Multiagent Adversarial Collaborative Learning via Mean-Field Theory

Guiyang Luo, Hui Zhang, Haibo He, *Fellow, IEEE*, Jinglin Li, *Member, IEEE*, and Fei-Yue Wang, *Fellow, IEEE*

***Abstract*—Multiagent reinforcement learning (MARL) has recently attracted considerable attention from both academics and practitioners. Core issues, e.g., the curse of dimensionality due to the exponential growth of agent interactions and nonstationary environments due to simultaneous learning, hinder the large-scale proliferation of MARL. These problems deteriorate with an increased number of agents. To address these challenges, we propose an adversarial collaborative learning method in a mixed cooperative–competitive environment, exploiting friend-or-foe Q-learning and mean-field theory. We first treat neighbors of agent *i* as two coalitions (*i*'s friend and opponent coalition, respectively), and convert the Markov game into a two-player zero-sum game with an extended action set. By exploiting mean-field theory, this new game simplifies the interactions as those between a single agent and the mean effects of friends and opponents. A neural network is employed to learn the optimal mean effects of these two coalitions, which are trained via adversarial *max* and *min* steps. In the *max* step, with fixed policies of opponents, we optimize the friends' mean action to maximize their rewards. In the *min* step, the mean action of opponents is trained to minimize the friends' rewards when the policies of friends are frozen. These two steps are proved to converge to a Nash equilibrium. Then, another neural network is applied to learn the best response of each agent toward the mean effects. Finally, the adversarial *max* and *min* steps can jointly optimize the two networks. Experiments on two platforms demonstrate the learning effectiveness and strength of our approach, especially with many agents.**

***Index Terms*—Adversarial collaborative learning (ACL), friend-or-foe Q-learning, mean-field theory, multiagent reinforcement learning (MARL).**

Manuscript received January 11, 2020; revised July 4, 2020; accepted September 13, 2020. This work was supported in part by the Natural Science Foundation of China under Grant 61876023, and in part by the National Science Foundation under Grant ECCS 1917275. This article was recommended by Associate Editor H. Zhang. *(Guiyang Luo and Hui Zhang contributed equally to this work.)*

Guiyang Luo and Jinglin Li are with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100088, China (e-mail: luoguiyang@bupt.edu.cn; jlli@bupt.edu.cn).

Hui Zhang is with the Technology and Engineering Group, Tencent Research, Beijing 100193, China, and also with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: zhanghui2015@ia.ac.cn).

Haibo He is with the Department of Electrical, Computer and Biomedical Engineering, University of Rhode Island, Kingston, RI 02881 USA (e-mail: haibohe@uri.edu).

Fei-Yue Wang is with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, also with the Innovation Center for Parallel Vision, Qingdao Academy of Intelligent Industries, Qingdao 266109, China, and also with the Institute of Systems Engineering, Macau University of Science and Technology, Macau, China (e-mail: feiyue@ieee.org).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2020.3025491

I. Introduction

DEEP reinforcement learning (DRL) [1] is a booming area of artificial intelligence, which has emerged as a powerful approach for sequential decision-making problems. The goal of DRL is to learn a behavior policy for each agent through trial and error, such that the cumulative reward is maximized, without knowing the underlying dynamics of the environment [2], which is formalized using the Markov decision process (MDP). Classical dynamic programming methods [3]–[6] achieve significant performance for MDP with an exact mathematical model. As for MDP with an unknown model, DRL is a more attractive alternative. DRL has achieved outstanding success in a wide range of fields, that is, superhuman performance on various challenging tasks, such as video games and board games [7], etc. The flourishing and prosperity of DRL have driven the study of multiagent reinforcement learning (MARL) [8]–[11] in a wide variety of domains, including robotic teams [12], network control [13], collaborative decision support systems [14], urban traffic control [15], autonomous driving [16], etc.

Existing DRL approaches, such as Q-learning or policy gradient, are poorly suited to MARL [17]. In MARL, individual agents can no longer perceive their environment as being stationary since it is also influenced by other agents' activities [18]. This environment nonstationarity prevents the system from a proper convergence to an equilibrium. Techniques, such as centralized training and exploration with decentralized execution (CTEDE) [19], opponent modeling [20], and communications between agents [21], have been proposed recently to alleviate the nonstationarity environment problem. Besides, the increased number of agents would result in the exponential growth of agent interactions, since changes in the policy of one agent will affect that of the others. This phenomenon is known as the curse of dimensionality problem [22], which makes MARL difficult to scale up to realistic multiagent problems [23]. Deep neural networks, with powerful generalization ability, have been widely applied to directly approximate the policy or the value function.

The nature of the interactions between agents can either be cooperative, competitive, or both, and many algorithms

are designed only for a particular nature of interaction [24]–[27]. In a mixed cooperative–competitive MARL system, an agent can cooperate with other agents to perform collaborative strategies to maximize their rewards, which may be minimized by other competitive agents [28]. In such a scenario, equilibrium plays a vital role in optimizing the strategies, since each agent learns not only the cooperative strategies but also competitive behavior to confront the adversary [29]. Littman [30] treated agents as either friends or foes in a multiagent general-sum game, with friends being assumed to work together to maximize their values, while foes are supposed to cooperatively minimize the friends' rewards, which has been proved to converge to a Nash equilibrium. However, it still suffers from the curse of dimensionality due to joint state–action pairs of a large population of agents.

This article proposes an adversarial collaborative learning (ACL) method that captures the nature of cooperation and competition in a distributed MARL system. ACL gradually addresses two simplified tasks sequentially rather than the complex approximation task directly. Inspired by friend-or-foe Q-learning [30], we treat all (neighboring) agents as two coalitions, that is, friend and opponent coalition. In the adversarial process, these two coalitions sequentially improve their strategies via adversarial training, which has proven to converge to a Nash equilibrium. Meanwhile, the mean-field theory is applied to reduce the dimensionality, which simplifies the interactions as those between a single agent and the mean effects of two coalitions.

Specifically, we first convert a Markov game into a two-player zero-sum game with an extended action set, with each coalition as a super player, that is, for each agent, its friends in the neighborhood act as a coalition, which maximizes their rewards, and its opponents are treated as another coalition, which minimizes the rewards of the first coalition (friends). With the help of mean-field theory, we first learn the mean actions of two coalitions via adversarial training, which consists of *max* and *min* steps. During the *max* step, we first froze the strategies of opponents, and try to optimize the strategies of friends to maximize their rewards. In the *min* step, the opponents are trained to minimize the friends' rewards. We theoretically prove that iterating the above two adversarial training steps would result in a Nash equilibrium. Furthermore, another network is adopted to respond optimally to the mean effects. The adversarial training is further applied to jointly optimize the two networks, iterating over *max* and *min* steps. Experiments on the battle game and multiagent particle environment (MPE) demonstrate the learning effectiveness and strength of our approach.

In particular, the main contributions of this article are summarized as follows.

1) We propose an ACL method in a mixed cooperative–competitive environment, in which each agent can efficiently learn not only the cooperative strategies with teammates but also competitive behaviors to confront the adversaries.
2) We convert the N-player Markov game into a two-player zero-sum game with an extended action set, which translates the complex approximation task into two simplified tasks. Specifically, we first learn the mean effects of two coalitions via adversarial training. Then, another network is adopted to respond optimally for each agent to the mean effects.
3) We theoretically prove that our proposed approach converges to a Nash equilibrium.

The remainder of this article is organized as follows. Section II reviews related works. Section III introduces the background and preliminary for MARL. In Section IV, we introduce the ACL formalism and in Section V, we present the implementation details for the ACL algorithm. The experimental evaluation and results are discussed in Section VI. Finally, Section VII concludes this work.

II. Related Work

The environment nonstationarity and the curse of dimensionality hinder the large-scale applications of MARL. A tremendous number of research works have been reported recently to address the above challenges, e.g., CTEDE, opponents modeling, communications between agents, and mean-field theory.

A. Centralized Training and Exploration With Decentralized Execution

By accessing the observations and actions of the opponents during training, agents do not experience unexpected changes in the dynamics of the environment, which results in the stabilization of the procedure [31]. Foerster *et al.* [19] proposed a multiagent actor–critic (AC) method, which adopted a single centralized critic to estimate the Q-function and decentralized actors to optimize the agents' policies. Furthermore, they used a counterfactual baseline that marginalizes out a single agent's action, while keeping the other agents' actions fixed, to address the challenge of multiagent credit assignment. Lowe *et al.* [28] proposed a simple extension of the deterministic policy gradient (MADDPG) method [32] where the critic is augmented with extra information about the policies of other agents, while the actor only has access to local information. Another extension of MADDPG is proposed by Li *et al.* [33], who adopted Minimax Q-learning in the critic to exhibit robustness against different opponents with altered policies. Corder *et al.* [17] augmented the centralized training phase with generative modeling so that agents may infer other agents' observations when provided with locally available context. COMA [19] is a new multiagent policy gradient method to address the challenges of multiagent credit assignment. They adopt the framework of centralized training with decentralized execution, which allows the use of joint action and all available state information in training and only uses local information (i.e., the agent's observations) at test time.

B. Opponents Modeling

By reasoning about other agents' intentions and predicting their behavior, the training process of the agents might be stabilized. Modeling other agents in multiagent systems has been widely studied and offers many research opportunities [20].

He *et al.* [34] presented neural-based models that jointly learn a policy and the behavior of opponents. Their approach consists of a policy learning module that predicts Q-values and an opponent learning module that infers opponent strategy. The combination of the two modules is done either by concatenating their hidden states or by the use of a mixture of experts, which enables faster learning and even allows modeling of changing opponent behavior. Raileanu *et al.* [35] proposed an approach where agents use their policy to predict the behavior of other agents. This method employs an AC architecture and reuses the same network for estimating the goals of the other agents. Foerster *et al.* [36] proposed a method in which each agent shapes the anticipated learning of the other agents in the environment. The proposed learning rule includes an additional term that accounts for the impact of one agent's policy on the anticipated parameter update of the other agents.

C. Communications Between Agents

Agents exchange information about their observations, actions, and intentions through communications, which could potentially stabilize the training. Foerster *et al.* [21] proposed deep distributed recurrent Q-Networks, where all agents share the same hidden layers and learn to communicate to solve riddles. Singh *et al.* [37] controlled continuous communication with a gating mechanism and adopted individualized rewards for each agent to gain better performance and scalability while fixing credit assignment issues. The gate either allows or blocks communication between agents. These two approaches assume that all agents have access to the hidden layers of the other agents. Mordatch and Abbeel [38] proposed a model that takes the messages of other agents as input and learns to output action and a new communication message.

D. Mean-Field Theory

Scalability is of great significance to MARL, since an agent also observes the other agents as part of the environment. Consequently, as the number of agents increases, the dimension of joint action increases exponentially. There have been several approaches applying mean-field approximation for reinforcement learning to improve the scalability. Mguni *et al.* [39] proposed a method for computing closed-loop optimal policies that scales independently of the number of agents, using mean-field games. Their method converges to optimal behavior with an unbounded number of interacting adaptive learners. Yang *et al.* [40] adopted mean-field theory to approximate the interactions within the population of agents as those between a single agent and the average effect from the overall population or neighboring agents. Yang *et al.* [40] exploited a discrete-time mean-field game to understand the aggregate effect of individual actions and predict the temporal evolution of the population distributions. Hu *et al.* [41] approximated the effect of other agents on a single agent by an averaged effect. They derived a Fokker–Planck equation that describes the evolution of the probability distribution of Q-value in the agent population.

Our work follows the same direction as Littman [42], Hu and Wellman [43], and Bowling and Veloso [44]. In particular, Littman [42] introduced a Q-learning algorithm called Minimax-Q for zero-sum two-player games. Hu and Wellman [43] extended it to the general-sum stochastic games by using a Nash equilibrium computation in the learning rule. Unfortunately, the convergence proof is incomplete. To deal with this, Bowling [45] proposed a thorough proof by strengthening the convergence conditions of the algorithm. Littman [30] presented the friend-or-foe Q-learning algorithm in which a learner is told to treat each agent as either a "friend" or "foe." Bowling and Veloso [44] examined the learning problem in the framework of Markov games to improve the convergence.

However, as these above approaches still correlate to the joint state–action space, the curse of dimensionality cannot be avoided as the number of agents grows larger. Our work addresses this issue by employing the mean-field approximation over the joint action space. The closest to our problem setting is that of Yang *et al.* [40], who treated the interactions within the population of agents as those between a single agent and the mean effect of the neighboring agents. There exist the following differences between this method and our proposed ACL algorithm.

1) During the training phase, we treat the neighboring agents as two coalitions, the friends and opponents. The friends work together to maximize their rewards while the opponents cooperate to minimize the friends' rewards.
2) We apply adversarial training to guide each coalition on how to cooperate as well as how to compete, which consists of *max* and *min* steps.

III. Preliminary

A. Markov Game

MARL can be modeled as an extended MDP, which is also known as a Markov game. A Markov game with N players can be represented by $(N, \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \mathcal{P}, \{r_i\}_{i=1}^N, \gamma)$, where $\mathcal{S}$ is the state space describing the environment and $\mathcal{A}_i$ is the action space for agent i. The environment responds to the joint action $\boldsymbol{a} \triangleq (a_1, \ldots, a_N)$ by changing its state from $s \in \mathcal{S}$ to some $s' \in \mathcal{S}$ according to the transition function $\mathcal{P}$. This transition function characterizes the stochastic evolution of states, that is, with current state $s \in \mathcal{S}$, the agents take actions $\boldsymbol{a}$ and the state transitions to s' with probability $\mathcal{P}(s'/s)$. Agent i will receive a reward $r_i(s, \boldsymbol{a})$. $\gamma \in (0, 1)$ is the discount factor across time.

The behavior of agents is described by their policies, which specifies how an agent responds to their observations. The joint policy of all agents can be denoted as $\boldsymbol{\pi} = [\pi_1, \pi_2, \ldots, \pi_N]$. The value function of agent i at state $s \in \mathcal{S}$ is defined as the expected cumulative discounted future reward

$$V_i^{\boldsymbol{\pi}}(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\pi}, \mathcal{P}}[r_i(t) | s_0 = s, \boldsymbol{\pi}]. \tag{1}$$

The action-value function (also known as Q-function) $Q_i^{\boldsymbol{\pi}}$ gives the expected reward obtained by the joint policy $\boldsymbol{\pi}$ from any state–action $(s, \boldsymbol{a})$ pair, which is defined by

$$Q_i^{\boldsymbol{\pi}}(s, \boldsymbol{a}) = r_i(s, \boldsymbol{a}) + \gamma \mathbb{E}_{s' \sim \mathcal{P}}\left[V_i^{\boldsymbol{\pi}}(s')\right] \tag{2}$$

where s' is the state at the next time step.

B. Nash Q-Learning

Each one-stage general-sum N-player game has a Nash equilibrium. This equilibrium is defined for a set of one-stage policies $\boldsymbol{\pi}^* = [\pi_1^*, \pi_2^*, \ldots, \pi_N^*]$, if single player i has nothing to gain by unilaterally changing only its own strategy, that is

$$r_i\left(\left[\pi_1^*, \ldots, \pi_i^*, \ldots, \pi_N^*\right]\right) \geq r_i\left(\left[\pi_1^*, \ldots, \pi_i, \ldots, \pi_N^*\right]\right) \quad (3)$$

for all other policies π_i and $1 \leq i \leq N$. A game can have more than one Nash equilibrium, and the expected payoff of player i can vary depending on the equilibrium considered.

The Nash Q-learning rule [43] updates a set of approximate Q-functions each time with an experience $\langle s, \boldsymbol{a}, s', \{r_1(s, \boldsymbol{a}), r_2(s, \boldsymbol{a}), \ldots, r_N(s, \boldsymbol{a})\}\rangle$ by the following equation:

$$\begin{aligned} Q_i^{\pi}(s, \boldsymbol{a}) \triangleq\ & (1 - \alpha(t))Q_i^{\pi}(s, \boldsymbol{a}) \\ & + \alpha(t)\left(r_i(s, \boldsymbol{a}) + \text{Nash}_i\left(s', Q_1, Q_2, \ldots, Q_N\right)\right) \end{aligned} \quad (4)$$

where $\alpha(t)$ is a set of learning rate, and $\text{Nash}_i(s', Q_1, Q_2, \ldots, Q_N) = V_i(s', \boldsymbol{\pi}^*)$, $\boldsymbol{\pi}^*$ are a set of Nash equilibrium for the one-stage game defined by the Q-functions $(Q_1, Q_2, \ldots, Q_N)$ at the state s'. For a single-player game, the Nash_i function is a simple maximization, that is, $\text{Nash}(s', \boldsymbol{\pi}) = \max_{\boldsymbol{a}} Q(s', \boldsymbol{a})$, which is Q-learning.

C. Friend or Foe

Littman proposed friend-or-foe Q-learning [30], which treated all other players as friends or foes, where player i's friends are assumed to work together to maximize i's value function, while i's foes are supposed to work together to minimize i's value function. Therefore, the Nash_i function is defined as

$$\begin{aligned} \text{Nash}_i\left(s', Q_1, Q_2, \ldots, Q_N\right) = & \max_{\pi \in \prod(\mathcal{A}_{f_1} \times \cdots \times \mathcal{A}_{f_m})} \\ & \min_{a_{o_1}, \ldots, a_{o_k}} \sum_{a_{f_1} \times \cdots \times a_{f_m}} \pi\left(a_{f_1}\right) \cdots \pi\left(a_{f_m}\right) Q_i(s, \boldsymbol{a}) \end{aligned} \quad (5)$$

where $\{f_1, f_2, \ldots, f_m\}$ is the set of friends and $\{o_1, o_2, \ldots, o_k\}$ is the set of foes. However, it still suffers from the curse of dimensionality, since the Q function is related to the joint action of all players.

D. Mean-Field Theory

The mean-field theory first appeared in physics for describing phase transitions in the work of Pierre Curie [46]. It studies the behavior of high-dimensional random (stochastic) models, with many individual components that interact with each other. In mean-field theory, the effect of all the other individuals on any given individual is approximated by a single averaged effect, thus reducing a many-body problem to an effective one-body problem. By replacing all interactions to any one body with an average or effective interaction, the behavior of a complex system can be obtained at a lower computational cost.

The mean-field theory has been applied to a wide range of fields outside of physics, including graphical models, neuroscience, artificial intelligence, epidemic models, queueing theory, computer network performance, and game theory.

IV. Adversarial Collaborative Learning

In an MARL system, all agents strategically and simultaneously evaluate their value functions based on the joint actions $\boldsymbol{a}$. Consequently, the dimension of joint action $\boldsymbol{a}$ grows proportionally with the number of agents N. To address this challenge, inspired by friend-or-foe Q-learning [30] and mean-field theory [40], we first convert the N-player Markov game into a two-player zero-sum game with an extended action set, which learns the Nash-equilibrium actions of these two players. Then, another network is adopted to learn the best response toward the Nash-equilibrium actions.

A. Constructed Two-Player Zero-Sum Game

This article investigates a mixed cooperative–competitive MARL, where agents require to collaborating with friends while competing with opponents. Each player i can identify its friends as $\Omega_i^f \triangleq \{f_1, f_2, \ldots, f_m\}$ (denoted thereafter as friends or friend coalition) and its opponents as $\Omega_i^o \triangleq \{o_1, o_2, \ldots, o_k\}$ (denoted thereafter as opponents or opponent coalition), where m and k are the number of friends and opponents, respectively, $\Omega_i^o \cup \Omega_i^f = \Omega_i$. For clarity, we assume $i \in \Omega_i^f$. We treat friend coalition Ω_i^f and opponent coalition Ω_i^o as two super agents with an extended action set and thus convert the multiplayer game into a two-player zero-sum game. Consequently, the original N-player Markov game $(N, \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \mathcal{P}, \{r_i\}_{i=1}^N, \gamma)$ can be converted into a new *two*-player Markov game, denoted by

$$\left(2, \mathcal{S}, \left[\boldsymbol{\mathcal{A}}_f^{sp}, \boldsymbol{\mathcal{A}}_o^{sp}\right], \mathcal{P}^{sp}, \left[r_f^{sp}, r_o^{sp}\right], \gamma\right) \quad (6)$$

where $\boldsymbol{\mathcal{A}}_f^{sp} = [\mathcal{A}_{f_1}, \mathcal{A}_{f_2}, \ldots, \mathcal{A}_{f_m}]$, $\boldsymbol{\mathcal{A}}_o^{sp} = [\mathcal{A}_{o_1}, \mathcal{A}_{o_2}, \ldots, \mathcal{A}_{o_k}]$, $r_f^{sp} = (1/|\Omega_i^f|)\sum_{f_j \in \Omega_i^f} r_{f_j}$, and $r_o^{sp} = (1/|\Omega_i^o|)\sum_{o_j \in \Omega_i^o} r_{o_j}$. The super player (friend coalition) takes action $\boldsymbol{a}_f^{sp} = [a_{f_1}, a_{f_2}, \ldots, a_{f_m}]$ to maximize its expected cumulative discounted reward and another super player (opponent coalition) responds with $\boldsymbol{a}_o^{sp} = [a_{o_1}, a_{o_2}, \ldots, a_{o_k}]$ to minimize the reward of friend coalition. The Q-function of the first super player can be denoted by

$$\begin{aligned} Q_f^{sp}\left(s, \boldsymbol{a}_f^{sp}, \boldsymbol{a}_o^{sp} | \boldsymbol{\pi}^{sp}\right) &= \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\pi}, \mathcal{P}}\left[r_f^{sp}(t) | s, \boldsymbol{a}_f^{sp}, \boldsymbol{a}_o^{sp}\right] \\ &= \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\pi}, \mathcal{P}} \\ &\quad \times \left[\left(\frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} r_{f_j}(t)\right) | s, \boldsymbol{a}_f^{sp}, \boldsymbol{a}_o^{sp}\right] \\ &= \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\pi}, \mathcal{P}}\left[r_{f_j}(t) | s, \boldsymbol{a}_f^{sp}, \boldsymbol{a}_o^{sp}\right] \end{aligned}$$

$$= \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} Q_{f_j}\left(s, \boldsymbol{a}_f^{sp}, \boldsymbol{a}_o^{sp}\right)$$
$$= \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} Q_{f_j}(s, \boldsymbol{a}). \tag{7}$$

In the constructed two-player Markov game, the two super players compete with each other, since one super player consists of friends, cooperating with each other to maximize their total rewards, and the other super player is formed by opponents, working together to minimize value function of the first super player. Consequently, the value function of the first super player can be denoted as

$$V_f^{sp}\left(s|\boldsymbol{\pi}_f^{sp}, \boldsymbol{\pi}_o^{sp}\right) = \max_{\boldsymbol{a}_f^{sp} \in \boldsymbol{\mathcal{A}}_f^{sp}} \min_{\boldsymbol{a}_o^{sp} \in \boldsymbol{\mathcal{A}}_o^{sp}} \sum_{\boldsymbol{a}_f^{sp}} \sum_{\boldsymbol{a}_0^{sp}} \times \boldsymbol{\pi}_f^{sp}\left(\boldsymbol{a}_f^{sp}\right) \boldsymbol{\pi}_o^{sp}\left(\boldsymbol{a}_o^{sp}\right) Q_f^{sp}\left(s, \boldsymbol{a}_f^{sp}, \boldsymbol{a}_o^{sp}\right) \tag{8}$$

where $\boldsymbol{\pi}_o^{sp}$ is the policy of the opponents and $\boldsymbol{\pi}^{sp} = [\boldsymbol{\pi}_o^{sp}, \boldsymbol{\pi}_f^{sp}]$. Given a fixed policy of opponents $\hat{\boldsymbol{\pi}}_o^{sp}$, the value function can be denoted as

$$V_f^{sp}(s) = \max_{\boldsymbol{a}_f^{sp} \in \boldsymbol{\mathcal{A}}_f^{sp}} \sum_{\boldsymbol{a}_f^{sp}} \boldsymbol{\pi}_f^{sp}\left(\boldsymbol{a}_f^{sp}\right) Q_f^{sp}\left(s, \boldsymbol{a}_f^{sp}, \hat{\boldsymbol{a}}_o^{sp}\right), \quad \hat{\boldsymbol{a}}_o^{sp} \sim \hat{\boldsymbol{\pi}}_o^{sp}. \tag{9}$$

Then, given a fixed policy of friends $\hat{\boldsymbol{\pi}}_f^{sp}$, the value function can be denoted as

$$V_f^{sp}(s) = \min_{\boldsymbol{a}_o^{sp} \in \boldsymbol{\mathcal{A}}_o^{sp}} \sum_{\boldsymbol{a}_0^{sp}} \boldsymbol{\pi}_o^{sp}\left(\boldsymbol{a}_o^{sp}\right) Q_f^{sp}\left(s, \hat{\boldsymbol{a}}_f^{sp}, \boldsymbol{a}_o^{sp}\right), \quad \hat{\boldsymbol{a}}_f^{sp} \sim \hat{\boldsymbol{\pi}}_f^{sp}. \tag{10}$$

Based on (9) and (10), we can train the multiagent system sequentially. First, given an initial (random) scheme of the opponents, the friend coalition learns to maximize the total rewards (*max* step). Then, with the fixed improved policy of the friends, the opponents adjust their strategies to minimize the rewards of the first super player (*min* step). According to Theorem 1, repeat the above two steps can result in a Nash equilibrium.

Theorem 1: In a finite-state stochastic game, the Nash Q-learning with $\text{Nash}_i(s', Q_1, Q_2, \ldots, Q_N)$ that is computed by (8) converges to the Nash Q-value, under several assumptions.

Proof: The proof can be found in the Appendix. ■

B. Mean-Field Theory Approximation

All agents act strategically and evaluate simultaneously their value functions based on the actions of other agents. Meanwhile, the dimension of joint action grows proportionally with the number of agents. Consequently, it becomes infeasible to learn the standard Q-function directly [40]. In our constructed two-player zero-sum game, each super player has a tremendous action space $\boldsymbol{\mathcal{A}}_f^{sp}$ or $\boldsymbol{\mathcal{A}}_o^{sp}$. Inspired by mean-field theory [40], we apply it to this constructed two-player zero-sum game to reduce the dimensionality.

During the *max* step, given a fixed policy of opponents $\hat{\boldsymbol{\pi}}_o^{sp}$, we factorize the Q-function using only the pairwise local interactions between $f_l \in \Omega_i^f$ and $\hat{\boldsymbol{a}}_o^{sp} \sim \hat{\boldsymbol{\pi}}_o^{sp}$

$$Q_f^{sp}\left(s, \boldsymbol{a}_f^{sp}, \hat{\boldsymbol{a}}_o^{sp}\right) = \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} Q_{f_j}\left(s, \boldsymbol{a}_f^{sp}, \hat{\boldsymbol{a}}_o^{sp}\right) \approx \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} \left(\frac{1}{\left|\Omega_i^f\right|} \sum_{f_l \in \Omega_i^f} Q_{f_j}\left(s, a_{f_l}, \hat{\boldsymbol{a}}_o^{sp}\right) \right). \tag{11}$$

Assume $\bar{a}_f$ as the mean action of friends Ω_i^f, that is, $\bar{a}_f = (1/|\Omega_i^f|) \sum_{f_l \in \Omega_i^f} a_{f_l}$. Consequently, each action a_{f_i} can be expressed as the mean action $\bar{a}_f$ plus a fluctuation δ_{f_i}, that is, $a_{f_i} = \bar{a}_f + \delta_{f_i}$. Assume $Q_{f_j}(s, \boldsymbol{a})$ is twice differentiable with respect to each action a_{f_i}, then, $Q_{f_j}(s, a_{f_l}, \hat{\boldsymbol{a}}_o^{sp})$ can be expended and expressed as

$$Q_{f_j}\left(s, a_{f_l}, \hat{\boldsymbol{a}}_o^{sp}\right) \approx Q_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right) + Q'_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right)\delta_{f_l} + 0.5 Q''_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right)\delta_{f_l}^2. \tag{12}$$

Combining (11) and (12), we have

$$\begin{aligned}
Q_f^{sp}\left(s, \boldsymbol{a}_f^{sp}, \hat{\boldsymbol{a}}_o^{sp}\right) &\approx \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} \times \left(\frac{1}{\left|\Omega_i^f\right|} \sum_{f_l \in \Omega_i^f} \Big[Q_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right) + Q'_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right)\delta_{f_l} + + 0.5 Q''_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right)\delta_{f_l}^2 \Big] \right) \\
&\approx \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} \left(Q_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right) + Q'_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right) \times \frac{1}{\left|\Omega_i^f\right|} \sum_{f_l \in \Omega_i^f} \delta_{f_l} + \frac{1}{2\left|\Omega_i^f\right|} \sum_{f_l \in \Omega_i^f} Q''_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right)\delta_{f_l}^2 \right) \\
&\approx \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} Q_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right) + \frac{1}{2\left|\Omega_i^f\right|^2} \sum_{f_j \in \Omega_i^f} R\left(a_{f_l}\right) \\
&\approx \frac{1}{\left|\Omega_i^f\right|} \sum_{f_j \in \Omega_i^f} Q_{f_j}\left(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}\right)
\end{aligned} \tag{13}$$

where $R(a_{f_l}) = \sum_{f_l \in \Omega_i^f} Q''_{f_j}(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp})\delta_{f_l}^2$ denotes the Taylor polynomial's remainder. The first-order term of (13) is ignored

since $(1/|\Omega_i^f|)\sum_{f_l\in\Omega_i^f}\delta_{f_l} = (1/|\Omega_i^f|)\sum_{f_l\in\Omega_i^f}a_{f_l} - \bar{a}_f = 0$. Meanwhile, if $Q_{f_j}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ is M-smooth, the remainder $R(a_{f_l})$ is bounded within a symmetric interval $[-2M, 2M]$ [40]. Consequently, $Q_f^{sp}(s,\boldsymbol{a}_f^{sp},\hat{\boldsymbol{a}}_o^{sp})$ is related to only the mean action $\bar{a}_f$ of friends Ω_i^f. Therefore, we have

$$Q_f^{sp}\left(s,\boldsymbol{a}_f^{sp},\hat{\boldsymbol{a}}_o^{sp}\right)\approx Q_f^{sp}\left(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right),\hat{\boldsymbol{a}}_o^{sp}\sim\hat{\boldsymbol{\pi}}_o^{sp}. \tag{14}$$

Similarly, in the *min* step, with a fixed policy of friends $\hat{\boldsymbol{\pi}}_f^{sp}$

$$Q_o^{sp}\left(s,\hat{\boldsymbol{a}}_f^{sp},\boldsymbol{a}_o^{sp}\right)\approx Q_o^{sp}\left(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o\right),\hat{\boldsymbol{a}}_f^{sp}\sim\hat{\boldsymbol{\pi}}_f^{sp}. \tag{15}$$

Iterations over the *max* step and *min* step would result in an equilibrium over the mean action of friends ($\bar{a}_f$) and opponents ($\bar{a}_o$). This equilibrium converges to the Nash Q-value, which has been proved in the Appendix. Then, the influence between agent i and its neighbors Ω_i is simplified as that between agent i and two virtual agents, the friend coalition and the opponent coalition, which is abstracted by the mean effects of both super players. Consequently, $Q_i(s,\boldsymbol{a})$ can be simplified as $Q_i(s,a_i,\bar{a}_f,\bar{a}_o)$. This simplification can be interpreted as each agent i responses optimally to the mean effects $(\bar{a}_f,\bar{a}_o)$ of two coalition. Finally, this improved Q-function $Q_i(s,a_i,\bar{a}_f,\bar{a}_o)$ can be improved via *max* and *min* step in return. This Q-function will eventually converge to the Nash Q-value as well, which has been proved in [40].

It is worth noting that the triplewise approximation of the agent, friend coalition, and opponent coalition, which significantly reduces the complexity of the interactions among agents, still preserves global interactions between any pair of agents implicitly [40]. The approximation error of the mean-field theory can be found in [47] and [48].

V. Implementation

We adopt neural networks to implement the ACL Q-functions $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$, $Q_o^{sp}(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o)$, and $Q_i(s,a_i,\bar{a}_f,\bar{a}_o)$. We first adopt adversarial training to train $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ and $Q_o^{sp}(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o)$ via the *max* and *min* step, which is referred to as adversarial training initialization (ATI). This training would output an equilibrium mean action $\bar{a}_f$ and $\bar{a}_o$. Then, we design another network to learn $Q_i(s,a_i,\bar{a}_f,\bar{a}_o)$, which will learn an action a_i for each agent to optimally respond to the mean effects. We could also incorporate the *max* and *min* step to further improve $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ and $Q_o^{sp}(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o)$, respectively, which we refer to as min–max enhancement (MME).

A. Centralized Training and Exploration With Decentralized Execution

Each agent must learn the cooperative strategy as well as adversarial strategy, since the agent can cooperate with its friends to maximize reward and also confront the opponents. Therefore, it is significantly challenging for the agent to learn an equilibrated strategy. To handle this challenge, we adopt the centralized training to guide each agent in learning cooperative strategy as well as adversarial strategy. During the training phase, friends and opponents are distinguished. The friends Ω_i^f and opponents Ω_i^o for agent i are treated as two super agents, which help in converting the multiplayer Markov game into a two-player zero-sum game with an extended action set. We first adopt neural networks to approximate $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ and $Q_o^{sp}(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o)$. These networks learn the mean effects of two coalitions. However, during the execution phase, each agent cannot distinguish the friends or opponents. Each agent has learned the mean effects of the two coalitions given a state during the centralized training phase. Consequently, using $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ and $Q_o^{sp}(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o)$, each agent can respond optimally to the mean effects $\bar{a}_f$ and $\bar{a}_o$.

During the experiments, we evaluate the proposed ACL algorithm with several other approaches. The experiments are conducted with the same condition during the execution phase, that is, each agent has no information about friends and opponents.

B. Adversarial Training Initialization

We first learn $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ and $Q_o^{sp}(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o)$ via the *max* and *min* step, which is shown in Fig. 1(a).

1) Max Step: With an experience $\langle s,\boldsymbol{a},s',[r_1,r_2,\ldots,r_N]\rangle$, for each agent i, we can distinguish its friends Ω_i^f and opponents Ω_i^o. The mean action of friends can be denoted as $\bar{a}_f = (1/|\Omega_i^f|)\sum_{f_l\in\Omega_i^f}a_{f_l}$ and the reward can be denoted as $r_f^{sp} = (1/|\Omega_i^f|)\sum_{f_l\in\Omega_i^f}r_{f_l}$. Then, the mean-field Q-function $Q_f^{sp}(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp})$ of the friend coalition is updated in a recurrent manner as

$$\begin{aligned}Q_{f,(t+1)}^{sp}\left(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right) = {}& Q_{f,(t)}^{sp}\left(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right)+\alpha(t)\\ &\times\left(Q_f^{\text{target}}-Q_{f,(t)}^{sp}\left(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right)\right)\end{aligned} \tag{16}$$

where $\alpha(t)$ is the learning rate and the target value Q_f^{target} is represented by

$$Q_f^{\text{target}} = r_f^{sp}+\gamma\max_{\bar{a}_f}Q_{f,(t)}^{sp}\left(s',\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right). \tag{17}$$

Finally, agent $j\in\Omega_i^f$ adopt a Boltzmann exploration policy expressed as

$$\pi_j\left(a_j|s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right)=\frac{\exp\left(-\beta Q_{f,(t)}^{sp}\left(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right)\right)}{\sum_{\bar{a}_f}\exp\left(-\beta Q_{f,(t)}^{sp}\left(s,\bar{a}_f,\hat{\boldsymbol{a}}_o^{sp}\right)\right)}. \tag{18}$$

By iterating (16)–(18), the mean actions $\bar{a}_f$ and the corresponding policies $\pi_j\in\boldsymbol{\pi}_f^{sp}, j\in\Omega_i^f$ for friends improve alternatively.

2) Min Step: In the *min* step, the opponents Ω_i^o cooperate with each other to minimize the reward of the friends. Therefore, the target value Q_o^{target} in *min* step is expressed as

$$Q_o^{\text{target}} = r_f^{sp}+\gamma\min_{\bar{a}_o}Q_{f,(t)}^{sp}\left(s',\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o\right). \tag{19}$$

Then, each opponent $j\in\Omega_i^o$ takes the action with minimum value and the Boltzmann exploration policy can be expressed as

$$\pi_j\left(a_j|s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o\right)=\frac{\exp\left(\beta Q_{f,(t)}^{sp}\left(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o\right)\right)}{\sum_{\bar{a}_o}\exp\left(\beta Q_{f,(t)}^{sp}\left(s,\hat{\boldsymbol{a}}_f^{sp},\bar{a}_o\right)\right)}. \tag{20}$$

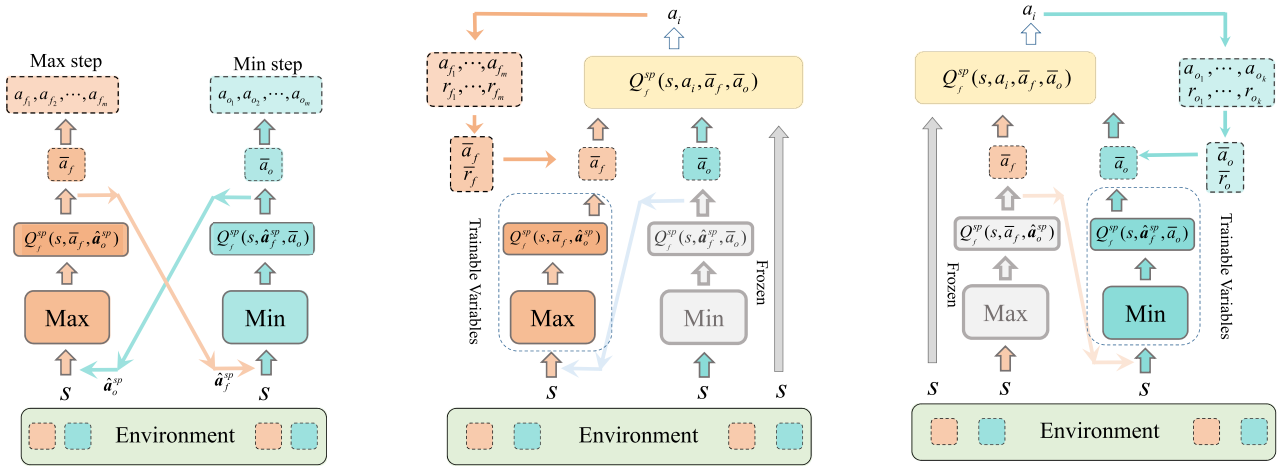

Fig. 1. ACL in a mixed cooperative–competitive environment. In (a), the mean effects ($\bar{a}_f$ and $\bar{a}_o$) of friends and opponents are trained via adversarial *max* and *min* step, respectively. In the *max* step ($Q_f^{sp}(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp})$), with fixed policies of opponents, we optimize the friends' mean action to maximize their rewards. In the *min* step ($Q_o^{sp}(s, \hat{\boldsymbol{a}}_f^{sp}, \bar{a}_o)$), the mean action of opponents is trained to minimize the friends' rewards when the policies of friends are frozen. In (b) and (c), each agent learns the best response toward the mean effects, via the adversarial *max* and *min* steps. (a) Initialization. (b) Max step. (c) Min step.

In the *max* step, the exploration policy shown in (18) would guide each friend to choose an action that would maximize the Q-function. However, in the *min* step, the exploration policy shown in (20) always chooses an action that would minimize the Q-function. Iterating these two steps, this algorithm would generate a Nash equilibrium, which is proved in the Appendix.

At the ATI phase, different from existing opponent modeling approaches, each agent controlled by ACL models the intentions and policies of other agents by learning the mean actions of its friends and opponents via adversarial training. Our proposed approach achieves significant improvements over existing opponent modeling approaches for application in large-scale agents from the following aspects.

1) *Reduced Complexity and Improved Efficiency:* Take the opponent modeling approach in [35] as an example, it suggested an approach where agents apply their network for estimating the goals of the other agents. Therefore, in the case of large-scale agents, each agent requires evaluating all other agents, which is of low efficiency. However, in our proposed approach, each agent only evaluates the mean actions of the friend coalition and opponent coalition, which can be proceeded at low computational complexity.
2) *Characteristics of Agents:* In a mixed cooperative–competitive MARL system, agents with different roles have different behaviors, e.g., the friends Ω_i^f cooperate with agent i in maximizing their rewards while the opponents Ω_i^o compete with agent i in minimizing the reward of agent i. Therefore, the existing opponent modeling approaches (e.g., [35]) could not seize these differences. However, our proposed approach considers the characteristics of agents, modeling the mean action of friends and opponents separately, via adversarial training. By modeling the intentions and policies of other agents, the training process of the agents might be stabilized, which could potentially deal with the nonstationarity problem in MARL [31].

C. Min–Max Enhancement

The ATI network would foresee the mean action of friends $\bar{a}_f$ and opponents $\bar{a}_o$. For each agent i with state s, $\bar{a}_f$ and $\bar{a}_o$ can be regarded as the empirical distribution of the behavior of their neighbors. Its friends and opponents act with different strategies, with the friends maximizing their rewards while the opponents minimizing the friends' reward. Each agent reacts to $\bar{a}_f$ and $\bar{a}_o$ accordingly, and we achieve this by training the Q-function $Q_i(s, a_i, \bar{a}_f, \bar{a}_o)$ via standard Q-learning. Each agent i is trained with experience $\langle s, \boldsymbol{a}, s', [r_1, r_2, \ldots, r_N]\rangle$ by minimizing the loss function

$$\text{Ł}^{\text{MME}} = \left(r_i + \max_{a_i} Q_i\big(s', a_i, \bar{a}_f, \bar{a}_o\big) - Q_i\big(s, a_i, \bar{a}_f, \bar{a}_o\big)\right)^2. \tag{21}$$

This new experience can further be applied to improve the ATI network. We could also incorporate *max* and *min* step into this improvement, which are shown in Fig. 1(b) and (c), respectively.

In the *max* step, we froze the min network to stabilize the opponents' strategies, and train the max network to maximize the rewards of friends. With $\bar{a}_f = (1/|\Omega_i^f|)\sum_{f_l\in\Omega_i^f} a_{f_l}$ and $r_f^{sp} = (1/|\Omega_i^f|)\sum_{f_l\in\Omega_i^f} r_{f_l}$, we can iterate (16)–(18) to improve the strategies of the friends. Therefore, we jointly minimize the loss Ł^{MME} and the loss $\text{Ł}^{max} = (Q_f^{\text{target}} - Q_f^{sp}(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp}))^2$.

Similarly, in the *min* step, we froze the max network and train the min network to minimize the rewards of friends. We jointly minimize Ł^{MME} and the loss $\text{Ł}^{\text{min}} = (Q_o^{\text{target}} - Q_o^{sp}(s, \hat{\boldsymbol{a}}_f^{sp}, \bar{a}_o))^2$.

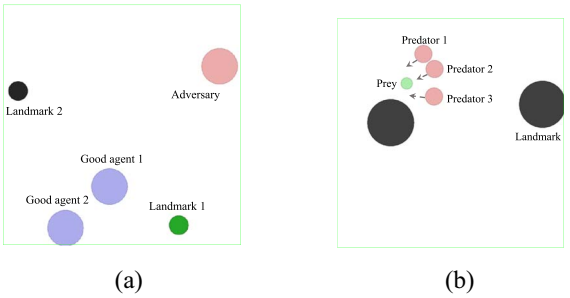

Fig. 2. Illustrations of tasks. (a) Physical deception in the MPE. (b) Predator–prey in the MPE.

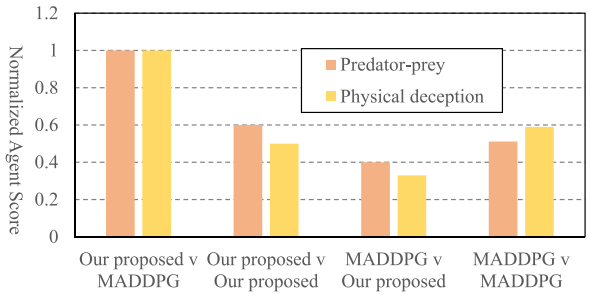

Fig. 3. Comparison between our proposed algorithm and MADDPG [28] in the MPE. Our proposed approach achieves a higher score than MADDPG in these two games.

VI. Experiments

The strength and effectiveness of our algorithm are validated in the MPE [38] and the battle game [50] as compared with several state-of-the-art methods (deep deterministic policy gradient (DDPG) [49], multiagent DDPG (MADDPG) [28], mean-field Q-learning [40], mean-field actor–critic [40], etc.). We adopt the centralized training, which is also used by MADDPG, mean-field Q-learning, and mean-field actor–critic. Each agent can distinguish its friends and opponents only during the centralized training phase. Each agent has no information about friends and opponents during the experiment evaluation as compared with other baseline approaches.

A. Multiagent Particle Environment

Environments: MPE [38] is a physically simulated 2-D environment in continuous space and discrete time. This environment consists of cooperative agents, adversarial agents, and landmarks. Both agent and landmark entities inhabit a physical location in space and poss descriptive physical characteristics, such as color and shape type. Besides, agents can direct their gaze to a location and act to move in the environment and direct their gaze. Also, they are affected by physical interactions with other agents. We perform experiments in two mixed cooperative and competitive scenarios, as shown in Fig. 2, to demonstrate the collaborative and adversarial strategies learned by our proposed ACL algorithm. We introduce these two scenarios as follows.

1) *Physical Deception:* In this game, three agents (one adversary and two good agents) cooperate to reach a single target landmark from a total of two landmarks. These agents are rewarded based on the minimum distance of any agent to the target. Only one agent needs to reach the target landmark. However, an adversary also desires to reach the target landmark. The adversary does not know which landmark is the correct one. These agents learn to spread out cooperatively and cover all landmarks to deceive the adversary.
2) *Predator–Prey:* In this variant of the classic predator–prey game, three slower cooperating agents must chase one faster adversary around a randomly generated environment with two large landmarks impeding the way. Each time the cooperative agents collide with an adversary, the agents are rewarded while the adversary is penalized.

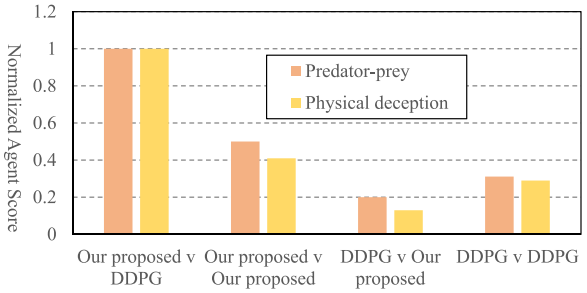

Fig. 4. Comparison between our proposed algorithm and DDPG [49] in the MPE. Our proposed approach achieves a higher score than DDPG in these two games.

Baseline Approaches: We compare our proposed approach against two baselines, which are proved successful in the MPE. The two baselines are DDPG [49] and MADDPG [28], respectively. To present a fair comparison, the same training configurations as in [28] are adopted. The environment requires both collaborations and competitions. Therefore, the agents and the adversary present diverse strategies. We design different algorithm combinations for these agents to evaluate the quality of policies learned in adversarial settings, that is, our proposed versus MADDPG, our proposed versus our proposed, MADDPG versus our proposed, and MADDPG versus MADDPG. These models are trained until convergence and then evaluated over 100 iterations. We take the averaged metrics as the final performance indicator.

Experimental Results: For the physical deception task, agents trained with our proposed algorithm are able to successfully deceive the adversary by covering all landmarks. Furthermore, the adversary score is quite low, especially when the adversary is trained with DDPG. A similar situation arises for the predator–prey task since agents trained with our proposed algorithm can achieve the highest score, as competing against the adversary controlled by other algorithms. The experimental results are shown in Figs. 3 and 4, respectively. ACL efficiently learns the correct behavior in both cases: in the physical deception, the agents learn to spread out cooperatively and cover all landmarks to deceive the adversary, while in the predator–prey, the slower agents can cooperatively catch the faster adversary.

However, this platform supports only a limited number of agents, and cannot be extended to the scenario of hundreds of agents. Our proposed ACL algorithm can effectively

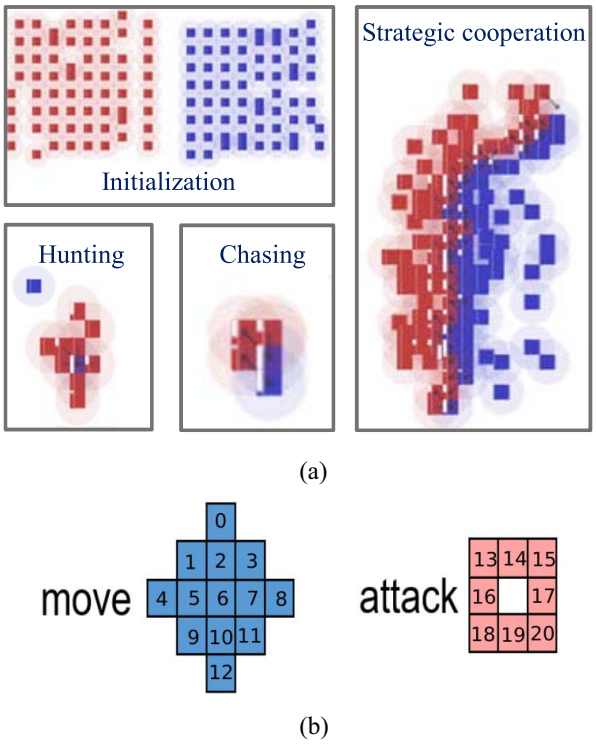

Fig. 5. Battle game is a mixed cooperative–competitive battle scenario. There exist two armies fighting against each other in a configured grid world. Agents must learn the GSC, LSC, and strategic competition, that is, cooperate with friends and compete with opponents. (a) Illustration of the battle game. (b) Action space in this game, with 13 actions for move and 8 actions for attack.

address the challenges of the curse of dimensionality due to the exponential growth of agent interactions and nonstationary environments due to simultaneous learning. Besides, ACL can learn very fast and efficiently since we divide the complex task into two simplified tasks. To further demonstrate the learning effectiveness and strength of our approach, especially with many agents, we adopt the battle game that supports the tasks and the applications that require hundreds to millions of agents.

B. Battle Game

Environments: The battle game is a mixed cooperative–competitive battle scenario developed by [50], which supports the tasks and the applications that require hundreds to millions of agents. The battle game involves two armies fighting against each other in a configured grid world, each empowered by a different MARL algorithm. As shown in Fig. 5(a), each army is initialized with the same number of homogeneous agents, and each agent can take action to move to a nearby empty grid or attack nearby agents. The discounted factor γ is set to be 0.95 and the mini-batch size is set to be 128. $\tau = 0.01$ is applied for updating the target networks. The size of the replay buffer is 10^6. The Adam optimizer with a learning rate of 0.001 and 0.0001 (learning rate is $\alpha = 10^{-4}$, and with a dynamic exploration rate linearly decays from 1.0 to 0.05) for the ATI phase and the MME phase, respectively. The view range for each agent is a circle with a radius of 6 and the attack range is a circle with a radius of 1.5. Each action is responded with a reward: -0.005 for every move, 0.2 for attacking an enemy, 5 for killing an enemy, -0.1 for attacking an empty grid, and -0.1 for being attacked or killed (the default reward setting). Agents should learn to collaborate with teammates to destroy the adversaries, getting higher rewards.

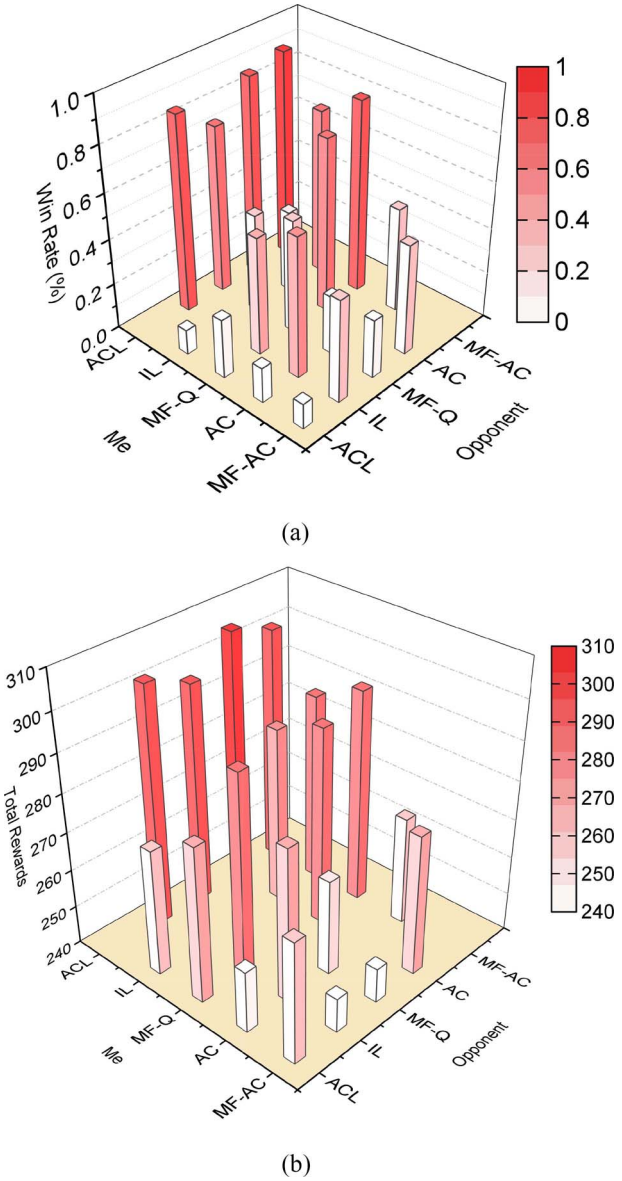

Fig. 6. Performance evaluations of ACL as compared with IL, AC, MF-Q, and MF-AC in terms of the win rate and total rewards. Our proposed ACL achieves the highest reward and win rate than other baseline methods. (a) Win rate. (b) Total rewards.

Baseline Approaches: We evaluate our proposed ACL algorithm as compared with four baselines, which are proved successful in the battle game. The four baselines are independent Q-learning (IL) [51], advantageous AC [52], and their counterparts that adopt mean-field theory, mean-field IL (MF-Q) [40], and mean-field AC (MF-AC) [40], respectively.

Experimental Results: We have conducted extensive experiments to illustrate the learning effectiveness and strength of our proposed approach. First, we evaluate the performance of our proposed ACL algorithm as compared with other baseline

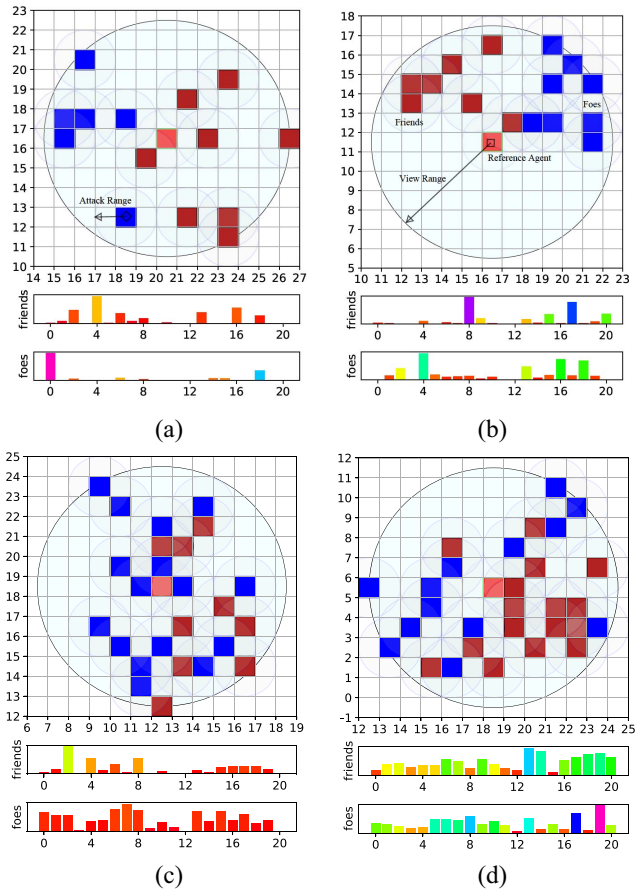

Fig. 7. Visualization for the mean effects leaned by the adversarial training initialization under four scenarios, shown in (a), (b), (c) and (d), respectively. The reference agent can see the states with the view range, and then it will learn or predict the mean actions of friends and opponents, which are encoded in the $Q_f^{sp}(s, \bar{a}_f, \hat{\boldsymbol{a}}_o^{sp})$ and $Q_o^{sp}(s, \hat{\boldsymbol{a}}_f^{sp}, \bar{a}_o)$, respectively. These two Q-functions are visualized by the histogram, indicating the potential rewards for each action under four different states.

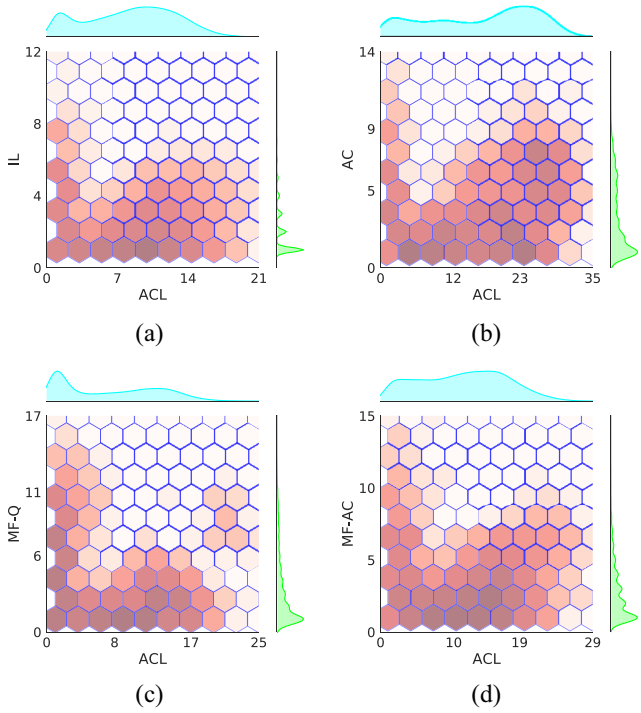

Fig. 8. Hexbin plot of the number of survivors over 2000 round of competitions. Each hexbin shows the number of survivors for two armies under 400 steps. This plot further demonstrates the performance of these algorithms. The army controlled by ACL has a greater number of survivors while another army controlled by other algorithms has a less number of survivors. (a) ACL versus IL. (b) ACL versus AC. (c) ACL versus MF-Q. (d) ACL versus MF-AC.

methods in terms of the win rate and total reward, which is shown in Fig. 6. Second, the learned mean effects of these two coalitions are visualized under four different states, which is shown in Fig. 7. This visualization demonstrates the effectiveness of ATI. Then, the hexbin plot of the number of survivors over 2000 round of competitions is shown in Fig. 8. Finally, we illustrate the states of agents at different time steps to further validate the effectiveness of the proposed ACL algorithm, which is shown in Fig. 9.

We train the above five models by self-plays and then use them for comparative battles. During the training phase, each army is initialized with 64 homogeneous agents. Agents can quickly learn the global strategic cooperation (GSC) and local strategic cooperation (LSC). GSC is shown in the right figure of Fig. 5(a)—the red army cooperates with each other trying to surround the blue army. However, as for the LSC, agents within a small local area can cooperate to chase and hunt. In the test phase, two armies that are controlled by the well-trained model of different algorithms compete with each other, that is, the agents within the same army cooperate with each other, while competing with the agents of another army (for clarity, these two armies are referred to as me and opponent, respectively). Fig. 6(a) and (b) show the win rate and total rewards over 2000 round cross-comparative experiments, respectively. Our proposed algorithm ACL outperforms MF-Q with a 76% win rate, even though that MF-Q can defeat other algorithms IL, AC, and MF-AC with a higher win rate. Specifically, ACL achieves 87%, 87%, and 91% win rate as competed against IL, AC, and MF-AC, respectively. The same results can also be found for total rewards.

Furthermore, we present the hexbin plot to show the relationship between the survivors of two armies over 2000 round of competitions, which is shown in Fig. 8. ACL can kill almost all other opponents, since the number of survivors concentrates on 0, which is shown in the density curve (the green one). However, the army controlled by ACL has a larger number of survivors, demonstrating its superiority. Opponents controlled by different algorithms have a distinct density curve of the survivor, e.g., the number of survivors is focused on 23 for the battle of ACL versus AC, while it is mainly distributed in 10 for the battle ACL versus IL.

To present a better evaluation of the cooperative and competitive strategies learned by our proposed algorithm, we have presented the snapshots of two games, which are shown in Fig. 9, where the red army is controlled by our proposed ACL algorithm, while another army is controlled by MF-AC and MF-Q, respectively. As can be shown from this figure, the army controlled by our proposed ACL algorithm can learn the cooperative strategies to besiege another army, thus having a large number of survivors.

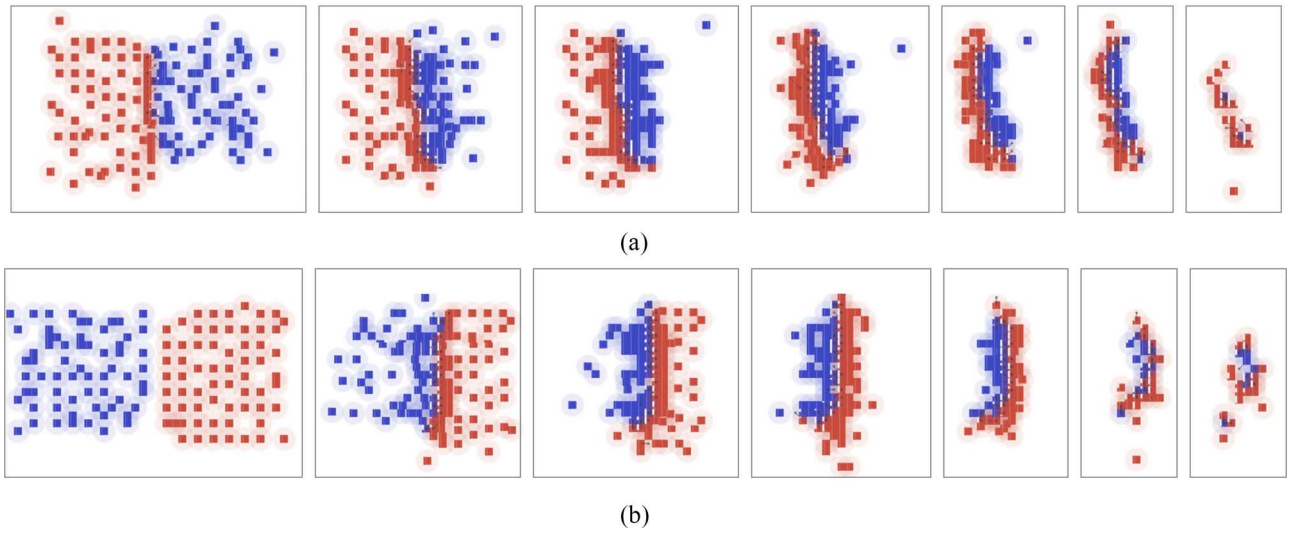

Fig. 9. Snapshots of two games. The red army is controlled by our proposed ACL algorithm, while another army is controlled by (a) MF-AC and (b) MF-Q, respectively. As can be shown from this figure, the army controlled by our proposed ACL algorithm can learn the cooperative strategies to besiege another army, thus having a large number of survivors.

VII. Conclusion

This article has proposed ACL in a mixed cooperative–competitive environment, which exploits the mean-field theory and friend-or-foe Q-learning. The proposed method first learns the mean action of friends and opponents through adversarial learning (AL), with friends cooperatively maximizing the reward while opponents minimizing it. Meanwhile, AL is also adopted to learn each agent's best response to the mean effects. Theoretical analysis of the convergence of the proposed algorithm to the Nash Q-value is provided. Experiments on the battle game and MPE have demonstrated the learning effectiveness and strength of our approach.

Our proposed ACL algorithm could address the challenges of the curse of dimensionality and nonstationary environments. The positive results in experiments indicate that it is possible for ACL to be applied in real-world applications, especially with many agents. However, in many large-scale MARL applications, agents could only interact directly with their neighbors (a small fraction of agents). Consequently, the actions and intentions are not consistent, which significantly impact the cooperative strategies. Communications between agents, or communications between the dominating agents potentially coordinate the inconsistent actions due to different view range. Besides, we will also extend ACL algorithms to the scenario with multiple coalitions, rather than only two coalitions, that is, friends and opponents. The strategies in such scenarios are more challenging since the interesting relationship is frail, that is, two agents may perform cooperatively or competitively under different situations.

Appendix
Proof of Convergence

We now prove the convergence of $\boldsymbol{Q}^t \triangleq [Q^t_{f_1}, \ldots, Q^t_{f_m}, Q^t_{o_1}, \ldots, Q^t_{o_k}]$ to the Nash Q-value $\boldsymbol{Q}^* \triangleq [Q^*_{f_1}, \ldots, Q^*_{f_m}, Q^*_{o_1}, \ldots, Q^*_{o_k}]$ when using the iteration of the ACL method. The proof is presented by showing that the operator $\mathcal{H}$, which is denoted as

$$\mathcal{H}\boldsymbol{Q}(s, \boldsymbol{a}_f, \boldsymbol{a}_o) = \boldsymbol{r}(s, \boldsymbol{a}_f, \boldsymbol{a}_o) + \gamma \sum_{s'} \mathcal{P}(s, \boldsymbol{a}_f, \boldsymbol{a}_o, s') \max_{\boldsymbol{a}'_f \in \mathcal{A}_1^{sp}} \min_{\boldsymbol{a}'_o \in \mathcal{A}_2^{sp}} \boldsymbol{Q}\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) \tag{22}$$

forms a contraction mapping with the fixed point, that is, $\|\mathcal{H}\boldsymbol{Q}^1 - \mathcal{H}\boldsymbol{Q}^2\|_\infty \le \gamma \|\boldsymbol{Q}^1 - \boldsymbol{Q}^2\|_\infty$.

Proof:

$$\begin{aligned}
&\left\|\mathcal{H}\boldsymbol{Q}^1 - \mathcal{H}\boldsymbol{Q}^2\right\|_\infty \\
&= \max_{s,\boldsymbol{a}} \left| \gamma \sum_{s'} \mathcal{P}(s, \boldsymbol{a}_f, \boldsymbol{a}_o, s') \right. \\
&\qquad \times \left[\max_{\boldsymbol{a}'_f \in \mathcal{A}_1^{sp}} \min_{\boldsymbol{a}'_o \in \mathcal{A}_2^{sp}} \boldsymbol{Q}^1\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) \right. \\
&\qquad \left. \left. - \max_{\boldsymbol{a}'_f \in \mathcal{A}_1^{sp}} \min_{\boldsymbol{a}'_o \in \mathcal{A}_2^{sp}} \boldsymbol{Q}^2\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) \right] \right| \\
&\le \max_{s,\boldsymbol{a}} \gamma \sum_{s'} \mathcal{P}(s, \boldsymbol{a}_f, \boldsymbol{a}_o, s') \\
&\times \left| \max_{\boldsymbol{a}'_f \in \mathcal{A}_1^{sp}} \min_{\boldsymbol{a}'_o \in \mathcal{A}_2^{sp}} \boldsymbol{Q}^1\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) \right. \\
&\qquad \left. - \max_{\boldsymbol{a}'_f \in \mathcal{A}_1^{sp}} \min_{\boldsymbol{a}'_o \in \mathcal{A}_2^{sp}} \boldsymbol{Q}^2\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) \right| \\
&\le \max_{s,\boldsymbol{a}} \gamma \sum_{s'} \mathcal{P}(s, \boldsymbol{a}_f, \boldsymbol{a}_o, s') \\
&\times \left| \max_{s^*, \boldsymbol{a}^*} \boldsymbol{Q}^1(s^*, \boldsymbol{a}^*) - \max_{s^*, \boldsymbol{a}^*} \boldsymbol{Q}^2(s^*, \boldsymbol{a}^*) \right| \\
&\le \max_{s,\boldsymbol{a}} \gamma \sum_{s'} \mathcal{P}(s, \boldsymbol{a}_f, \boldsymbol{a}_o, s') \left\|\boldsymbol{Q}^1 - \boldsymbol{Q}^2\right\|_\infty \\
&= \gamma \left\|\boldsymbol{Q}^1 - \boldsymbol{Q}^2\right\|_\infty .
\end{aligned} \tag{23}$$

The ACL learning algorithm determines the optimal Q-function using point samples. Let $\boldsymbol{\pi}$ be some random policy such that

$$P_{\boldsymbol{\pi}}[\boldsymbol{A}_t = \boldsymbol{a}|s_t = s] > 0 \tag{24}$$

for all state–action pairs $(s, \boldsymbol{a})$. Then, given any initial $\boldsymbol{Q}^0$, ACL uses the following update rule:

$$\begin{aligned}\boldsymbol{Q}^{t+1}(s_t, \boldsymbol{a}_f, \boldsymbol{a}_o) = &\big(1 - \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\big)\boldsymbol{Q}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \\ &+ \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \\ &\times \left(\boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f \in \boldsymbol{\mathcal{A}}_1^{sp}} \min_{\boldsymbol{a}'_o \in \boldsymbol{\mathcal{A}}_2^{sp}} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right)\right).\end{aligned} \tag{25}$$

This leads to the main theorem in Theorem 2.

Theorem 2: Given a finite-state stochastic game, the ACL learning algorithm converges to the optimal Q-value $\boldsymbol{Q} = [Q_1^*, \ldots, Q_N^*]$, given by the update rule

$$\begin{aligned}\boldsymbol{Q}^{t+1}(s, \boldsymbol{a}_f, \boldsymbol{a}_o) = &\big(1 - \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\big)\boldsymbol{Q}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \\ &+ \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \\ &\times \left(\boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f \in \boldsymbol{\mathcal{A}}_1^{sp}} \min_{\boldsymbol{a}'_o \in \boldsymbol{\mathcal{A}}_2^{sp}} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right)\right)\end{aligned} \tag{26}$$

as long as the following conditions are met.

Condition 1: All state–action pairs should be visited infinitely often, and the reward is bounded by some constant K.

Condition 2: $\sum_t \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) = \infty$ and $\sum_t(\alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o))^2 < \infty$ for all $(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \in \mathcal{S} \times \mathcal{A}$.

Condition 3: Agent's policy is greedy in the limit with infinite exploration (GLIE). In the case with the Boltzmann policy, the policy becomes greedy with respect to the Q-function in the limit as the temperature decays asymptotically to zero.

To establish Theorem 2, we need an auxiliary result from stochastic approximation which can be indicated as follows.

Theorem 3: A random iterative process $\Delta^{t+1}(x) = (1 - \alpha^t(x))\Delta^t(x) + \alpha^t(x)F^t(x)$ converges to zero with probability 1 under the following assumptions.

1) The state space is finite.
2) $0 \le \alpha^t \le 1$, $\sum_t \alpha^t(x) = \infty$, and $\sum_t(\alpha^t(x))^2 < \infty$.
3) $\|\mathbb{E}\{F^t(x)|\mathcal{F}^t\}\|_W \ge \gamma\|\Delta^t\|_W$, where $\gamma \in (0, 1)$.
4) $\boldsymbol{var}\{F^t(x)|\mathcal{F}^t\} \ge C(1 + \|\Delta^t\|_W)^2$, where C is some constant.

Here, $\mathcal{F}^t = \{\Delta^{t+1}, \Delta^t, \ldots, F^t, \ldots, \alpha^t\}$ stands for the past at step t; α^t, Δ^t, $F^t \in \mathcal{F}^t$ and $\|.\|_W$ is a weighted maximum norm.

Proof: See [53]. ∎

By substarcting $\boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)$ on both sides of (26) and letting $\boldsymbol{\Delta}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) = \boldsymbol{Q}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)$, we can have

$$\begin{aligned}\boldsymbol{\Delta}^{t+1}(s, \boldsymbol{a}_f, \boldsymbol{a}_o) = &\big(1 - \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\big)\boldsymbol{\Delta}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \\ &+ \alpha^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) \\ &\times \left[\boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f \in \boldsymbol{\mathcal{A}}_1^{sp}} \min_{\boldsymbol{a}'_o \in \boldsymbol{\mathcal{A}}_2^{sp}} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right)\right. \\ &\left. - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\right].\end{aligned} \tag{27}$$

From the comparison with the random iterative process in Theorem 2, we present the relation such that

$$\boldsymbol{F}^t(x, a, b) = \boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'} \min_{\boldsymbol{b}'} \boldsymbol{Q}^t(s', a', b') - \boldsymbol{Q}^*(s, a, b). \tag{28}$$

We need to prove that the operator $\mathcal{H}$ meets Theorem 2's third and fourth conditions. According to (28), we have that $\boldsymbol{F}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) = \boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f} \min_{\boldsymbol{a}'_o} \boldsymbol{Q}^t(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o) - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)$

$$\begin{aligned}\mathbb{E}\big[\boldsymbol{F}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)|\boldsymbol{\mathcal{F}}^t\big] &= \sum_{s'} \mathcal{P}(s, \boldsymbol{a}_f, \boldsymbol{a}_o, s') \\ &\quad \times \left[\boldsymbol{r}^t \gamma \max_{\boldsymbol{a}'_f} \min_{\boldsymbol{a}'_o} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right)\right. \\ &\quad \left. - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\right] \\ &= \mathcal{H}\boldsymbol{Q}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o).\end{aligned} \tag{29}$$

As the operator $\mathcal{H}$ forms a contraction mapping on the complete metric space with the fixed point $\boldsymbol{Q}^*$ being the Nash Q-value of the entire game, we have that $\mathcal{H}\boldsymbol{Q}^* = \boldsymbol{Q}^*$. For details of the proof of this equation, we refer readers to [54]. By using this fact, (29) can further be illustrated by $\mathbb{E}[\boldsymbol{F}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)|\boldsymbol{\mathcal{F}}^t] = \mathcal{H}\boldsymbol{Q}_t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) - \mathcal{H}\boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)$, and $\|\mathbb{E}[\boldsymbol{F}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)|\boldsymbol{\mathcal{F}}^t]\|_\infty \le \gamma\|\boldsymbol{Q}^t - \boldsymbol{Q}^*\|_\infty = \gamma\|\boldsymbol{\Delta}^t\|_\infty$.

Furthermore, it can be proved that the fourth condition is also met

$$\begin{aligned}&\mathbf{var}\big[\boldsymbol{F}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)|\boldsymbol{\mathcal{F}}^t\big] \\ &= \mathbb{E}\left[\left(\boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f} \min_{\boldsymbol{a}'_o} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\right.\right. \\ &\qquad \left.\left. - \big(\mathcal{H}\boldsymbol{Q}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o) - \boldsymbol{Q}^*(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\big)\right)^2\right] \\ &= \mathbb{E}\left[\left(\boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f} \min_{\boldsymbol{a}'_o} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right) - \mathcal{H}\boldsymbol{Q}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)\right)^2\right] \\ &= \mathbf{var}\left[\boldsymbol{r}^t + \gamma \max_{\boldsymbol{a}'_f} \min_{\boldsymbol{a}'_o} \boldsymbol{Q}^t\left(s', \boldsymbol{a}'_f, \boldsymbol{a}'_o\right)|\boldsymbol{\mathcal{F}}^t\right].\end{aligned} \tag{30}$$

Since $\boldsymbol{r}^t$ is bounded when employ Condition 1, it is clearly verifies $\mathbf{var}[\boldsymbol{F}^t(s, \boldsymbol{a}_f, \boldsymbol{a}_o)|\boldsymbol{\mathcal{F}}^t] \le C(1 + \|\boldsymbol{\Delta}_t\|_W^2)$ for some constant C. Finally, with all conditions met, it follows Theorem 2 that $\boldsymbol{\Delta}^t$ converges to zero with probability 1, that is to say, $\boldsymbol{Q}_t$ converges to $\boldsymbol{Q}_*$ with probability 1. ∎

REFERENCES

[1] F. L. Lewis and D. Liu, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*, vol. 17. Somerset, U.K.: Wiley, 2013.

[2] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 2, pp. 156–172, Mar. 2008.

[3] H. Zhang, C. Qin, and Y. Luo, "Neural-network-based constrained optimal control scheme for discrete-time switched nonlinear system using dual heuristic programming," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 839–849, Jul. 2014.

[4] C. Qin, H. Zhang, Y. Wang, and Y. Luo, "Neural network-based online H_∞ control for discrete-time affine nonlinear system using adaptive dynamic programming," *Neurocomputing*, vol. 198, pp. 91–99, Jul. 2016.

[5] H. Zhang, Q. Wei, and Y. Luo, "A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 4, pp. 937–942, Aug. 2008.

[6] C. Qin, H. Zhang, and Y. Luo, "Model-free H_∞ control design for unknown continuous-time linear system using adaptive dynamic programming," *Asian J. Control*, vol. 18, no. 2, pp. 609–618, 2016.

[7] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[8] F. L. Da Silva, R. Glatt, and A. H. R. Costa, "MOO-MDP: An object-oriented representation for cooperative multiagent reinforcement learning," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 567–579, Feb. 2019.

[9] Z. Zhang, Y.-S. Ong, D. Wang, and B. Xue, "A collaborative multiagent reinforcement learning method based on policy gradient potential," *IEEE Trans. Cybern.*, early access, Aug. 21, 2019, doi: 10.1109/TCYB.2019.2932203.

[10] H. Zhang, J. Zhang, G.-H. Yang, and Y. Luo, "Leader-based optimal coordination control for the consensus problem of multiagent differential games via fuzzy adaptive dynamic programming," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 1, pp. 152–163, Feb. 2015.

[11] J. Zhang, Z. Wang, and H. Zhang, "Data-based optimal control of multiagent systems: A reinforcement learning design approach," *IEEE Trans. Cybern.*, vol. 49, no. 12, pp. 4441–4449, Dec. 2019.

[12] E. A. O. Diallo and T. Sugawara, "Learning strategic group formation for coordinated behavior in adversarial multi-agent with double DQN," in *Proc. Int. Conf. Principles Pract. Multi-Agent Syst.*, 2018, pp. 458–466.

[13] H. A. Al-Rawi, M. A. Ng, and K.-L. A. Yau, "Application of reinforcement learning to routing in distributed wireless networks: A review," *Artif. Intell. Rev.*, vol. 43, no. 3, pp. 381–416, 2015.

[14] M. Rahimi, S. Gibb, Y. Shen, and H. M. La, "A comparison of various approaches to reinforcement learning algorithms for multi-robot box pushing," in *Proc. Int. Conf. Eng. Res. Appl.*, 2018, pp. 16–30.

[15] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *Proc. Learn. Inference Control Multi-Agent Syst. (NIPS)*, 2016. [Online]. Available: https://www.fransoliehoek.net/publications/htmlfiles/b2hd-VanDerPol16LICMAS.html

[16] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," 2016. [Online]. Available: arXiv:1610.03295.

[17] K. Corder, M. M. Vindiola, and K. Decker, "Decentralized multi-agent actor-critic with generative inference," 2019. [Online]. Available: arXiv:1910.03058.

[18] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4295–4304.

[19] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2974–2982.

[20] S. V. Albrecht and P. Stone, "Autonomous agents modelling other agents: A comprehensive survey and open problems," *Artif. Intell.*, vol. 258, pp. 66–95, Feb. 2018.

[21] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate to solve riddles with deep distributed recurrent Q-networks," 2016. [Online]. Available: arXiv:1602.02672.

[22] Z. Zhang, D. Zhao, J. Gao, D. Wang, and Y. Dai, "FMRQ—A multiagent reinforcement learning algorithm for fully cooperative tasks," *IEEE Trans. Cybern.*, vol. 47, no. 6, pp. 1367–1379, Jun. 2017.

[23] J. Foerster *et al.*, "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1146–1155.

[24] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2017, pp. 66–83.

[25] M. Lauer and M. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 535–542.

[26] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 2681–2690.

[27] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," *PLoS ONE*, vol. 12, no. 4, 2017, Art. no. e0172395.

[28] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Assoc., Inc., 2017, pp. 6379–6390.

[29] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 2817–2826.

[30] M. L. Littman, "Friend-or-foe Q-learning in general-sum games," in *Proc. Int. Conf. Mach. Learn.*, vol. 1, 2001, pp. 322–328.

[31] G. Papoudakis, F. Christianos, A. Rahman, and S. V. Albrecht, "Dealing with non-stationarity in multi-agent deep reinforcement learning," 2019. [Online]. Available: arXiv:1906.04737.

[32] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, 2014, pp. 387–395.

[33] S. Li, Y. Wu, X. Cui, H. Dong, F. Fang, and S. Russell, "Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 4213–4220.

[34] H. He, J. Boyd-Graber, K. Kwok, and H. Daumé, III, "Opponent modeling in deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1804–1813.

[35] R. Raileanu, E. Denton, A. Szlam, and R. Fergus, "Modeling others using oneself in multi-agent reinforcement learning," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 4257–4266.

[36] J. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, and I. Mordatch, "Learning with opponent-learning awareness," in *Proc. 17th Int. Conf. Auton. Agents MultiAgent Syst.*, 2018, pp. 122–130.

[37] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," 2018. [Online]. Available: arXiv:1812.09755.

[38] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1495–1502.

[39] D. Mguni, J. Jennings, and E. M. de Cote, "Decentralised learning in systems with many, many strategic agents," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 4686–4693.

[40] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 5571–5580.

[41] S. Hu, C.-W. Leung, and H.-F. Leung, "Modelling the dynamics of multiagent q-learning in repeated symmetric games: a mean field theoretic approach," in *Advances in Neural Information Processing Systems 32*. New York, NY, USA: Curran Assoc., Inc., 2019, pp. 12102–12112.

[42] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proc. Mach. Learn.*, 1994, pp. 157–163.

[43] J. Hu and M. P. Wellman, "Multiagent reinforcement learning: Theoretical framework and an algorithm," in *Proc. Int. Conf. Mach. Learn.*, vol. 98, 1998, pp. 242–250.

[44] M. Bowling and M. Veloso, "Multiagent learning using a variable learning rate," *Artif. Intell.*, vol. 136, no. 2, pp. 215–250, 2002.

[45] M. Bowling, "Convergence problems of general-sum multiagent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2000, pp. 89–94.

[46] L. P. Kadanoff, "More is the same; phase transitions and mean field theories," *J. Stat. Phys.*, vol. 137, nos. 5–6, p. 777, 2009.

[47] L. Ying, "On the approximation error of mean-field models," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 44, no. 1, pp. 285–297, 2016.

[48] N. Gast, "Expected values estimated via mean-field approximation are 1/N-accurate," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 1, pp. 1–26, 2017.

[49] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015. [Online]. Available: arXiv:1509.02971.

[50] L. Zheng *et al.*, "MAgent: A many-agent reinforcement learning platform for artificial collective intelligence," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 8222–8223.

[51] M. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," in *Proc. AAAI Fall Symp. Series*, 2015, pp. 29–37.

[52] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[53] T. Jaakkola, M. I. Jordan, and S. P. Singh, "Convergence of stochastic iterative dynamic programming algorithms," in *Advances in Neural Information Processing Systems*. San Mateo, CA, USA: Morgan Kaufmann, 1994, pp. 703–710.

[54] J. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games," *J. Mach. Learn. Res.*, vol. 4, pp. 1039–1069, Nov. 2003.

Guiyang Luo received the Ph.D. degree in computer science and technology from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2020.

He is currently a Postdoctoral Fellow with the State Key Laboratory of Networking and Switching Technology, BUPT. His current research interests include multiagent systems and intelligent transportation systems.

Hui Zhang received the Ph.D. degree from the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China, and the University of Chinese Academy of Sciences, Beijing, in 2020.

From 2018 to 2019, she was a Visiting Scholar with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA. She is currently a Senior Researcher with Tencent, Beijing. Her research interests include computer vision, text recognition, and intelligent transportation systems.

Haibo He (Fellow, IEEE) received the B.S. and M.S. degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical engineering from Ohio University, Athens, OH, USA, in 2006.

He is currently the Robert Haas Endowed Chair Professor with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA.

Dr. He received the IEEE International Conference on Communications Best Paper Award in 2014, the IEEE CIS Outstanding Early Career Award in 2014, and the National Science Foundation CAREER Award in 2011. He is currently an Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. He was the General Chair of the IEEE Symposium Series on Computational Intelligence in 2014.

Jinglin Li (Member, IEEE) received the Ph.D. degree in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, China, in 2004.

He is currently a Professor of Computer Science and Technology and the Director of the Switching and Intelligent Control Research Center, State Key Laboratory of Networking and Switching Technology, Beijing. His research interests are mainly in the areas of mobile Internet, Internet of Things, Internet of Vehicles, convergence network, and service technologies.

Fei-Yue Wang (Fellow, IEEE) received the Ph.D. degree in computer and systems engineering from the Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990.

He is currently the Director of the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His current research focuses on methods and applications for parallel systems, social computing, and knowledge automation.

Dr. Wang received the 2nd Class National Prize in Natural Sciences of China in 2007, and was awarded the Outstanding Scientist by ACM for his work in intelligent control and social computing. He received the IEEE ITS Outstanding Application and Research Awards in 2009 and 2011, and the IEEE SMC Norbert Wiener Award in 2014. He was the Founding Editor-in-Chief of the *International Journal of Intelligent Control and Systems* from 1995 to 2000 and *IEEE Intelligent Transportation Systems Magazine* from 2006 to 2007, and an Editor-in-Chief of IEEE INTELLIGENT SYSTEMS from 2009 to 2012 and IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS from 2009 to 2016. He is currently an Editor-in-Chief of China's *Journal of Command and Control*. Since 1997, he has been serving as the General or Program Chair of more than 20 IEEE, INFORMS, ACM, and ASME conferences. He was the President of the IEEE ITS Society from 2005 to 2007, Chinese Association for Science and Technology, USA, in 2005, and the American Zhu Kezhen Education Foundation from 2007 to 2008, and the Vice President of the ACM China Council from 2010 to 2011. Since 2008, he has been the Vice President and the Secretary General of the Chinese Association of Automation. He is an elected Fellow of INCOSE, IFAC, ASME, and AAAS.