent i where S refers to the joint state $(s_1, ..., s_N)$ and A refers to the joint actions $(a_1, ..., a_N)$. The model can be reduced to a POMDP when $|I| = 1$.

Dynamic programming algorithms have been developed for POMG (Hansen et al., 2004; Kumar and Zilberstein, 2009), in which agents maintain a belief over the actual state of the environment and other agents' policies. However, applying it to high-dimensional problems becomes intractable, and assumptions are often relaxed or applied to simpler problems. Complexities such as competing goals, non-stationarity and incomplete information make the problem even harder. Examples of POMG include autonomous driving (Palanisamy, 2020) and partially observable gridworld games (Moreno et al., 2021), few other works have yet studied POMG.

4 Taxonomy of Multiagent Deep Reinforcement Learning Algorithms

We will now introduce the taxonomy of this paper. We first discuss the four main challenges inherent in multiagent settings: (1) computational complexity, (2) non-stationarity, (3) partial observability and (4) credit assignment. We then provide

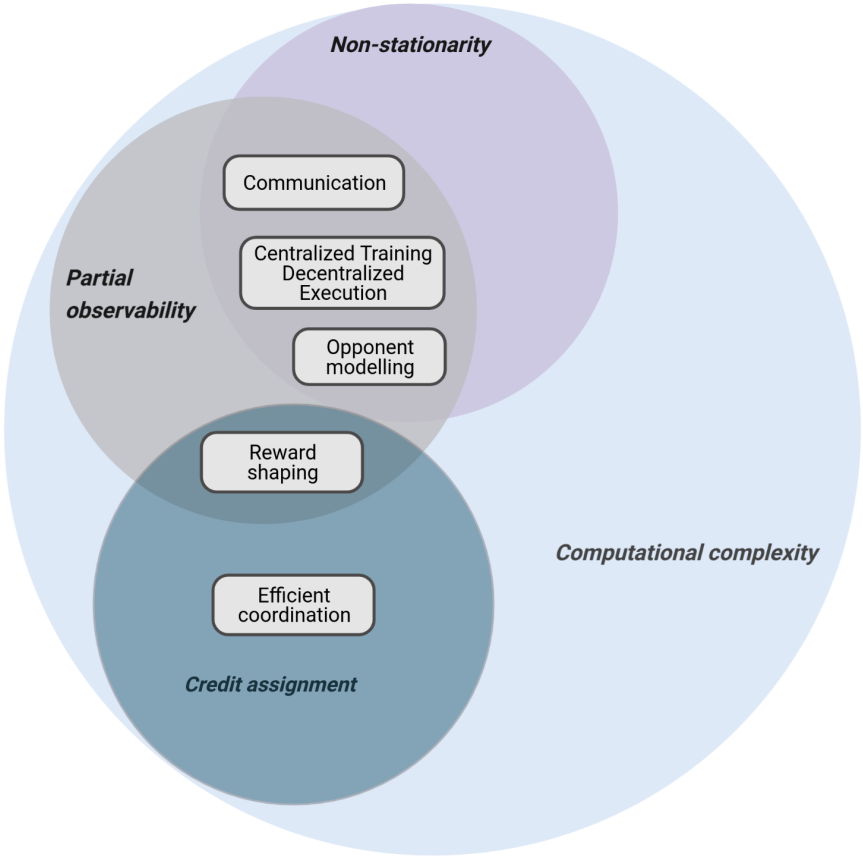

Fig. 4 Venn diagram of challenges and solutions The taxonomy of MADRL algorithms comprises five groups: centralised training and decentralised execution, opponent modelling, communication, efficient coordination and reward shaping. Approaches may tackle one or more challenges: non-stationarity, partial observability, credit assignment and computational complexity. This Venn diagram shows the relations between the surveyed groups of studies and the addressed challenges.

an overview of current deep learning approaches and discuss how these algorithms address these challenges. The surveyed studies cover the whole learning process of an agent: starting from the training scheme to how it learns and interacts with the environment and incorporates feedback, as shown in Figure 3. The reviewed algorithms have been categorised into one of the following groups: (1) centralised training and decentralised execution, (2) opponent modelling, (3) communication, (4) efficient coordination and (5) reward shaping. Figure 4 shows the relationship between the reviewed studies and the challenges that they address. Finally, Table 1 presents examples of studies along with their main challenges and solutions.

Table 1 Overview of studies along with the problem representation, main challenges, evaluation domains and solutions

Study	Evaluation domain	Problem representation	Main challenge(s)	Approach	Method
Moreno et al. 2021	Running with Scissors	POMG	Partial observability	Opponent modelling	Learning recursive belief models
Sukhbaatar, 2016	Traffic Junction	Dec-POMDP	Partial observability, non-stationarity	Communication	Communication using backpropagation
Sunehag et al., 2017	Switch	Dec-POMDP	Partial observability, non-stationarity	Centralized training and decentralized execution	Value-Decomposition Networks
Heinrich and Silver, 2016	Leduc Poker	Incomplete information extensive-form game	Partial observability, computational complexity	Opponent modelling	Neural Fictious Self-Play
Bowling et al., 2015	Heads-up limit hold'em Poker	Incomplete information extensive-form game	Partial observability, computational complexity	Opponent modelling	Self-play based on counterfactual regret minimization
Silver et al., 2018	Go	Complete information extensive-form game	Computational complexity	Opponent modelling	Self-play and Monte Carlo Tree Search
Nguyen et al., 2018	Matching taxi supply and demand	Dec-POMDP	Credit assignment, partial observability, non-stationarity	Reward shaping	Difference rewards: wonderful life utility and aristocratic utility
Leibo et al., 2017	Sequential Social Dilemma	Markov game	Credit assignment	Efficient coordination	Learn policy dynamics of DQN agents by altering variables
Vinyals et al., 2019	StarCraft	Dec-POMDP	Credit assignment, partial observability, nonstationarity, computational complexity	Opponent modelling	Population-based self-play
Jaderberg et al., 2019	Capture the flag				
Berner et al., 2019	Dota 2	Dec-POMDP	Credit assignment, partial observability, nonstationarity, computational complexity	Opponent modelling	OpenAI Five: self-play against itself and past selves

4.1 Challenges

Reinforcement learning in a multiagent environment comes with numerous challenges. Addressing these challenges is a prerequisite for the development of effective learning approaches. Despite promising results in the literature, computational complexity, non-stationarity, partial observability and credit assignment remain largely unsolved. We turn to each of these aspects next.

4.1.1 Computational Complexity

Training multiple learning agents is challenging due to the curse of dimensionality: the state-action space increases exponentially with the addition of every agent. Training a deep RL model for a single agent already requires substantial resources and this gets worse for multiple interacting agents. We face slow learning of new tasks and, in the worst-case, tasks even become infeasible to master. Hence, a significant number of studies focus on the efficient usage of resources and scalability of algorithms.

4.1.2 Non-stationarity

In a multiagent environment, all agents learn, interact and change the environment concurrently. Consequently, the state transitions and rewards are no longer stationary, and agents keep adapting to other agents' changing policies. This violates the Markov assumption, which is problematic since most RL algorithms assume a stationary environment to guarantee convergence. Recent works have addressed non-stationarity differently, focusing on various variables: such as the setting (cooperative, competitive or mixed), whether and how opponents are modelled, the training process of agents, the availability of opponent information, and whether the execution of actions is centralised or decentralised (Papoudakis et al., 2019). There is also a wide range of sophistication between algorithms; some algorithms simply ignore that the environment is non-stationary, assuming that other agents are part of the environment, while more complex methods involve opponent modelling with recursive reasoning (Hernandez-Leal et al., 2019b). One way to address non-stationarity is to learn as much as possible about the environment, for example, through opponent modelling and exchanging information between agents. For a thorough overview of how algorithms model and cope with non-stationarity, we refer to surveys on non-stationarity (Papoudakis et al., 2019; Hernandez-Leal et al., 2019b).

4.1.3 Partial Observabilty

In a partially observable environment, agents do not have access to the global state and have to make decisions based on local observations. This results in incomplete and asymmetric information across agents, which makes training difficult. Other agents' rewards and actions are not always visible, making it difficult to attribute a change in the environment to an agent's own action. Partial observability has been mainly studied in the setting where a group of agents maximise a team reward via a joint policy (e.g. in the Dec-POMDP setting). The two main approaches are the

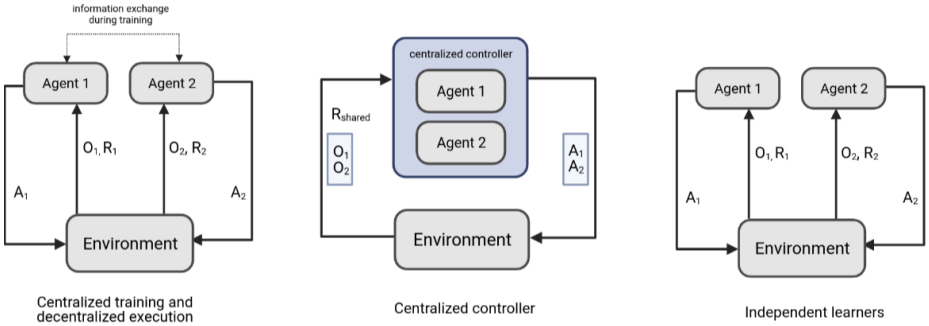

Fig. 5 Overview of training schemes The three main training schemes in multiagent settings are centralised training and decentralised execution, using a centralised controller and independent learning. The most popular approach is centralised training with decentralised execution, in which agents are allowed to share information during training, but actions are executed decentrally based on local observations. Using a centralised controller reduces the problem to a single-agent problem but is computationally infeasible. Finally, independent learners consider other agents as part of the environment but ignore the non-stationarity problem.

centralised training and decentralised execution paradigm and using communication to exchange information about the environment.

4.1.4 Credit Assignment

Two credit assignment problems are inherent in multiagent settings. The first problem is that an agent cannot always determine its individual contribution to the joint reward signal due to other concurrently acting agents in the same environment (Minsky, 1961). This makes learning a good policy more difficult as the agent cannot tell whether changes in the global reward was due to its own actions or others in the environment. An alternative to the global reward structure is to let agents learn based on a local reward: a reward based on the part of the environment that an agent can directly observe. However, while an agent may more easily increase his local reward, this approach encourages selfish behaviour that may lower overall group performance. Hence, reward shaping methods have been introduced to deal with the credit assignment problem.

The second problem refers to the construction of a reward function to promote effective collaborative behaviour. This is especially difficult when there are mixed incentives in an environment, such as social dilemmas. The lazy agent problem is also undesirable (Sunehag et al., 2017): when multiple agents are interacting at the same time, and one agent learns a good policy, the second agent can hold back to avoid affecting the performance of the first agent.

4.2 Centralised Training and Decentralised Execution

We will now turn to the approaches that have been developed to address these challenges.

A main challenge in MADRL is to design a multiagent training scheme that is efficient and that can deal with the non-stationarity and partial observability problem. The three most common training schemes are visually depicted in Figure 5. One of the most simple multiagent training schemes is to train multiple collaborating agents with a centralised controller, and to reduce it to a single-agent problem. All agents send their observations and policies to a central controller, and the central controller decides which action to take for each agent. This method mitigates the problem of partial observability when agents have incomplete information about the environment. However, using a centralised controller is computationally expensive in large environments. On the other extreme, all agents can learn an individual action-value function and view other agents as part of the environment (Tan, 1993). This method does not allow agents to coordinate with each other and ignores the non-stationarity problem.

An approach that combines both centralised and decentralised processing, is centralised training and decentralised execution (Kraemer and Banerjee, 2016). The main idea is that agents can access extra information during training, such as other agents' observations and rewards, gradients and parameters. Agents execute their policy decentrally based on local observations. Centralised training and decentralised execution mitigates non-stationarity and partial observability, as it stabilises the learning environment of agents, even when other agents' policies are changing. Centralised training and decentralised execution methods can be divided into value-based methods and policy-based methods.

Value-based methods focus on how to decouple centrally learned value functions and use them for decentralised execution. Value-function factorisation is one of the most popular methods in this category (Sunehag et al., 2017; Rashid et al., 2018; Son et al., 2019; Mahajan et al., 2020; Rashid et al., 2020; Yang et al., 2020). Value Decomposition Networks (VDN) (Sunehag et al., 2017) decompose the team value function into a sum of linear, individual value functions. The optimal policy arises by acting greedily with respect to the Q-value during execution. QMIX (Rashid et al., 2018) improves VDN's performance by treating the joint value function as a nonlinear combination of individual value functions and a monotonic constraint. However, this constraint limits the performance of collaborating agents that require significant coordination (Rashid et al., 2020). QTRAN (Son et al., 2019) employs a different factorisation method that can escape the monotonicity and additivity constraints. However, it relies on regularisations to maintain tractable computations, which may impede performance on complex multiagent settings (Mahajan et al., 2020). Numerous algorithms build further upon QMIX. For instance, Weighted QMIX extends QMIX to nonmonotonic environments by placing more weights on better joint actions (Rashid et al., 2020). Multi-Agent Variational Exploration (MAVEN) (Mahajan et al., 2020) addresses the inefficient exploration problem in QMIX by allowing committed exploration: to perform coordinated exploratory actions over extended time-steps via a latent space for hierarchical control.

Policy-based methods mainly focus on the Actor-Critic architecture. These studies use a centralised critic to train decentralised actors. Counterfactual Multiagent (COMA) (Foerster et al., 2018a) uses a centralised critic to approximate the Q-function, and decentralised actors to optimise policies. The centralised critic has access to the joint action and all available state information, while each agent's policy is only dependent on its historical action-observation sequence. Along the

same line, Multi-Agent Deep Deterministic Policy Gradient (MADDPG) extends the Actor-Critic algorithm so that the critic has access to extra information during training and the actor only has access to local information (Lowe et al., 2020). As opposed to COMA, which uses one centralised critic for all agents, MADDPG has a centralised critic for each agent to have different reward functions in competitive environments. MADDPG is also capable of learning continuous policies, whereas COMA focuses on discrete policies. Several studies build upon MADDPG. For instance, R-MADDPG (Wang et al., 2020) extends the MADDPG algorithm to the partially observable environment by having both a recurrent actor and critic that keep a history of previous observations, and M3DDPG (Li et al., 2019) incorporates minimax optimisation to learn robust policies against agents with changing strategies. Since these methods concatenate all the observations in the critic, the input dimension increases exponentially with each agent. Hence, several studies devise more efficient methods to deal with this problem. For instance, mean-field Actor-Critic (Yang et al., 2018) factorises the Q-function using only the interaction with the neighbouring agents based on mean-field theory (Stanley, 1971), and the idea of dropout[3] can be extended to MADDPG to handle the large input space (Kim et al., 2019).

This training scheme has been applied to solve complex strategy games such as StarCraft Micromanagement (Foerster et al., 2018a) and hide-and-seek (Baker et al., 2019).

4.3 Opponent Modelling

Opponent modelling refers to the construction of models of the beliefs, behaviours, and goals of other agents in the environment (Albrecht and Stone, 2018). These opponent models can be used by an agent to guide its decision-making. Opponent modelling algorithms generally take a sequence of interactions with the modelled opponent as input and predict action probabilities as output. After generating the opponent's model, an agent can derive its policy based on that model. This method helps an agent to discover the competitor's intentions. Opponent modelling mitigates the non-stationarity and partially observability problem as agents collect historical observations to learn about the environment (i.e. opponents), allowing agents to track and switch between policies. This method is especially beneficial in the adversarial setting when the opponent has opposing interests, and other approaches such as communication and centralised training that require the opponents' information are unlikely. For a comprehensive overview of opponent modelling, we refer to other work (Albrecht and Stone, 2018).

Early methods assumed fixed play of opponents. Neural Fictitious Self-Play (NFSP) extends the idea of fictitious play (Brown, 1951) with neural networks to approach a Nash equilibrium in imperfect information games such as Poker (Heinrich and Silver, 2016). The main idea is to keep track of the opponents' historical behaviours and to choose a best response to the opponents' average strategies.

While NFSP requires actual interaction with the opponent, other methods do not. For instance, counterfactual regret minimisation has achieved successes in

[3] Randomly dropping units in the neural network to avoid overfitting (Srivastava et al., 2014).

poker (Bowling et al., 2015). AlphaZero achieved remarkable results in Go, chess, and Shogi, using a neural network with self-play and Monte Carlo Tree Search (Silver et al., 2017). MuZero was able to achieve this without a given model. Instead of modelling the entire environment, it focused on the three core elements most relevant for planning: the value, policy and reward (Schrittwieser et al., 2020). Still, these studies assume that the opponent follows a stationary strategy.

Later approaches look at non-stationary environments in which an agent has to track, switch, and possibly predict behaviour. A number of studies achieved superhuman performance using self-play in real-time strategy games characterised by long time horizons, non-stationary environments, partially-observed states, and high dimensional state and action spaces. OpenAI Five employs a similar method to fictitious play in playing Dota 2, but the algorithm learns a distribution over opponents and uses the latest policy instead of the average policy (Berner et al., 2019). This infrastructure has also been used to solve hide-and-seek, but hide-and-seek agents can act independently as the training scheme is centralised training and decentralised execution (Baker et al., 2019). In Capture-the-Flag and StarCraft II, a population of agents is trained to introduce variation. Policies are made more robust by letting agents play with sampled opponents and teammates from this population in a league (Jaderberg et al., 2019; Vinyals et al., 2019).

Some studies assume that the opponent switches between a set of stationary policies over time (He et al., 2016; Everett and Roberts, 2018; Zheng et al., 2018b). These algorithms derive the optimal policy based on the learned opponent's model and identify when the opponent changes the behaviour, and the agent has to relearn a new policy. Over time, the agent has a library of inferred opponent strategies and associated best response policies. The two main challenges are designing a policy detection mechanism and learning a best-response policy. Some studies use a variant of Bayes' rule to learn opponent models and assign probabilities to the opponent's available actions. An agent starts with a prior belief that is continually updated during interaction to make it more accurate. Switching Agent Model (SAM) learns opponent models from observed state-action trajectories in combination with a Bayesian neural network (Everett and Roberts, 2018). A Deep Deterministic Policy Gradient algorithm (Lillicrap et al., 2019) is used to learn the best response. Distilled Policy Network-Bayesian Policy Reuse+ (DPN-BPR+) (Zheng et al., 2018b) extends the Bayesian Policy Reuse+ algorithm (BPR+) (Hernandez-Leal et al., 2016) with a neural network to detect the opponent's policy via both its behaviour and the reward signal. The latter uses policy distillation (Rusu et al., 2016) to learn and reuse policies efficiently. Others use a form of deep Q-learning (Mnih et al., 2013). Deep Reinforcement Opponent Network (DRON) (He et al., 2016) uses one network to learn the Q-values to derive an optimal policy and a second network to learn the opponent policy representation, in addition to expert networks that capture different types of opponent strategies. A drawback of DRON is that it relies on handcrafted opponent features. Previous methods assume that the opponent remains stationary within an episode. Deep Policy Inference Q-Network (DPIQN) and Deep Recurrent Policy Inference Q-Network (DRPIQN) (Hong et al., 2018) incorporate policy features as a hidden vector into the deep Q-network to adapt itself to unfamiliar opponents. DRPIQN uses a Long Short Term Memory (LSTM) layer so that agents can learn in partially observable environments.

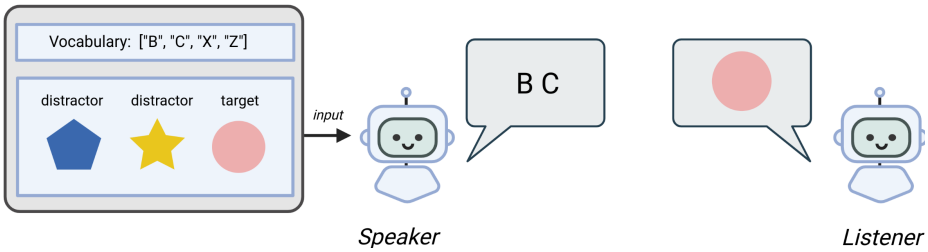

Fig. 6 Basic referential game In this basic referential game example, two agents have to develop a communication protocol so that the sender can translate the target into a message and the listener can understand which one is the target. The game works as follows. The sender receives as input three images. One is the target, and the other two are distractions. The speaker has to use the symbols in the vocabulary, which consists of the symbols "B", "C", "X" and "Z", to send a message to the listener. The listener sees the messages and has to guess the target message. If the target is correct, both agents receive a reward.

Previous approaches do not consider an intellectual and reasoning opponent. According to the theory of mind, people attribute mental states to others, such as beliefs, intents and emotions (Premack and Woodruff, 1978). These models help to analyse and infer others' behaviours and are essential in social interaction (Frith and Frith, 2005). Learning with Opponent-Learning Awareness (LOLA) (Foerster et al., 2018b) anticipates and shapes opponents' behaviour. Specifically, it includes a term that considers the impact of an agent's policy on the learning behaviour of opponents. One drawback is that LOLA assumes access to the opponent's parameters, which is unlikely in an adversarial setting. Others focus on recursive reasoning by learning models over the belief states of other players, a nesting of beliefs that can be represented in the form: "I believe that you believe that I believe" (Wen et al., 2019; Tian et al., 2021). The Probabilistic Recursive Reasoning (PR2) framework (Wen et al., 2019) first reflect on the opponent's perspective: what the opponents would do given that the opponents know the agent's current state and action. Given the potential actions of the opponent, the agent selects a best response. The recursive reasoning process can be viewed as a hierarchical process with k-levels of reasoning. At level $k = 0$, agents take random actions (Dai et al., 2020) or act based on historical interactions, the main assumption in traditional opponent modelling methods (Wen et al., 2019). At $k = 1$, an agent selects its best response to the agents acting at lower levels. Studies show that it pays off to reason about opponent's intelligence levels (Tian et al., 2021) and that reasoning at a higher level is beneficial as it leads to faster convergence (Dai et al., 2020) and better performance (Moreno et al., 2021).

4.4 Communication

Through communication, agents can pass information between each other, to reduce the complexity of finding good policies. For instance, agents exploring different parts of the environment can share observations to mitigate partial observability and share their intents to anticipate each others' actions in a non-stationary

setting. Communication can also be used for transfer learning so that more experienced agents can share their knowledge to accelerate the learning of new agents (Taylor and Stone, 2009). One of the fundamental questions in communication is how language emerges between agents with no predefined communication protocol, and, subsequently, how meaning and syntax evolve through interaction. A further question is how communication facilitates interaction in collaborative and competitive environments.

Several studies investigate how agents learn a successful communication protocol. A communication protocol should inform agents which concepts to communicate and how to translate these concepts into messages (Hausknecht and Stone, 2016). Many studies approach this problem as a referential game (Lazaridou et al., 2017; Havrylov and Titov, 2017). A referential game is a two-agent game in which a sender and a receiver must develop a communication protocol to refer to an object (Figure 6). In the basic version of this game, the sender sends two images and a message from a fixed vocabulary to the receiver. One of the images is the target, which the receiver has to identify based on the message. Both agents receive a reward when the classification is correct (Lazaridou et al., 2017). To succeed in this game, agents need to understand the image content and express the content through a common language. The language can be discrete, where messages are a single symbol (Lazaridou et al., 2017) or a sequence of symbols (Havrylov and Titov, 2017), or continuous, where messages are continuous vectors (Sukhbaatar et al., 2016).

Using DRL, end-to-end policies can be learned in which agents receive image pixels as input and a corresponding message as output. For example, two agents represented as simple feed-forward networks can learn a communication protocol to solve the basic referential game (Lazaridou et al., 2017). Language also emerges in more complicated versions of the game that require dialogue (Jorge et al., 2017; Das et al., 2017; Kottur et al., 2017) or negotiation (Cao et al., 2018) between agents. Agents trained with deep recurrent Q-networks (DRQN) (Jorge et al., 2017) and REINFORCE (Das et al., 2017; Kottur et al., 2017) are able to learn a communication protocol from scratch. Since communication is not always meaningful, it is important to develop metrics for emergent communication. An example is when an agent sends a message that has no actual impact on the environment. Agents with the capacity to communicate should exhibit positive signalling and positive listening (Lowe et al., 2019). The first is that messages correlate with observation or actions, and the second is that the update of beliefs or behaviour after receiving a message. Most studies focus solely on positive signalling metrics. However, positive signalling may occur without positive listening (Lowe et al., 2019), which indicates that there was no actual communication.

In contrast to earlier works that consider communication as the primary learning goal, other works consider communication an instrument to learn a specific task. The majority of these studies focus on coordination in collaborative environments and show that communication improves overall performance. Differentiable Interagent Learning (DIAL) (Foerster et al., 2016) uses centralised training and decentralised execution. Communication is continuous during training and discrete during the execution of the task. Continuous communication during training is particularly effective as it enables the exchange of gradients between agents, which improves performance. CommNet shows that the exchange of discrete symbols is less efficient than continuous communication, as the latter enables the use of back-

propagation to train agents efficiently (Sukhbaatar et al., 2016). While DIAL and CommNet base their approach on DQRN, later studies propose the Actor-Critic architecture, including Actor-Coordinator-Critic Net (ACCNet) (Mao et al., 2017), Bidirectionally Coordinated Network (BiCNet) (Peng et al., 2017) and MADDPG (Lowe et al., 2020). This architecture can solve more complex problems than previous approaches and works for continuous actions. In addition, when critics are individually learned (Jiang and Lu, 2018) instead of centrally computed (Iqbal and Sha, 2019), agents have different reward functions which is suitable for competitive settings.

Communication also allows peer-to-peer teaching. More experienced agents communicate their knowledge to learning agents, which accelerates learning of a new task (Da Silva et al., 2017; Omidshafiei et al., 2019; Ilhan et al., 2019; Amir et al., 2016). However, having agents send messages to all agents is costly and inefficient. Thus, an important question is how to filter the most important messages and to whom to send them. One approach is to limit communication bandwidth (Foerster et al., 2016; Kim et al., 2020) or use a communication budget (Ilhan et al., 2019; Omidshafiei et al., 2019). Others use metrics to identify relevant messages, such as attention mechanisms. In its simplest form, this is a vector of importance weights (Peng et al., 2018; Geng et al., 2019; Mao et al., 2020). An alternative is to keep confidence scores about states (Da Silva et al., 2017). Yet, communication comes at a cost and increased complexity. Negative transfer can also happen, for example, when the message contains inaccurate or noisy information, such that performance may become worse (Taylor and Stone, 2009). Therefore, it is essential to trade off benefits and costs or find a better way to filter valuable information.

4.5 Efficient Coordination

Another group of studies investigate agents' emergent behaviours and look at how cooperating agents can coordinate actions most efficiently. These studies are conducted in mixed environments with elements of both cooperation and competition.

A key question is how to design reward functions so that agents adapt to each others' actions, avoid conflicting behaviour and achieve efficient coordination. By engineering the reward function, competitive or cooperative behaviours can be stimulated (Tampuu et al., 2017). While early studies look at how agents can maximise external rewards, recent works assume that agents are intrinsically motivated.

The majority of studies look at multiagent behaviour in social dilemmas (Eccles et al., 2019; Lerer and Peysakhovich, 2018; Leibo et al., 2017; McKee et al., 2020; Jaques et al., 2019; Peysakhovich and Lerer, 2017). Earlier studies, mainly influenced by game theory, have looked at social dilemmas as a matrix game in which agents choose pure cooperation or pure defect. Recent studies generalise these social dilemmas to temporally and spatially extended Markov games, also known as a sequential social dilemma (Leibo et al., 2017). This setting is more realistic as people can adapt and change their strategy. One notable example is the repeated prisoner's dilemma. In each turn, each agent decides whether to cooperate or defect. When both agents cooperate, both agents get good rewards. Contrary, defection improves one agent's reward at the expense of the other agent. Thus, an

agent can decide to retaliate or trust the opponent, dependent on the actions in the previous round.

One of the first sequential social dilemma studies examined how policies change due to environmental factors or agent properties (Leibo et al., 2017). They found that agents learn more aggressive policies when resources are limited. In addition, manipulating the discount rate over the rewards, batch size and the number of hidden units in the network, affected emerging social behaviour. While this study took a descriptive approach to understand how behaviours change to different rules and conditions, others took a prescriptive approach in which agents learn to cooperate without being exploited (Lerer and Peysakhovich, 2018; Wang et al., 2018). The general approach comprises two steps: first, detect the level of cooperation of the opponent, and then mimic or reciprocate with a slightly higher-level cooperation policy to induce cooperation without getting exploited. This approach is based on the Tit-for-Tat principle (Axelrod and Hamilton, 1981): the strategy suggests cooperation in the first round and copies the opponent's behaviour afterwards.

Previous approaches assume that the only incentive for cooperation is external reward. However, there is a rapidly growing literature where cooperation occurs from social behaviour and intrinsic motivation (McKee et al., 2020; Jaques et al., 2019; Peysakhovich and Lerer, 2017; Hughes et al., 2018).

Psychology research has shown that people do not always seek to maximise utility (Dovidio, 1984). In addition, intrinsic reward may be a good alternative in sparse environments. Several attempts have been made to design these internal rewards. For instance, inequity aversion, which refers to the preference for fairness and resistance against inequitable outcomes (Fehr and Schmidt, 1999), has shown to improve coordination in social dilemmas (Hughes et al., 2018). The main idea is to punish agents that deviate too much from the average behaviour. Underperforming and overperforming agents are both undesirable, as the first may exhibit free-riding behaviour while the latter may be operating a defective policy. Another approach is to make agents care about the rewards of teammates (Peysakhovich and Lerer, 2017; Jaques et al., 2019).

Prosocial behaviour improves the convergence probabilities of policy gradient based agents, even if only one of the two players displays social behaviour (Peysakhovich and Lerer, 2017). In addition, rewarding actions that lead to a relatively more significant change in the other agent's behaviour may lead to increased cooperation (Jaques et al., 2019). Another study introduces heterogeneity in intrinsic motivation (McKee et al., 2020). Specifically, the study compares homogenous agents to heterogeneous agents with different degrees of social value orientation. The results show that homogenous altruistic agents earn relatively high rewards, yet it appears that they adopt a lazy agent approach and produce highly specialised agents. This problem is not evident in heterogeneous groups. Hence, it shows that the widely adopted joint return approach may be undesirable as it masks high levels of inequality amongst agents.

While studies show that shaping reward functions can lead to better coordination, it is very challenging to tune the trade-off between the intrinsic and external reward, and whether it gives rise to cooperative behaviour may depend on the actual task and environment.

Table 2 Overview of solutions to the credit assignment problem

Study	Implicit/ explicit	Approach	Algorithm
Foerster et al. 2018	Explicit	Difference rewards	COMA: uses a counterfactual baseline to marginalise out the action of an agent.
Yu et al., 2019	Explicit	Potential-based rewards	MA-AIRL: extends maximum entropy inverse reinforcement learning to Markov games. A potential-based function is used to deal with reward shaping ambiguity.
Devlin et al., 2014	Explicit	Difference rewards and potential-based rewards	DriP: uses potential based reward shaping to improve difference rewards.
Sunehag et al., 2017	Implicit	Value-based: deep Q-learning	VDN: decomposes the team value function into a sum of linear, individual value functions.
Zhou et al., 2020	Implicit	Policy-based: actor-critic	LICA: a centralized critic maps current state information into a set of weights, and in turn, mixes individual action vectors into the joint action value estimate.

4.6 Reward Shaping

The credit assignment problem refers to the situation when individual agents cannot view their contribution to the joint team reward due to a partially observable environment. Researchers have introduced implicit and explicit reward shaping methods to deal with this problem. An overview of the reviewed reward shaping methods is given in Table 2.

The general solution to this problem is reward shaping, with difference rewards and potential-based reward shaping as the two main classes. Difference rewards consider both the individual and the global reward (Foerster et al., 2018a; Proper and Tumer, 2012; Nguyen et al., 2018; Castellini et al., 2020) and help an agent understand its impact on the environment by removing the noise created by other acting agents. Specifically, it is defined as $D_i(z) = G(z) - G(z - z_i)$ where D_i is the difference reward of agent i, $G(z)$ is the global reward considering the joint state-action z, and $G(z - z_i)$ is a modified version of the state-action vector z in which agent i takes a default action, or more intuitively, the global reward without the contribution of agent i (Yliniemi and Tumer, 2014). COMA (Foerster et al., 2018a) takes inspiration from difference rewards and uses a counterfactual baseline that marginalises out the action of an individual agent, keeping the actions of other agents' constant.

Potential-based reward shaping has also received attention lately (Suay et al., 2016; Devlin et al., 2014). Formally, it is defined as $F(s, s') = \gamma\Phi(s') - \Phi(s)$ (Ng et al., 1999) where $\Phi(s)$ is a potential function which returns the potential for state s and γ is the discount factor. It is a method to incorporate additional information into the reward function to accelerate learning. This approach has been proven not to alter the set of Nash equilibria in a Markov game (Devlin and Kudenko, 2011), even when the potential function changes dynamically during learning (Devlin and Kudenko, 2012). Combining the two approaches allows agents

to converge significantly faster than using difference rewards alone (Devlin et al., 2014). However, these reward shaping methods require manual tuning for each environment which is not efficient.

Previous approaches evaluate an agent's action against a baseline to extract its individual effect and belong to the class of explicit credit assignment. In contrast, implicit methods do not work with baselines. Value-based methods decomposes the global value function into individual state-action values, also known as value mixing methods, such as VDN (Sunehag et al., 2017), QMIX (Rashid et al., 2018) and QTRAN (Son et al., 2019) to filter out agent's individual contribution. However, these methods may not handle continuous action spaces effectively. Policy-based algorithms include Learning Implicit Credit Assignment (LICA) (Zhou et al., 2020) and Decomposed Multi-Agent Deep Deterministic Policy Gradient (DE-MADDPG) (Sheikh and Bölöni, 2020). LICA extends the idea of value mixing to policy-based methods. Under the centralised training and decentralised execution framework, a centralised critic is represented by a hypernetwork that maps state information into a set of weights that mixes individual action values into the joint action value. DE-MADDPG extends previous deterministic policy gradient methods using a dual-critic framework. The global critic takes as input all agents' observations and actions and estimates the global reward. The local critic receives as input only the local observation and action of an agent and estimates the local reward. This framework achieves better and more stable performance than earlier deterministic policy gradient methods.

5 Discussion

We have surveyed a range of studies in MADRL. While integrating deep neural networks in reinforcement learning has dramatically improved agents' learning in more complex and larger environments, we wish to highlight some current limitations and future research directions in the field.

In the development from single-agent reinforcement learning to multiagent reinforcement learning, most earlier studies used a game-theoretic lens to study interactive decision-making, assuming perfectly rational agents who maximise their behaviour through a deliberate optimisation process. However, while game theory's strength lies in its generalizability and mathematical precision, experiments have shown that it is often a poor representation of actual human behaviour (Colman, 2003). Furthermore, the curse of dimensionality and non-stationarity problems place severe restrictions on the size of the problems that can be solved by these approaches.

Researchers must consider irrational and altruistic decision-making, especially if we wish to extend artificial intelligence (AI) to more realistic environments or design applications for human-AI interaction in larger and more complex problems. We have seen that prosocial agents can achieve better group outcomes (Peysakhovich and Lerer, 2017; Hughes et al., 2018). Yet, studies are still limited, and we encourage fellow researchers to deepen our understanding in this field.

We also want to bring attention to the design and assumptions in current research. Many studies assume homogeneous agents; from a practical viewpoint, this may accelerate learning since agents can share policies and parameters. Agents

thus only need to learn one policy and may better anticipate the behaviour of other agents. However, whether this also leads to better performance in the final task is an open question. For instance, a soccer team usually consist of a forward, midfielder, defender and goalkeeper. The team's success is partly determined by how well each of them fulfils these different roles. Thus, an interesting question is whether it pays off to let each agent learn its own policy and have heterogeneous teams. While homogeneous agents can still act differently due to different observations input, the observation space must be the same size. This assumption does not always hold. For instance, agents have different observation spaces in soccer as individuals occupy different positions in the field. Preliminary results show that despite making the learning slower at the beginning, heterogeneous teams perform better at the final task (Kurek and Jaśkowski, 2016). Another study provides formal proof for parameter sharing between heterogeneous agents (Terry et al., 2021), which may mitigate the slow start problem.

Studies may also rely on unrealistic assumptions. For instance, multiple studies require access to opponents information, such as trajectories or parameters, while their problem domain actually gives an incentive to hide information. Others assume fixed behaviours of agents or that agents can view the global state.

Another issue is the generalizability of studies. For example, many studies require hand-crafted features or rewards specific to the environment. In addition, a majority of the studies are evaluated in two-player games. As a result, a danger exists that the agent's policy overfits to the behaviour of the second agent (i.e. the lazy agent problem) and that it does not generalise to other settings. Future research should integrate more realistic assumptions and work on the generalizability of studies to settings with more players or different environments.

While MADRL has seen a significant improvement in the types and complexities of challenges it can address, several hurdles remain. For example, problems associated with large search spaces, partially observable environments, non-stationarity, sparse rewards and the exploration-exploitation trade-off remain challenging. These issues are partly due to computational constraints, such that assumptions are often relaxed. We want to point out two other research areas, namely evolutionary algorithms and psychology, that may help researchers address some of the open questions.

5.1 Evolutionary Algorithms

Evolutionary algorithms (EAs) are inspired by nature's creativity and simulate the process of organic evolution to solve optimisation problems. In simple terms, a randomly initialised population of individual solutions evolves toward better regions of the search space via selection, mutation and recombination operators. A fitness function evaluates the quality of the individuals and favours the reproduction of those with a higher fitness score, while mutation maintains diversity in the population (Bäck and Schwefel, 1993). An early study sheds light on how EAs deal with RL problems (Moriarty et al., 1999) and has been confirmed by recent studies (Bloembergen et al., 2015; Drugan, 2019; Arulkumaran et al., 2019; Lehman et al., 2018a,b; Conti et al., 2018; Such et al., 2018; Zhang et al., 2017). EAs offer a novel perspective to scaling RL multiagent systems as it is highly parallelisable, and there is no need for backpropagation (Such et al., 2018; Khadka et al., 2020).

EAs have been compared with popular value-based and policy-gradient algorithms such as DQN and A3C (Such et al., 2018). Novelty search (Such et al., 2018; Conti et al., 2018) is a promising area (Lehman and Stanley, 2008) since it encourages exploration on tasks with sparse rewards and deceptive local optima—problems that remain an issue with conventional reward-maximising methods. EAs have been shown to work well with non-stationarity and partial observability, as it continually uses and evolves a population of agents instead of a single agent (Moriarty et al., 1999; Liu et al., 2020). EAs can evolve agents with different policies (Gomes et al., 2017, 2014; Nitschke et al., 2012), such that heterogeneity can be introduced in team-based learning. Population-based training has proven powerful to achieve superhuman behaviour in Capture the Flag (Jaderberg et al., 2019), and StarCraft (Vinyals et al., 2019).

5.2 Psychology

Many of the key ideas in reinforcement learning, such as operant conditioning and trial-and-error, originated in psychology and cognitive science research (Sutton et al., 1998). Interestingly, recent MADRL studies started moving towards more human-like agents, showing that characteristics like reciprocity and intrinsic motivation do pay off.

We believe that psychology may provide more useful insights in dealing with current problems in MARL. For instance, bounded rationality (Simon, 1990) models describe how individuals make decisions under a finite amount of knowledge, time and attention. To deal with bounded rationality, people use heuristics, or mental shortcuts, to solve problems quickly and efficiently (Gigerenzer and Goldstein, 1996). While heuristics are already used in RL research to deal with large and complex problems, selecting suitable heuristics is still an open question (Nareyek, 2003).

Psychology has a long tradition of investigating heuristics and may offer new perspectives. In addition, heuristics aid in filtering relevant information in a complex world, which may benefit agents in partially observable environments or countering negative knowledge transfer (Marewski et al., 2010). However, intuitive judgement can also lead to biases and suboptimal decision-making (Gilovich et al., 2002). Humans are also capable of creative problem-solving, which is a prerequisite for innovation. Likewise, agents need to explore the environment to find more optimal solutions. A first approach of combining creativity with reinforcement learning shows that creativity offers the potential to explore promising solution spaces whereas traditional methods fail (Colin et al., 2016). We encourage researchers to build upon recent prosocial behaviour studies to deepen our understanding of efficient coordination.

6 Conclusion

The current survey has presented an overview of the challenges inherent in multi-agent representations. We have identified five different research areas in MADRL that aim to mitigate one or multiple of these challenges: (1) centralised training

and decentralised execution, (2) opponent modelling, (3) communication, (4) efficient coordination and (5) reward shaping. While early studies drew inspiration from game theory and were evaluated on grid-based games, the field is moving towards more sophisticated and human-like representations. Nevertheless, dealing with large problem spaces in non-stationary and partially observable settings remains an open issue.

Existing research has approached this problem mainly from traditional, computational, RL perspectives. While the combination of deep learning with value-based and policy-based methods have shown to mitigate the problem, they seem to be only part of the answer. We encourage researchers to take an interdisciplinary perspective on developing new solutions and benefit from the knowledge of sociology, psychology and evolutionary algorithms. Finally, we believe this integration leads to more realistic scenarios that humans encounter in practice, such that the findings may eventually be fruitful in real-world applications.

7 Declarations

7.1 Funding

No funding was received to assist with the preparation of this manuscript.

7.2 Conflicts of interests

The authors have no conflicts of interest to declare that are relevant to the content of this article.

7.3 Availability of data and material

Not applicable

7.4 Code availability

Not applicable

References

Albrecht SV, Stone P (2018) Autonomous agents modelling other agents: A comprehensive survey and open problems. Artificial Intelligence 258:66–95

Amato C, Oliehoek F (2015) Scalable planning and learning for multiagent pomdps. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 29, pp 1995–2002

Amir O, Kamar E, Kolobov A, Grosz B (2016) Interactive teaching strategies for agent training. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence 2016, URL https://www.microsoft.com/en-us/research/publication/interactive-teaching-strategies-agent-training/

Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA (2017) Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine 34(6):26–38

Arulkumaran K, Cully A, Togelius J (2019) Alphastar: An evolutionary computation perspective. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp 314–315

Åström KJ (1965) Optimal control of markov decision processes with incomplete state estimation. Journal of Mathematical Analysis and Applications 10:174–205

Axelrod R, Hamilton WD (1981) The evolution of cooperation. Science 211(4489):1390–1396

Bäck T, Schwefel HP (1993) An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation 1(1):1–23

Bahdanau D, Brakel P, Xu K, Goyal A, Lowe R, Pineau J, Courville A, Bengio Y (2017) An actor-critic algorithm for sequence prediction. arXiv preprint arXiv:160707086

Baker B, Kanitscheider I, Markov T, Wu Y, Powell G, McGrew B, Mordatch I (2019) Emergent tool use from multi-agent autocurricula. arXiv preprint arXiv:190907528

Bao W, Liu Xy (2019) Multi-agent deep reinforcement learning for liquidation strategy analysis. arXiv preprint arXiv:190611046

Bellman R (1957) A markovian decision process. Journal of Mathematics and Mechanics pp 679–684

Berner C, Brockman G, Chan B, Cheung V, Debiak P, Dennison C, Farhi D, Fischer Q, Hashme S, Hesse C, Józefowicz R, Gray S, Olsson C, Pachocki JW, Petrov M, de Oliveira Pinto HP, Raiman J, Salimans T, Schlatter J, Schneider J, Sidor S, Sutskever I, Tang J, Wolski F, Zhang S (2019) Dota 2 with large scale deep reinforcement learning. arXiv preprint arXiv:191206680

Bernstein DS, Givan R, Immerman N, Zilberstein S (2002) The complexity of decentralized control of markov decision processes. Mathematics of Operations Research 27(4):819–840

Bloembergen D, Tuyls K, Hennes D, Kaisers M (2015) Evolutionary dynamics of multi-agent learning: A survey. Journal of Artificial Intelligence Research 53:659–697

Bowling M, Veloso M (2001) Rational and convergent learning in stochastic games. In: International Joint Conference on Artificial Intelligence, Citeseer, vol 17, pp 1021–1026

Bowling M, Veloso M (2002) Multiagent learning using a variable learning rate. Artificial Intelligence 136(2):215–250

Bowling M, Burch N, Johanson M, Tammelin O (2015) Heads-up limit hold'em poker is solved. Science 347(6218):145–149

Brown GW (1951) Iterative solution of games by fictitious play. Activity Analysis of Production and Allocation 13(1):374–376

Brown N, Sandholm T (2018) Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. Science 359(6374):418–424

Brown N, Sandholm T (2019) Superhuman ai for multiplayer poker. Science 365(6456):885–890

Busoniu L, Babuska R, De Schutter B (2008) A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 38(2):156–172

Cao K, Lazaridou A, Lanctot M, Leibo JZ, Tuyls K, Clark S (2018) Emergent communication through negotiation. arXiv preprint arXiv:180403980

Castellini J, Devlin S, Oliehoek FA, Savani R (2020) Difference rewards policy gradients. arXiv preprint arXiv:201211258

Chu T, Wang J, Codecà L, Li Z (2020) Multi-agent deep reinforcement learning for large-scale traffic signal control. IEEE Transactions on Intelligent Transportation Systems 21(3):1086–1095

Chua K, Calandra R, McAllister R, Levine S (2018) Deep reinforcement learning in a handful of trials using probabilistic dynamics models. arXiv preprint arXiv:180512114

Colin TR, Belpaeme T, Cangelosi A, Hemion N (2016) Hierarchical reinforcement learning as creative problem solving. Robotics and Autonomous Systems 86:196–206

Colman AM (2003) Cooperation, psychological game theory, and limitations of rationality in social interaction. Behavioral and Brain Sciences 26:139–198

Conti E, Madhavan V, Such FP, Lehman J, Stanley KO, Clune J (2018) Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. arXiv preprint arXiv:171206560

Da Silva FL, Costa AHR (2019) A survey on transfer learning for multiagent reinforcement learning systems. Journal of Artificial Intelligence Research 64:645–703

Da Silva FL, Glatt R, Costa AHR (2017) Simultaneously learning and advising in multiagent reinforcement learning. In: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), pp 1100–1108

Dai Z, Chen Y, Low BKH, Jaillet P, Ho TH (2020) R2-b2: Recursive reasoning-based bayesian optimization for no-regret learning in games. In: Proceedings of the 37th International Conference on Machine Learning, PMLR, pp 2291–2301

Das A, Kottur S, Moura JM, Lee S, Batra D (2017) Learning cooperative visual dialog agents with deep reinforcement learning. In: Proceedings of the IEEE international Conference on Computer Vision, pp 2951–2960

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

Devlin S, Yliniemi L, Kudenko D, Tumer K (2014) Potential-based difference rewards for multiagent reinforcement learning. In: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems, pp 165–172

Devlin SM, Kudenko D (2012) Dynamic potential-based reward shaping. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems, IFAAMAS, pp 433–440

Diallo EAO, Sugiyama A, Sugawara T (2017) Learning to coordinate with deep reinforcement learning in doubles pong game. In: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, pp 14–19

Dovidio JF (1984) Helping behavior and altruism: An empirical and conceptual overview. Advances in Experimental Social Psychology 17:361–427
Drugan MM (2019) Reinforcement learning versus evolutionary computation: A survey on hybrid algorithms. Swarm and Evolutionary Computation 44:228–246
Du Y, Han L, Fang M, Liu J, Dai T, Tao D (2019) Liir: Learning individual intrinsic reward in multi-agent reinforcement learning. Advances in Neural Information Processing Systems 32:4403–4414
Eccles T, Hughes E, Kramár J, Wheelwright S, Leibo JZ (2019) Learning reciprocity in complex sequential social dilemmas. arXiv preprint arXiv:190308082
Everett R, Roberts S (2018) Learning against non-stationary agents with opponent modelling and deep reinforcement learning. In: 2018 AAAI spring symposium series
Fehr E, Schmidt KM (1999) A theory of fairness, competition, and cooperation. The Quarterly Journal of Economics 114(3):817–868
Foerster J, Assael IA, De Freitas N, Whiteson S (2016) Learning to communicate with deep multi-agent reinforcement learning. Advances in Neural Information Processing Systems 29:2137–2145
Foerster J, Farquhar G, Afouras T, Nardelli N, Whiteson S (2018a) Counterfactual multi-agent policy gradients. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 32
Foerster JN, Chen RY, Al-Shedivat M, Whiteson S, Abbeel P, Mordatch I (2018b) Learning with opponent-learning awareness. arXiv preprint arXiv:170904326
Frith C, Frith U (2005) Theory of mind. Current Biology 15(17):644–645
Geng M, Xu K, Li Y, Liu S, Ding B, Wang H (2019) Attention-based fault-tolerant approach for multi-agent reinforcement learning systems. arXiv preprint arXiv:191002240
Gigerenzer G, Goldstein DG (1996) Reasoning the fast and frugal way: models of bounded rationality. Psychological Review 103(4):650
Gilovich T, Griffin D, Kahneman D (2002) Heuristics and biases: The psychology of intuitive judgment. Cambridge University Press
Gomes J, Mariano P, Christensen AL (2014) Avoiding convergence in cooperative coevolution with novelty search. In: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems, pp 1149–1156
Gomes J, Mariano P, Christensen AL (2017) Dynamic team heterogeneity in cooperative coevolutionary algorithms. IEEE Transactions on Evolutionary Computation 22(6):934–948
Graesser L, Keng WL (2019) Foundations of deep reinforcement learning: theory and practice in Python. Addison-Wesley Professional
Greensmith E, Bartlett PL, Baxter J (2004) Variance reduction techniques for gradient estimates in reinforcement learning. Journal of Machine Learning Research 5(9)
Gronauer S, Diepold K (2021) Multi-agent deep reinforcement learning: a survey. Artificial Intelligence Review pp 1–49
Grondman I, Busoniu L, Lopes GA, Babuska R (2012) A survey of actor-critic reinforcement learning: Standard and natural policy gradients. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42(6):1291–1307
Gupta JK, Egorov M, Kochenderfer M (2017) Cooperative multi-agent control using deep reinforcement learning. In: International Conference on Autonomous

Agents and Multiagent Systems, Springer, pp 66–83
Hamrick JB, Friesen AL, Behbahani F, Guez A, Viola F, Witherspoon S, Anthony T, Buesing L, Veličković P, Weber T (2021) On the role of planning in model-based deep reinforcement learning. arXiv preprint arXiv:201104021
Hansen EA, Bernstein DS, Zilberstein S (2004) Dynamic programming for partially observable stochastic games. In: American Association for Artificial Intelligence, vol 4, pp 709–715
Hausknecht M, Stone P (2016) Grounded semantic networks for learning shared communication protocols. In: International Conference on Machine Learning (Workshop)
Havrylov S, Titov I (2017) Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. arXiv preprint arXiv:170511192
He H, Boyd-Graber J, Kwok K, Daumé III H (2016) Opponent modeling in deep reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 1804–1813
Heinrich J, Silver D (2016) Deep reinforcement learning from self-play in imperfect-information games. arXiv preprint arXiv:160301121
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-Leal P, Rosman B, Taylor ME, Sucar LE, Munoz de Cote E (2016) A bayesian approach for learning and tracking switching, non-stationary opponents. In: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, pp 1315–1316
Hernandez-Leal P, Kaisers M, Baarslag T, de Cote EM (2019a) A survey of learning in multiagent environments: Dealing with non-stationarity. arXiv preprint arXiv:170709183
Hernandez-Leal P, Kartal B, Taylor ME (2019b) A survey and critique of multiagent deep reinforcement learning. Autonomous Agents and Multi-Agent Systems 33(6):750–797
Hong ZW, Su SY, Shann TY, Chang YH, Lee CY (2018) A deep policy inference q-network for multi-agent systems. arXiv preprint arXiv:171207893
Hughes E, Leibo JZ, Phillips MG, Tuyls K, Duéñez-Guzmán EA, Castañeda AG, Dunning I, Zhu T, McKee KR, Koster R, et al. (2018) Inequity aversion improves cooperation in intertemporal social dilemmas. arXiv preprint arXiv:180308884
Ilhan E, Gow J, Perez-Liebana D (2019) Teaching on a budget in multi-agent deep reinforcement learning. In: 2019 IEEE Conference on Games (CoG), IEEE, pp 1–8
Iqbal S, Sha F (2019) Actor-attention-critic for multi-agent reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 2961–2970
Jaderberg M, Czarnecki WM, Dunning I, Marris L, Lever G, Castaneda AG, Beattie C, Rabinowitz NC, Morcos AS, Ruderman A, et al. (2019) Human-level performance in 3d multiplayer games with population-based reinforcement learning. Science 364(6443):859–865
Jaques N, Lazaridou A, Hughes E, Gulcehre C, Ortega P, Strouse D, Leibo JZ, De Freitas N (2019) Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 3040–3049
Jiang J, Lu Z (2018) Learning attentional communication for multi-agent cooperation. arXiv preprint arXiv:180507733

Jin J, Song C, Li H, Gai K, Wang J, Zhang W (2018) Real-time bidding with multi-agent reinforcement learning in display advertising. In: Cuzzocrea A, Allan J, Paton NW, Srivastava D, Agrawal R, Broder AZ, Zaki MJ, Candan KS, Labrinidis A, Schuster A, Wang H (eds) Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp 2193–2201

Johanson M, Burch N, Valenzano R, Bowling M (2013) Evaluating state-space abstractions in extensive-form games. In: Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems, pp 271–278

Jorge E, Kågebäck M, Johansson FD, Gustavsson E (2017) Learning to play guess who? and inventing a grounded language as a consequence. arXiv preprint arXiv:161103218

Khadka S, Majumdar S, Miret S, McAleer S, Tumer K (2020) Evolutionary reinforcement learning for sample-efficient multiagent coordination. arXiv preprint arXiv:190607315

Kim DK, Liu M, Omidshafiei S, Lopez-Cot S, Riemer M, Habibi G, Tesauro G, Mourad S, Campbell M, How JP (2020) Learning hierarchical teaching policies for cooperative agents. arXiv preprint arXiv:190303216

Kim W, Cho M, Sung Y (2019) Message-dropout: An efficient training method for multi-agent deep reinforcement learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 6079–6086, DOI https://doi.org/10.1609/aaai.v33i01.33016079

Konda VR, Tsitsiklis JN (2003) Actor-critic algorithms. Journal on Control and Optimization 42(4):1143–1166

Kottur S, Moura JM, Lee S, Batra D (2017) Natural language does not emerge 'naturally' in multi-agent dialog. arXiv preprint arXiv:170608502

Kraemer L, Banerjee B (2016) Multi-agent reinforcement learning as a rehearsal for decentralized planning. Neurocomputing 190:82–94

Kuhn HW, Tucker AW (1953) Contributions to the theory of games, vol 2. Princeton University Press

Kumar A, Zilberstein S (2009) Dynamic programming approximations for partially observable stochastic games. In: Proceedings of the Twenty-Second International FLAIRS Conference, pp 547–552

Kurek M, Jaśkowski W (2016) Heterogeneous team deep q-learning in low-dimensional multi-agent environments. In: 2016 IEEE Conference on Computational Intelligence and Games (CIG), IEEE, pp 1–8

Lazaridou A, Peysakhovich A, Baroni M (2017) Multi-agent cooperation and the emergence of (natural) language. arXiv preprint arXiv:161207182

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444

Lehman J, Stanley KO (2008) Exploiting open-endedness to solve problems through the search for novelty. In: Artificial Life XI, Citeseer, pp 329–336

Lehman J, Chen J, Clune J, Stanley KO (2018a) Es is more than just a traditional finite-difference approximator. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 450–457, DOI https://doi.org/10.1145/3205455.3205474

Lehman J, Chen J, Clune J, Stanley KO (2018b) Safe mutations for deep and recurrent neural networks through output gradients. arXiv preprint arXiv:171206563

Leibo JZ, Zambaldi V, Lanctot M, Marecki J, Graepel T (2017) Multi-agent reinforcement learning in sequential social dilemmas. arXiv preprint arXiv:170203037

Lerer A, Peysakhovich A (2018) Maintaining cooperation in complex social dilemmas using deep reinforcement learning. arXiv preprint arXiv:170701068

Levine S (2017) Berkeley CS 294-112, Lecture Notes: Model-Based Reinforcement Learning. URL: `http://rail.eecs.berkeley.edu/deeprlcourse-fa17/f17docs/lecture_9_model_based_rl.pdf`. Last visited on 2021/05/12

Li S, Wu Y, Cui X, Dong H, Fang F, Russell S (2019) Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 4213–4220

Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, Silver D, Wierstra D (2019) Continuous control with deep reinforcement learning. arXiv preprint arXiv:150902971

Littman ML (1994) Markov games as a framework for multi-agent reinforcement learning. In: 11th International Conference on Machine Learning, Elsevier, pp 157–163

Liu Z, Chen B, Zhou H, Koushik G, Hebert M, Zhao D (2020) Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments. arXiv preprint arXiv:200715724

Lowe R, Foerster J, Boureau YL, Pineau J, Dauphin Y (2019) On the pitfalls of measuring emergent communication. arXiv preprint arXiv:190305168

Lowe R, Wu Y, Tamar A, Harb J, Abbeel P, Mordatch I (2020) Multi-agent actor-critic for mixed cooperative-competitive environments. arXiv preprint arXiv:170602275

Mahajan A, Rashid T, Samvelyan M, Whiteson S (2020) Maven: Multi-agent variational exploration. arXiv preprint arXiv:191007483

Mao H, Alizadeh M, Menache I, Kandula S (2016) Resource management with deep reinforcement learning. In: Ford B, Snoeren AC, Zegura EW (eds) Proceedings of the 15th ACM Workshop on Hot Topics in Networks, ACM Press, pp 50–56, DOI https://doi.org/10.1145/3005745.3005750

Mao H, Gong Z, Ni Y, Xiao Z (2017) Accnet: Actor-coordinator-critic net for "learning-to-communicate" with deep multi-agent reinforcement learning. arXiv preprint arXiv:170603235

Mao H, Zhang Z, Xiao Z, Gong Z, Ni Y (2020) Learning multi-agent communication with double attentional deep reinforcement learning. Autonomous Agents and Multi-Agent Systems 34(1):1–34

Marewski JN, Gaissmaier W, Gigerenzer G (2010) Good judgments do not require complex cognition. Cognitive Processing 11(2):103–121

McKee KR, Gemp I, McWilliams B, Duéñez-Guzmán EA, Hughes E, Leibo JZ (2020) Social diversity and social preferences in mixed-motive reinforcement learning. arXiv preprint arXiv:200202325

Minsky M (1961) Steps toward artificial intelligence. Proceedings of the IRE 49(1):8–30, DOI 10.1109/JRPROC.1961.287775

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. arXiv preprint arXiv:13125602

Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, et al. (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533

Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning.

In: Balcan MF, Weinberger KQ (eds) Proceedings of The 33rd International Conference on Machine Learning, PMLR, pp 1928–1937
Moerland TM, Broekens J, Jonker CM (2021) Model-based reinforcement learning: A survey. arXiv preprint arXiv:200616712
Moravčík M, Schmid M, Burch N, Lisý V, Morrill D, Bard N, Davis T, Waugh K, Johanson M, Bowling M (2017) Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. Science 356(6337):508–513
Moreno P, Hughes E, McKee KR, Pires BA, Weber T (2021) Neural recursive belief states in multi-agent reinforcement learning. arXiv preprint arXiv:210202274
Moriarty DE, Schultz AC, Grefenstette JJ (1999) Evolutionary algorithms for reinforcement learning. Journal of Artificial Intelligence Research 11:241–276
Nareyek A (2003) Choosing search heuristics by non-stationary reinforcement learning. In: Metaheuristics: Computer Decision-Making, Springer, pp 523–544
Ng AY, Harada D, Russell S (1999) Policy invariance under reward transformations: Theory and application to reward shaping. In: ICML, vol 99, pp 278–287
Nguyen DT, Kumar A, Lau HC (2018) Credit assignment for collective multiagent rl with global rewards. In: Proceedings of the 31th Advances in Neural Information Processing Systems, MIT Press
Nguyen TT, Nguyen ND, Nahavandi S (2020) Deep reinforcement learning for multiagent systems: a review of challenges, solutions, and applications. IEEE Transactions on Cybernetics 50(9):3826–3839, DOI 10.1109/TCYB.2020.2977374
Nitschke GS, Eiben A, Schut MC (2012) Evolving team behaviors with specialization. Genetic Programming and Evolvable Machines 13(4):493–536
Omidshafiei S, Kim DK, Liu M, Tesauro G, Riemer M, Amato C, Campbell M, How JP (2019) Learning to teach in cooperative multiagent reinforcement learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 6128–6136
Palanisamy P (2020) Multi-agent connected autonomous driving using deep reinforcement learning. In: International Joint Conference on Neural Networks, IEEE, pp 1–7
Papoudakis G, Christianos F, Rahman A, Albrecht SV (2019) Dealing with non-stationarity in multi-agent deep reinforcement learning. arXiv preprint arXiv:190604737
Peng P, Wen Y, Yang Y, Yuan Q, Tang Z, Long H, Wang J (2017) Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. arXiv preprint arXiv:170310069
Peng Z, Zhang L, Luo T (2018) Learning to communicate via supervised attentional message processing. In: Proceedings of the 31st International Conference on Computer Animation and Social Agents, pp 11–16
Peters J, Schaal S (2008) Natural actor-critic. Neurocomputing 71(7-9):1180–1190
Peysakhovich A, Lerer A (2017) Prosocial learning agents solve generalized stag hunts better than selfish ones. arXiv preprint arXiv:170902865
Plaat A (2020) Learning to Play: Reinforcement Learning and Games. Springer Nature
Polydoros AS, Nalpantidis L (2017) Survey of model-based reinforcement learning: Applications on robotics. Journal of Intelligent & Robotic Systems 86(2):153–173
Premack D, Woodruff G (1978) Does the chimpanzee have a theory of mind? Behavioral and Brain Sciences 1(4):515–526

Proper S, Tumer K (2012) Modeling difference rewards for multiagent learning. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems), Conitzer, Winikoff, Padgham, pp 1397–1398

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

Rashid T, Farquhar G, Peng B, Whiteson S (2020) Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. arXiv preprint arXiv:200610800

Rusu AA, Colmenarejo SG, Gulcehre C, Desjardins G, Kirkpatrick J, Pascanu R, Mnih V, Kavukcuoglu K, Hadsell R (2016) Policy distillation. arXiv preprint arXiv:151106295

Schrittwieser J, Antonoglou I, Hubert T, Simonyan K, Sifre L, Schmitt S, Guez A, Lockhart E, Hassabis D, Graepel T, et al. (2020) Mastering atari, go, chess and shogi by planning with a learned model. Nature 588(7839):604–609

Shapley LS (1953) Stochastic games. Proceedings of the National Academy of Sciences 39(10):1095–1100

Sheikh HU, Bölöni L (2020) Multi-agent reinforcement learning for problems with combined individual and team reward. In: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, pp 1–8

Shoham Y, Leyton-Brown K (2008) Multiagent systems: Algorithmic, game-theoretic, and logical foundations. Cambridge University Press

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, 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, Antonoglou I, Huang A, Guez A, Hubert T, Baker L, Lai M, Bolton A, et al. (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359

Simon HA (1990) Bounded rationality. Springer

Son K, Kim D, Kang WJ, Hostallero DE, Yi Y (2019) Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 5887–5896

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15(1):1929–1958

Stanley HE (1971) Phase transitions and critical phenomena. Clarendon, Oxford

Suay HB, Brys T, Taylor ME, Chernova S (2016) Learning from demonstration for shaping through inverse reinforcement learning. In: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, pp 429–437

Such FP, Madhavan V, Conti E, Lehman J, Stanley KO, Clune J (2018) Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. arXiv preprint arXiv:171206567

Sukhbaatar S, Szlam A, Fergus R (2016) Learning multiagent communication with backpropagation. arXiv preprint arXiv:160507736

Sunehag P, Lever G, Gruslys A, Czarnecki WM, Zambaldi V, Jaderberg M, Lanctot M, Sonnerat N, Leibo JZ, Tuyls K, et al. (2017) Value-decomposition net-

works for cooperative multi-agent learning. arXiv preprint arXiv:170605296
Sutton RS, Barto AG (2018) Reinforcement learning: An introduction. MIT press
Sutton RS, Barto AG, et al. (1998) Introduction to reinforcement learning, vol 135. MIT press Cambridge
Tampuu A, Matiisen T, Kodelja D, Kuzovkin I, Korjus K, Aru J, Aru J, Vicente R (2017) Multiagent cooperation and competition with deep reinforcement learning. PloS one 12(4):1–15, DOI https://doi.org/10.1371/journal.pone.0172395
Tan M (1993) Multi-agent reinforcement learning: Independent vs. cooperative agents. In: Proceedings of the tenth International Conference on Machine Learning, pp 330–337
Taylor ME, Stone P (2009) Transfer learning for reinforcement learning domains: A survey. Journal of Machine Learning Research 10(7)
Terry JK, Grammel N, Hari A, Santos L, Black B (2021) Revisiting parameter sharing in multi-agent deep reinforcement learning. arXiv preprint arXiv:200513625
Tian R, Tomizuka M, Sun L (2021) Learning human rewards by inferring their latent intelligence levels in multi-agent games: A theory-of-mind approach with application to driving data. arXiv preprint arXiv:210304289
Vinyals O, Babuschkin I, Czarnecki WM, Mathieu M, Dudzik A, Chung J, Choi DH, Powell R, Ewalds T, Georgiev P, et al. (2019) Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature 575(7782):350–354
Wang RE, Everett M, How JP (2020) R-maddpg for partially observable environments and limited communication. arXiv preprint arXiv:200206684
Wang W, Hao J, Wang Y, Taylor M (2018) Towards cooperation in sequential prisoner's dilemmas: a deep multiagent reinforcement learning approach. arXiv preprint arXiv:180300162
Wen Y, Yang Y, Luo R, Wang J, Pan W (2019) Probabilistic recursive reasoning for multi-agent reinforcement learning. arXiv preprint arXiv:190109207
Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning 8(3-4):229–256
Schroeder de Witt C, Foerster J, Farquhar G, Torr P, Boehmer W, Whiteson S (2019) Multi-agent common knowledge reinforcement learning. Advances in Neural Information Processing Systems 32:9927–9939
Yang Y, Wang J (2021) An overview of multi-agent reinforcement learning from game theoretical perspective. arXiv preprint arXiv:201100583
Yang Y, Luo R, Li M, Zhou M, Zhang W, Wang J (2018) Mean field multi-agent reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 5571–5580
Yang Y, Hao J, Chen G, Tang H, Chen Y, Hu Y, Fan C, Wei Z (2020) Q-value path decomposition for deep multiagent reinforcement learning. In: International Conference on Machine Learning, PMLR, pp 10706–10715
Ye N, Somani A, Hsu D, Lee WS (2017) Despot: Online pomdp planning with regularization. Journal of Artificial Intelligence Research 58:231–266
Yliniemi L, Tumer K (2014) Multi-objective multiagent credit assignment through difference rewards in reinforcement learning. In: Asia-Pacific Conference on Simulated Evolution and Learning, Springer, pp 407–418
Zhang K, Yang Z, Başar T (2021) Multi-agent reinforcement learning: A selective overview of theories and algorithms. arXiv preprint arXiv:191110635

Zhang X, Clune J, Stanley KO (2017) On the relationship between the openai evolution strategy and stochastic gradient descent. arXiv preprint arXiv:171206564

Zheng Y, Meng Z, Hao J, Zhang Z (2018a) Weighted double deep multiagent reinforcement learning in stochastic cooperative environments. In: Pacific Rim International Conference on Artificial Intelligence, Springer, pp 421–429

Zheng Y, Meng Z, Hao J, Zhang Z, Yang T, Fan C (2018b) A deep bayesian policy reuse approach against non-stationary agents. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp 962–972

Zhou M, Liu Z, Sui P, Li Y, Chung YY (2020) Learning implicit credit assignment for cooperative multi-agent reinforcement learning. arXiv preprint arXiv:200702529

Zhu Y, Mottaghi R, Kolve E, Lim JJ, Gupta A, Fei-Fei L, Farhadi A (2017) Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: 2017 IEEE international Conference on Robotics and Automation (ICRA), IEEE, pp 3357–3364