Neurocomputing 517 (2023) 165–172

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Multi-agent actor-critic with time dynamical opponent model

Yuan Tian [a], Klaus-Rudolf Kladny [a], Qin Wang [a], Zhiwu Huang [c], Olga Fink [b,*]

[a] ETH Zürich, Switzerland
[b] EPFL, Switzerland
[c] Singapore Management University, Singapore

ARTICLE INFO

Article history:
Received 23 March 2022
Revised 28 September 2022
Accepted 18 October 2022
Available online 31 October 2022
Communicated by Zidong Wang

Keyword:
Reinforcement learning
Multi-agent reinforcement learning
Multi-agent systems
Opponent modeling
Non-stationarity

ABSTRACT

In multi-agent reinforcement learning, multiple agents learn simultaneously while interacting with a common environment and each other. Since the agents adapt their policies during learning, not only the behavior of a single agent becomes non-stationary, but also the environment as perceived by the agent. This renders it particularly challenging to perform policy improvement. In this paper, we propose to exploit the fact that the agents seek to improve their expected cumulative reward and introduce a novel *Time Dynamical Opponent Model* (TDOM) to encode the knowledge that the opponent policies tend to improve over time. We motivate TDOM theoretically by deriving a lower bound of the log objective of an individual agent and further propose *Multi-Agent Actor-Critic with Time Dynamical Opponent Model* (TDOM-AC). We evaluate the proposed TDOM-AC on a differential game and the Multi-agent Particle Environment. We show empirically that TDOM achieves superior opponent behavior prediction during test time. The proposed TDOM-AC methodology outperforms state-of-the-art Actor-Critic methods on the performed tasks in cooperative and **especially** in mixed cooperative-competitive environments. TDOM-AC results in a more stable training and a faster convergence. Our code is available at https://github.com/Yuantian013/TDOM-AC.

1. Introduction

Multi-agent systems have recently found applications in many different domains, including traffic control [1], games [2–4], consensus tracking control [5,6] and swarm control [7]. The complexity of the tasks in these applications often precludes the usage of predefined agent behaviors and stipulates the agents to learn a policy, and to define the problem as multi-agent reinforcement learning (MARL). In such cases, multiple agents learn simultaneously while interacting with a common environment. Since the agents adapt their policies during learning, not only the behavior of a single agent becomes non-stationary, but also the environment as perceived by the agents [8]. Since most of the conventional Reinforcement Learning (RL) approaches assume stationary system dynamics [9], they usually perform poorly when required to interact with multiple adaptive agents in a shared environment [10,8].

A common approach in MARL is to explicitly consider the presence of opponents by modeling their policies using an opponent model [11,12] (In the following, the word "opponents" refers to other agents in an environment irrespective of the environment's cooperative or adversarial nature). An accurate opponent model can provide informative cues to future behaviors of the opponents. However, such a precise prediction is challenging as the opponents' policies are changing over time [12].

In our novel approach, entitled *Time Dynamical Opponent Model* (TDOM), we aim to address the challenge of non-stationarity of the agent's behavior by modeling the opponent policy parameters as a dynamical system which are generally used to model the evolution of systems in time [13]. Here, we build the system dynamics on the prior knowledge that all agents are concurrently trying to improve their policies with respect to their individual cumulative reward. It is worth mentioning that TDOM is highly general and can further support all kinds of opponent objectives, i.e. cooperative, competitive or mixed settings.

By deriving a lower bound on the log-objective of an individual agent, we further propose a Multi-agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC) for mixed cooperative-competitive tasks. The proposed TDOM-AC framework comprises a *Centralized Training and Decentralized Execution* (CTDE), see Fig. 1. In this framework, centralized critics provide additional information to guide the training [14,10]. However, this information is not used at execution time. Each agent only has access to

* Corresponding author.
E-mail address: olga.fink@epfl.ch (O. Fink).

https://doi.org/10.1016/j.neucom.2022.10.045
0925-2312/© 2022 The Author(s). Published by Elsevier B.V.

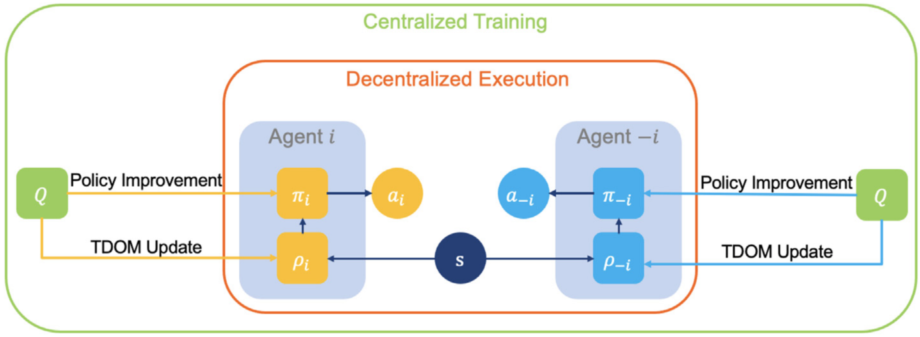


Fig. 1. An overview of the proposed framework, where i indicates one of the agents and $-i$ refers to the other agents. In the proposed framework, the decision making process is fully decentralized. Each agent only observes the state s, and then infers the other opponents' behaviors $\hat{a}_{-i}$ via its own opponent model $\phi_i(s)$. Based on the state and the predicted opponent behaviors, the agent selects the action a_i via its policy $\pi_i(s, \hat{a}_{-i})$.

the state information and can only select an action based on its own prediction of other opponents' actions.

We evaluate the proposed TDOM-AC on a Differential Game and a Multi-agent Particle Environment and compare the performance to two state-of-the-art actor-critic algorithms, namely *Regularized Opponent Model with Maximum Entropy Objective* (ROMMEO) [12] and *Probabilistic Recursive Reasoning* (PR2) [15]. We demonstrate empirically that the proposed TDOM algorithm achieves superior opponent behavior prediction during execution time. The proposed TDOM-AC outperforms the considered baselines on the performed tasks and considered measures. TDOM-AC results in a more stable training, faster convergence and **especially a superior performance in mixed cooperative-competitive environments**.

The remainder of this paper is organized as follows: Section 2 provides a brief overview of the related works of this study. Section 3 and 4 introduces the proposed opponent model and TDOM-AC. Section 5 interprets and compares the results of the simulations. In Section 6, the conclusion and future work are presented.

2. Related work

Multi-Agent systems (MAS) encompass decision-making of multiple agents interacting in a shared environment [16]. For complex tasks where using predefined agent behaviors is not possible, MARL enables the agent to learn from the interaction with the environment [17]. One of the main challenges in MARL is the inherent non-stationarity. To address this challenge, one direction has been to account for the behaviors of other agents through a centralized critic by adopting the CTDE framework [18,19]. For value-based approaches in the CTDE framework, methods usually rely on restrictive structural constraints or network architectures, such as QDPP [20], QMIX [21], FOP [22], QTRAN [23], and VDN [24]. For actor-critic based methods, these approaches usually include an additional policy with supplementary opponent models that can reason about other agents' believes [15], private information [25], behavior [10], strategy [26] and other characteristics. With the supplementary opponent models, these works can also be linked to the field of opponent modeling (OM) [27,11].

There are several ways to model the behavior of opponents. One of them is to factorize the joint policy $\pi(a^{-i}, \mathbf{a}^{-i}|s)$ in different ways. This has been done in previous works [11,12,15]. Also, different objective functions for the opponent model have been implemented. *Multi-agent Deep Deterministic Policy Gradient* (MADDPG) [10] approximates opponents' policy by maximizing the log probability of other agents' actions with an entropy regularizer; PR2 [15] considers an optimization-based approximation to infer the unobservable opponent policy via variational inference [28] and ROMMEO adopts the regularized opponent model with maximum entropy objective, which can be interpreted as a combination of MADDPG and PR2. However, the existing approaches either suffer from high computational cost due to the recursive reasoning policy gradient [15], or are limited to specific types of environments [12]. In this work, we propose an alternative opponent model motivated by a temporal improvement assumption to overcome these limitations.

An earlier approach that explicitly addresses opponent-learning awareness is *Learning with Opponent Learning Awareness* (LOLA) [29]. When performing the policy update, any agent optimises its return under a one-step-look-ahead of the opponent learning. However, it is limited by strong assumptions. Specifically, these subsume access to both exact gradients and Hessians of the value function. Furthermore, a specific network design is required. Although the authors have subsequently proposed a variant of their approach, the *policy gradient-based naive learner* (NL-PG) with fewer assumptions, the intrinsic on-policy design inherently suffers from data inefficiency. Also, LOLA only supports two-agent systems, while we are considering approaches that allow for arbitrarily many agents.

3. Method

3.1. Assumptions

In this work, we aim to tackle the mentioned limitations outlined in Section 2. For fair comparison, we adopt the same observability assumptions from previous work [15,12,10]. Since in cooperative games all the agents receive the same reward and in zero-sum games the opponents' rewards can easily be inferred from ones own reward, we assume all agents can access each other's rewards, just like in LOLA [29]. In contrast to vanilla LOLA, we do not make the assumption of the observability of opponent policies.

3.2. Markov game

An N-agents Markov game [30], also referred to as N-agents stochastic game [31], is defined by a tuple $(\mathcal{S}, \mathcal{A}^1 \ldots \mathcal{A}^n, r^1 \ldots r^n, p, \mathcal{T}, \gamma)$, where $\mathcal{S}$ is the state space, and $\mathcal{A}^i$ is the action space. At at time step t, agent i chooses its action $a^i_t \in \mathcal{A}^i$ according to the policy conditioning on the observed state $s_t \in \mathcal{S}$. And $r^i_t \in \mathbb{R}$ is the corresponding rewards assigned to agent i, which is obtained from the pre-defined reward function $r^i_t = r^i_t(s_t, a^i_t, \mathbf{a}^{-i}_t)$, where the $\mathbf{a}^{-i}_t$ refers to the set of opponent actions. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the state transition function, p is the initial state distribution and γ is the discount factor. At each time step t, actions are taken simultaneously by all agents. Each agent aims to maximize its own expected discounted sum of rewards. Thus, for each individual agent i, the objective for its policy π_i can be expressed as:

$$J(\pi_i) = \max_{\pi_i} \sum_{t=0}^{\infty} \mathbb{E}\left[\gamma^t r^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right)\right] \tag{1}$$

We note that since multiple adaptive agents interact in a shared environment, each agent's rewards and the environment transitions depend also on the actions of the opponents [8]. Thus, the unobservable dynamic policies of the opponents induce non-stationarity in the environment dynamics from the perspective of a single agent. To address this challenge, we propose to consider the agent policy parameters as a dynamical system in which we encode the prior knowledge that all agents are concurrently trying to improve their policies.

3.3. Time dynamical opponent model

To introduce our methodology, we begin by deriving a lower bound for the maximization objective in Eq. 1, in which we omit some of the parameterization notation for less cluttering:

$$\max_{\pi^i,\, \rho^i} \mathbb{E}_{a_t^i \sim \pi^i\left(\cdot|\hat{\mathbf{a}}_t^{-i}\right),\, \hat{\mathbf{a}}_t^{-i} \sim \rho^i,\, \mathbf{a}_t^{-i} \sim \tilde{\pi}^{-i}} \left[Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right)\right], \tag{2}$$

where for lighter notation we omit the $s_t \sim d^{\pi}$, which means sampling a state from the discounted state visitation distribution d^{π} using current policies $\pi := \{\pi^j\}_j$, where $\pi^j(\cdot|\mathbf{a}^{-j}, s)$. $\rho^i(\cdot|s)$ refers to the belief of agent i about opponents $-i$, also known as *opponent model*. Furthermore, we define $\tilde{\pi}^j(\cdot|s)$ to be

$$\tilde{\pi}^j\left(\mathbf{a}^j|s\right) := \int_{\mathcal{A}^{-j}} \pi^j\left(a^j|\mathbf{a}^{-j}, s\right) \rho^j\left(\mathbf{a}^{-j}|s\right) d\mathbf{a}^{-j}, \tag{3}$$

which can be interpreted as the marginal policy of agent j. Then we can formulate the marginal opponent policies to be $\tilde{\pi}^{-i} := \{\tilde{\pi}^j\}_{j\in -i}$

The presented maximization objective means that agent i aims to maximize its Q-function given that all agents play their current policies $\tilde{\pi}_t^{-i}$ which are unknown to agent i.

We can now derive a lower bound of the log objective of agent i:

$$\begin{aligned}
&\log \mathbb{E}_{a_t^i \sim \pi_t^i\left(\cdot|\hat{a}_t^{-i}\right),\, \hat{\mathbf{a}}_t^{-i} \sim \rho_t^i,\, \mathbf{a}_t^{-i} \sim \tilde{\pi}_t^{-i}} \left[Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right)\right] \\
&= \log \int_{\mathcal{A}^i} \int_{\mathcal{A}^{-i}} \int_{\mathcal{A}^{-i}} Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right) \tilde{\pi}_t^{-i}\left(\mathbf{a}_t^{-i}|s_t\right) \rho_t^i\left(\hat{\mathbf{a}}_t^{-i}|s_t\right) \pi_t^i\left(a_t^i|\hat{\mathbf{a}}_t^{-i}, s_t\right) \\
&\quad d\hat{\mathbf{a}}_t^{-i}\, d\mathbf{a}_t^{-i}\, da_t^i \\
&= \log \int_{\mathcal{A}^i} \int_{\mathcal{A}^{-i}} \int_{\mathcal{A}^{-i}} Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right) \frac{\tilde{\pi}_t^{-i}\left(\mathbf{a}_t^{-i}|s_t\right)}{\rho_t^i\left(\mathbf{a}_t^{-i}|s_t\right)} \rho_t^i\left(\mathbf{a}_t^{-i}|s_t\right) \rho_t^i\left(\hat{\mathbf{a}}_t^{-i}|s_t\right) \\
&\quad \pi_t^i\left(a_t^i|\hat{\mathbf{a}}_t^{-i}, s_t\right) d\hat{\mathbf{a}}_t^{-i}\, d\mathbf{a}_t^{-i}\, da_t^i \\
&\geqslant \mathbb{E}_{a_t^i \sim \pi_t^i\left(\cdot|\hat{\mathbf{a}}_t^{-i}\right),\, \hat{\mathbf{a}}_t^{-i} \sim \rho_t^i,\, \mathbf{a}_t^{-i} \sim \rho_t^i} \left[\log Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right) + \log\left(\frac{\tilde{\pi}^{-i}\left(\mathbf{a}_t^{-i}|s_t\right)}{\rho_t^i\left(\mathbf{a}_t^{-i}|s_t\right)}\right)\right] \\
&= \mathbb{E}_{a_t^i \sim \tilde{\pi}_t^i,\, \mathbf{a}_t^{-i} \sim \rho_t^i} \left[\log Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right)\right] - \mathrm{KL}\left(\rho_t^i(\cdot|s_t)\, ||\, \tilde{\pi}_t^{-i}(\cdot|s_t)\right).
\end{aligned} \tag{4}$$

If we furthermore make the assumption that:

$$Q^{\mathrm{opt}} = \max_{a^i} Q^i\left(s_t, a^i, \mathbf{a}^{-i}\right) \quad \forall \mathbf{a}^{-i} \in \mathcal{A}^{-i}, \tag{5}$$

for some fixed Q^{opt}, we see that we can maximize this lower bound by minimizing the Kullback–Leibler Divergence $\mathrm{KL}\left(\rho_t^i(\cdot|s)\, ||\, \tilde{\pi}_t^{-i}(\cdot|s_t)\right)$ w.r.t. ρ_t^i and then maximizing the Q-function w.r.t. π_t^i:

$$\max_{\pi_t^i} \mathbb{E}_{a_t^i \sim \pi_t^i\left(\cdot|\mathbf{a}_t^{-i}\right),\, \mathbf{a}_t^{-i} \sim \rho_t^i} \left[Q^i\left(s_t, a_t^i, \mathbf{a}_t^{-i}\right)\right]. \tag{6}$$

However, the method proposed above has an obvious issue: How can we minimize $\mathrm{KL}\left(\rho_t^i(\cdot|s)\, ||\, \tilde{\pi}_t^{-i}(\cdot|s_t)\right)$ if $\tilde{\pi}_t^{-i}$ is not available to agent i?

In order to address this question, we propose to utilize prior information about the opponents' learning process. Using this information would enable to better model their non-stationary behavior. Specifically, there exists one aspect that to the best of our knowledge has not been considered before in opponent modelling: Over time, each agent j is expected to improve its policy using policy network parameters θ_t^j [1] in order to maximize its expected cumulative reward under the given system dynamics and opponent policies. This can be expressed as an ordinary differential equation (ODE):

$$\begin{aligned}
\frac{d}{dt}\, \theta_t^j \quad &\approx \nabla_{\theta^j} \mathbb{E}_{\pi_{\theta_t}^j,\, \pi_t^{-j}} \left[\sum_{t=0}^{\infty} \gamma^t\, r_t^j\left(a_t^j,\, \mathbf{a}_t^{-j},\, s_t\right)\right] \\
&= \nabla_{\theta^j} \mathbb{E}_{\pi_\theta^j,\, \pi_t^{-j}} \left[Q^j\left(a_t^j,\, \mathbf{a}_t^{-j},\, s_t\right)\right],
\end{aligned} \tag{7}$$

where $\pi^{-j} := \{\pi^k\}_{k\in -j}$.

We propose to encode this knowledge in the opponent model design. We would like to make explicit here that unlike in policy improvement, the opponent model is designed to simulate the policy optimization process for all opponents instead of maximizing their expected Q-value. It is worth to point out that unlike LOLA [29] which considers the opponent's policy update to optimize the agent's policy, our agent takes the opponents' policy improvement assumption into account to optimize its opponent model instead of the policy directly.

In order to minimize $\mathrm{KL}\left(\rho^i(\cdot|s)\, ||\, \tilde{\pi}^{-i}(\cdot|s_t)\right)$, we exploit the temporal improvement assumption for discrete time dynamics, parameterized by θ^{-i}:

$$\theta_t^{-i} \approx \theta_{t-1}^{-i} + \eta \nabla_{\theta^{-i}} \mathbb{E}_{\tilde{\pi}_{t-1}^i,\, \tilde{\pi}_\theta^{-i}} \left[Q^{-i}\left(a^i,\, \mathbf{a}^{-i},\, s\right)\right], \tag{8}$$

for some $\eta > 0$. However, the opponent model cannot be updated like this since neither $\tilde{\pi}_{\theta_t}^{-i}$ nor θ_t^{-i} are directly available to agent i. Hence, we take our best approximation $\rho_{\psi_t}^i$ which is our opponent model which is parameterized by ψ_t^i and update as

$$\psi_t^i \leftarrow \psi_{t-1}^i + \eta \nabla_{\psi^i} \mathbb{E}_{a^i \sim \tilde{\pi}_{t-1}^i,\, \mathbf{a}^{-i} \sim \rho_{\psi_t^i}} \left[Q^{-i}\left(a^i,\, \mathbf{a}^{-i},\, s\right)\right]. \tag{9}$$

We point out that the Q mentioned above can represent any type of critic function, such as Q-function, soft Q-function or advantage function.

To summarize, firstly, we derive a learning objective for agent i's policy π^i. We show that a good opponent model can alleviate the non-stationarity problem of policy updates in MARL. With an accurate opponent prediction, each agent can access a more reliable Q estimation, which provides better guidance for its own policy update and further allows the agent to become a better collaborator or stronger adversary to influence other agents in cooperative and competitive settings, respectively. Secondly, we propose a novel approach that exploits the temporal improvement assumption to guide the opponent model evolution. Since each agent dose not have access to the opponent's policy in the general MARL setup, it is not feasible to explicitly minimize the KL-Divergence between the opponent model and the underlying opponent policy. However, evolving the opponent model by simulating the policy optimization process can implicity minimize the corresponding KL-Divergence. We experimentally demonstrate the effectiveness and accuracy of the proposed TDOM in the Section 5.

[1] When parameterizing a function for agent j, we will always write e.g. π_θ^j instead of $\pi_{\theta^j}^j$.

4. Multi-agent actor-critic with time dynamical opponent model (TDOM-AC)

With the proposed TDOM, we introduce Multi-Agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC). TDOM-AC follows the CTDE framework [14,10]. There are three main modules in the proposed TDOM-AC: Centralized Q-function $Q(s, a^i, \mathbf{a}^{-i})$, opponent model $\rho(\cdot|s)$ and policy $\pi(\cdot|s, \hat{\mathbf{a}}^{-i})$. We further use neural networks (NNs) as function approximators, particularly applicable in high-dimensional and/or continuous multi-agent tasks. For an individual agent, i, the three modules are parameterized by ϕ^i, θ^i and ψ^i, respectively. The functions are updated using stochastic gradient based optimization with learning rates $\eta_\cdot$:

$$\begin{aligned} \phi^i_{t+1} &\leftarrow \phi^i_t + \eta_\phi \hat{\nabla}_{\phi^i} J\left(\phi^i_t\right) \\ \theta^i_{t+1} &\leftarrow \theta^i_t + \eta_\theta \hat{\nabla}_{\theta^i} J\left(\theta^i_t\right) \end{aligned} \tag{10}$$

and as elucidated in Section 3.3,

$$\psi^i_{t+1} \leftarrow \psi^i_t + \eta_\psi \hat{\nabla}_{\psi^i} J\left(\psi^i_t\right). \tag{11}$$

We would like to clarify that although Eqs. 10 and 11 perform similar operations, their underlying idea is different. We can interpret Eq. 10 as an approximation of a policy improvement and evaluation step without running it until convergence. However, Eq. 11 does not follow this idea. Instead, this update is based on the temporal improvement assumption with the underlying goal of minimizing the Kullback–Leibler divergence to the true marginal opponent policies $\tilde{\pi}^{-i}$ instead of policy improvement.

In the proposed TDOM-AC, experience replay buffer D is used [32], where the off-policy experiences of all agents are recorded. In a scenario with N agents, at time step t, a tuple $[s_t, s_{t+1}, a^1_t, \ldots, a^N_t, r^1_t, \ldots, r^N_t]$ is recorded.

We adopt the maximum entropy reinforcement learning (MERL) framework [33] to enable a richer exploration and a better learning stability. It is easy to see that the derivation still holds. We merely omit the adjustments in the previous sections for the purpose of readability. The centralized soft Q-function parameters can be trained to minimize the soft Bellman residual:

$$J\left(\phi^i\right) = \mathbb{E}_{(s_t, a_t, \mathbf{a}^{-i}_t, s_{t+1}) \sim D} \frac{1}{2}\left[Q^i_\phi(s_t, a_t, \mathbf{a}^{-i}_t) - \left(r^i_t + \gamma V(s_{t+1})\right)\right]^2, \tag{12}$$

where the value function V is implicitly parameterized by the soft Q-function [33] parameters. The objective function becomes:

$$\begin{aligned} J\left(\phi^i\right) = -\mathbb{E}_{(s_t, a_t, s_{t+1}) \sim D,\, \hat{\mathbf{a}}^{-i}_{t+1} \sim \rho^i_\psi,\, \hat{a}^i_{t+1} \sim \pi^i_\theta} \Big[\Big(& Q^i_\phi(s_t, a^i_t, \mathbf{a}^{-i}_t) - \big(r^i(s_t, a^i_t, \mathbf{a}^{-i}_t) \\ & + \gamma \big(Q^i_{\bar{\phi}}(s_{t+1}, \hat{a}^i_{t+1}, \hat{\mathbf{a}}^{-i}_{t+1}) - \alpha \log \pi^i(\hat{a}^i_{t+1}|s_{t+1}, \hat{\mathbf{a}}^{-i}_{t+1}) \\ & - \alpha \log \rho^i_\psi(\hat{\mathbf{a}}^{-i}_{t+1}|s_{t+1}) \big) \big) \Big)^2 \Big]. \end{aligned} \tag{13}$$

The $Q^i_{\bar{\phi}}$ is the target soft Q-network that has the same structure as Q^i and is parameterized by $\overline{\phi^i}$, but updated through exponentially moving average of the soft Q-function weights [32].

According to the MERL objective, the TDOM-based policy is learned by directly minimizing the expected KL-divergence between normalized centralized soft Q-function:

$$J\left(\theta^i\right) = \mathbb{E}_{s \sim D,\, \hat{\mathbf{a}}^{-i}_{t+1} \sim \rho^i_\psi}\left[Q^i_\phi(s, a^i, \hat{\mathbf{a}}^{-i}) - \alpha \log \pi^i_\theta(a^i|s, \hat{\mathbf{a}}^{-i})\right], \tag{14}$$

where α is the temperature parameter that determines the relative importance of the entropy term versus the reward, thus controls the stochasticity of the optimal policy. In order to achieve a low variance estimator of $J\left(\theta^i\right)$, we apply the reparameterization trick [34] for modeling the policy:

$$a^i_t = f^i_\theta(\epsilon; s, \mathbf{a}^{-i}), \tag{15}$$

where ϵ_t is a noise vector that is sampled from a fixed distribution. A common choice is a Gaussian distribution $\mathcal{N}$. We can now rewrite the objective in Eq. 14 as

$$\begin{aligned} J\left(\theta^i\right) = \; & \mathbb{E}_{s \sim D,\, \hat{\mathbf{a}}^{-i}_{t+1} \sim \rho^i_\psi,\, \epsilon \sim \mathcal{N}} \\ & \times \left[Q^i\left(s, f^i_\theta(\epsilon; s, \mathbf{a}^{-i}), \mathbf{a}^{-i}\right) - \alpha \log \pi^i_\theta\left(f^i_\theta(\epsilon; s, \mathbf{a}^{-i})|s, \mathbf{a}^{-i}\right)\right]. \end{aligned} \tag{16}$$

Let $\mathbf{Q}^{-i}(s, a^i, \hat{\mathbf{a}}^{-i}) := \sum_{j \in -i} Q^j_\phi(s, a^i, \hat{\mathbf{a}}^{-i})$. Then, according to Eq. 9, the objective for the TDOM model can be written as

$$J\left(\psi^i\right) = \mathbb{E}_{s \sim D,\, \hat{\mathbf{a}}^{-i} \sim \rho^i_\psi,\, a^i \sim \pi^i_\theta}\left[\mathbf{Q}^{-i}(s, a^i, \hat{\mathbf{a}}^{-i}) - \alpha \log \rho^i_\psi(\hat{\mathbf{a}}^{-i}|s)\right]. \tag{17}$$

However, in mixed cooperative-competitive environments, agents may have conflicting interests which can *neutralize* the gradient in this formulation. We illustrate this by an example of a two-player zero-sum Markov game:

$$r^1(s, a^1, a^2) = -r^2(s, a^1, a^2), \qquad \forall s \in \mathcal{S},\ (a^1, a^2) \in \mathcal{A}^2. \tag{18}$$

Assumption 1. The Q-function approximations Q^1_ϕ and Q^2_ϕ for agent 1 and agent 2 respectively, have converged to their true functions $Q^1_{\pi^1_\theta, \pi^2_\theta}$ and $Q^2_{\pi^1_\theta, \pi^2_\theta}$.

Theorem 1. In this setting, the gradient $\nabla_{\psi^i} J\left(\psi^i\right)$ is exclusively determined by entropy terms.

Proof 1. With $p(\tau)$ denoting the trajectory distribution, observe that the structure of the true $Q^1_{\pi^1_\theta, \pi^2_\theta}$ is:

$$\begin{aligned} & Q^1_{\pi^1_\theta, \pi^2_\theta}(s_0, a^1_0, a^2_0) \\ \triangleq \; & r^1(s_0, a^1_0, a^2_0) + \mathbb{E}_{s \sim p(s_1|a^1_0, a^2_0)}\left(\gamma V^1_{\pi^1_\theta, \pi^2_\theta}(s_1)\right) \\ \triangleq \; & r^1(s_0, a^1_0, a^2_0) + \mathbb{E}_{\tau \sim p(\tau)}\left[\sum_{t=1}^{\infty} \gamma^t \left(r^1_t(s_t, a^1_t, a^2_t) \mathcal{H}\left(\pi^1_\theta(a^1_t|s_t, a^2_t)\rho^1_\psi(a^2_t|s_t)\right)\right)\right] \\ = \; & -r^2(s_0, a^1_0, a^2_0) - \mathbb{E}_{\tau \sim p(\tau)}\left[\sum_{t=1}^{\infty} \gamma^t \left(r^2_t(s_t, a^1_t, a^2_t) + \mathcal{H}\left(\pi^1_\theta(a^1_t|s_t, a^2_t)\rho^1_\psi(a^2_t|s_t)\right)\right)\right] \\ = \; & \left(\sum_{t=1}^{\infty} \gamma^t \mathcal{H}\left(\pi^2_\theta(a^2_t|s_t, a^1_t)\rho^2_\psi(a^1_t|s_t)\right) - \gamma^t \mathcal{H}\left(\pi^1_\theta(a^1_t|s_t, a^2_t)\rho^1_\psi(a^2_t|s_t)\right)\right) \\ & - Q^2_{\pi^1_\theta, \pi^2_\theta}(s_0, a^1_0, a^2_0), \end{aligned} \tag{19}$$

where $\mathcal{H}(\cdot)$ denotes Shannon entropy. For lighter notation, let

$$\mathcal{E} = \left(\sum_{t=1}^{\infty} \gamma^t \mathcal{H}\left(\pi^2(a^2_t|s_t, a^1_t)\rho^2_\psi(a^1_t|s_t)\right) - \gamma^t \mathcal{H}\left(\pi^1(a^1_t|s_t, a^2_t)\rho^1_\psi(a^2_t|s_t)\right)\right). \tag{20}$$

Then, we can determine the gradient as:

$$\begin{aligned} \nabla_{\psi^i} J\left(\psi^i\right) &= \mathbb{E}_{\tau \sim p,\, \epsilon \sim \mathcal{N}}\Big[\nabla_{\psi^i} Q^2_{\pi^1, \pi^2}\left(s_0, f^i_\psi(\epsilon; s_0)\right) - \nabla_{\psi^i} Q^2_{\pi^1, \pi^2}\left(s_0, f^i_\psi(\epsilon; s_0)\right) \\ &\quad + \nabla_{\psi^i} \mathcal{E} - \alpha \nabla_{\psi^i} \log\left(\rho^i_\psi\left(f^i_\psi(\epsilon; s_0)|s_0\right)\right)\Big] \\ &= \mathbb{E}_{\tau \sim p,\, \epsilon \sim \mathcal{N}}\left[\nabla_{\psi^i} \mathcal{E} - \alpha \nabla_{\psi^i} \log\left(\rho^i_\psi\left(f^i_\psi(\epsilon; s_0)|s_0\right)\right)\right] \\ &= \mathbb{E}_{\tau \sim p}\left[\nabla_{\psi^i} \mathcal{E} + \alpha \nabla_{\psi^i} \mathcal{H}\left(\rho^i_\psi(\cdot, \cdot|s_0)\right)\right]. \end{aligned} \tag{21}$$

To alleviate the potential issue of neutralized gradients, we propose to modify the TDOM objective to be based on empirical data. Specifically, we modify the objective function as

$$J\left(\psi^i\right) = \mathbb{E}_{(s,\mathbf{a}^{-j})\sim D,\ \hat{a}^j\sim\rho^i_\psi}\left[\sum_{j\in -i} Q^j_\phi\left(s,\hat{a}^j,\mathbf{a}^{-i\setminus\{j\}},a^i\right) - \alpha\log\rho^j_\psi\left(\hat{\mathbf{a}}^{-j}|s\right)\right]. \quad (22)$$

Note that again we use the reparameterization trick [33] in order to be able to exchange expectation and gradient, while still sampling from the opponent model ρ^i_ψ. The pseudo-code can be found below 1.

Algorithm 1: Multi-agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC)

Initialize replay buffer D and randomly initialize N soft Q networks $Q^{1..n}_{\phi_{1..n}}$, N policy networks $\pi^{1..n}_{\theta_{1..n}}$, and opponent model $\rho^{1..n}_{\psi_{1..n}}$ with parameters $\phi_{i..n}, \theta_{1..n}$ and $\psi_{1..n}$.
Initialize the parameters of target networks with $Q^{1..n}_{\bar{\phi}_{1..n}}$
for each iteration **do**
  Sample s_0 according to $p_0(\cdot)$
  while Not done **do**
  for each agent **do**
    Sample $\hat{\mathbf{a}}^{-i}_t$ from $\rho^i(\cdot|s_t)$ and a^i_t from $\pi^i(\cdot|s_t,\hat{\mathbf{a}}^{-i}_t)$
    Combine the true actions $\mathbf{a}_t = [a^1_t,\ldots,a^n_t]$ and take one step forward
  end for
  Observe $s_{t+1}, \mathbf{r}_t = [r^1_t,\ldots,r^n_t]$ and store $(s_t,\mathbf{a}_t,\mathbf{r}_t,s_{t+1})$ in D
  Sample minibatches of N transitions from D
  for each agent **do**
    Estimate policy gradient according to Eqs. 13, 16, and 22:

$$\begin{aligned}
\phi^i_{t+1} &\leftarrow \phi^i_t + \eta_\phi\hat{\nabla}_\phi J\left(\phi^i_t\right)\\
\theta^i_{t+1} &\leftarrow \theta^i_t + \eta_\theta\hat{\nabla}_\theta J\left(\theta^i_t\right)\\
\psi^i_{t+1} &\leftarrow \psi^i_t + \eta_\psi\hat{\nabla}_\psi J\left(\psi^i_t\right).
\end{aligned}$$

    Update the parameters of target networks $Q^{1..n}_{\bar{\phi}_{1..n}}$
  end for
  end while
end for

5. Simulation results

We compare the proposed TDOM-AC to two state-of-the-art algorithms based on opponent modelling: PR2 [15] and ROMMEO [12], which have shown a better performance with respect to the considered measures compared to *Multi-Agent Soft Q-Learning* MASQL [35] and MADDPG [10] in previous studies. We evaluate the performance of the proposed TDOM-AC methods on a differential game [35,15,12] and the multi-agent particle environments [10]. Those tasks contain fully cooperative and mixed cooperative-competitive objectives with challenging non-trivial equilibria [15] and continuous action space. All the tasks are adopted from PR2 and ROMMEO for adequate comparison.

To reduce the performance difference caused solely by entropy regularization, we add an entropy term to the PR2 objective and equip it with a stochastic policy since both TDOM-AC and ROMMEO employ the maximum entropy reinforcement learning framework. This has been shown to yield better exploration and sample efficiency [33].

For the simulation settings, all policies and opponent models use a fully connected multi-layer perceptron (MLP) with two hidden layers of 256 units each, outputting the mean μ and standard deviation σ of a univariate Gaussian distribution. All hidden layers use the leaky-RelU activation function and we adopt the same invertible squashing function technique as [33] for the output layer. For the Q-network, we use a fully-connected MLP with two hidden layers of 256 units with leaky-Relu activation function, outputting the Q-value. We employ the Adam optimizer with the learning rate $3e-4$ and batch size 256. The target smoothing coefficient τ, entropy control parameter α and the discount factor γ are $0.01, 1$, and 0.95 respectively. All training hyper-parameters are derived from the SAC algorithm (as published in [33]) without any additional adaptations.

5.1. Differential game

The differential Max-of-Two Quadratic Game is a single step continuous action space decision making task, where the gradient update tends to direct the training agent to a sub-optimal point [12]. The reward surface is displayed in the Fig. 2. There exists a local maximum 0 at $(-5,-5)$ and a global maximum 10 at $(5,5)$, with a deep valley positioned in the middle. The agents are rewarded by their joint actions, following the rule: $r_1 = r_2 = max(f_1, f_2)$, where $f_1 = 0.8 * \left[-\left(\frac{a_1+5}{3}\right)^2 - \left(\frac{a_2+5}{3}\right)^2\right]$ and $f_1 = \left[-\left(\frac{a_1-5}{1}\right)^2 - \left(\frac{a_2-5}{1}\right)^2\right] + 10$. Both of the agents have the same continuous action space in the range $[-10, 10]$. Compared to other state-of-the art approaches, TDOM-AC shows a superior performance. In Fig. 3, the learning path of the proposed TDOM-AC is displayed, where the lighter (yellow) dots are sampled later. This indicates a stable and fast convergence. In Fig. 3, the learning curves of all considered algorithms are displayed. Both TDOM-AC and ROMMEO show a fast and stable convergence. However, ROMMEO fails for some random seeds, resulting in a lower average performance. We note that the maximum-entropy version of PR2 indeed converges faster than the original version [15]. Nevertheless, the learning process fluctuates significantly and it suffers from substantial computational cost, see Table 1.

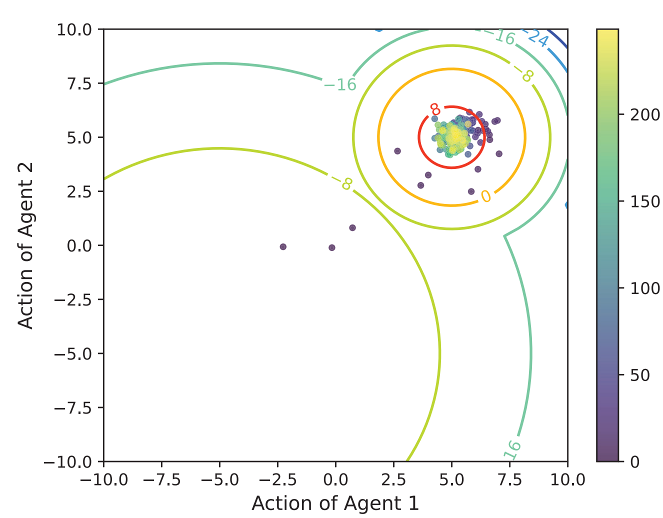


Fig. 2. Reward surface and learning path of agents trained by TDOM-AC. Scattered points are actions taken at each step, the lighter points are sampled later during training.

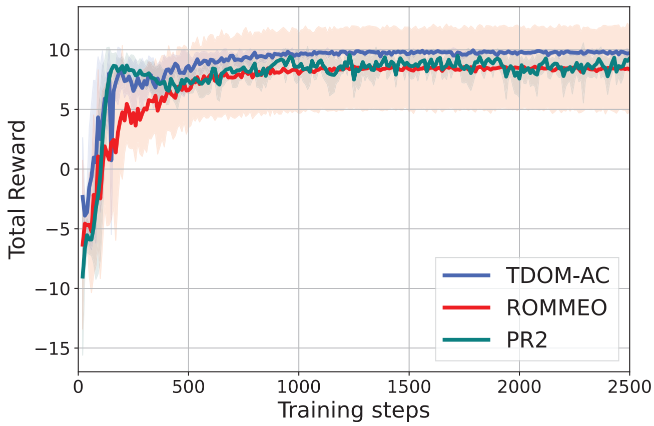


Fig. 3. Average performance of TDOM-AC and other baselines, where the shaded areas show the 1-SD confidence intervals over multiple random seeds.

Table 1
Average running time (seconds) per update of different methods.

Methods	TDOM-AC	ROMMEO	PR2
Running time	**0.068s**	0.089*s*	0.436*s*

5.2. Cooperative navigation

Cooperative Navigation is a three-agent fully cooperative task. The three agents should learn to cooperate to reach and cover three randomly generated landmarks. The agents can observe the relative positions of other agents and landmarks and are collectively rewarded based on the proximity of any agent to each landmark. Besides this, the agents are being penalized when colliding with each other. The expected behavior is to "cover" the three landmarks as fast as possible without any collision. The result shows that TDOM-AC outperforms all other considered baseline algorithms in terms of both faster convergence and a better performance, see Fig. 4. Also, the TDOM-AC attains more accurate opponent behavior prediction, despite the fact that the agents do not have direct access to any opponent action distribution, see Fig. 5. This is in contrast to ROMMEO, which utilizes a regularized opponent model, the regularization being the KL divergence between the opponent model and the empirical opponent distribution.

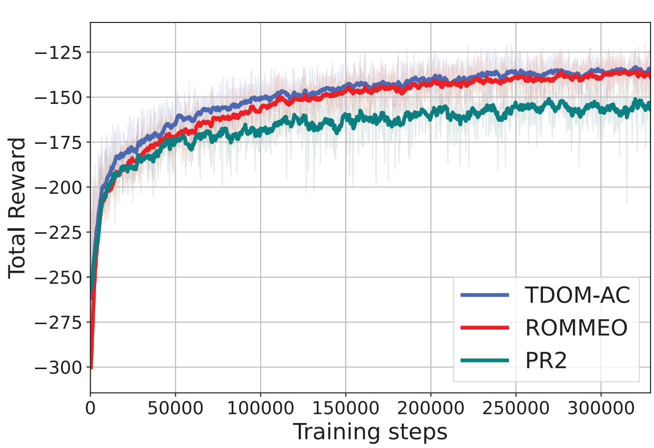


Fig. 4. Moving average of total reward of TDOM-AC and other baselines on Cooperative Navigation.

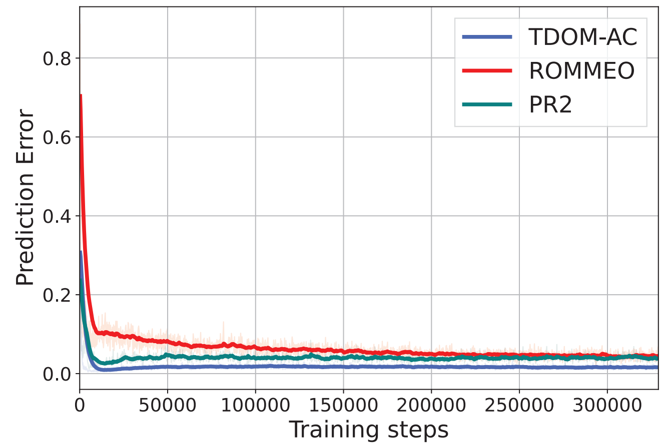


Fig. 5. The test time opponents' behaviors prediction error of TDOM-AC and other baselines.

Table 2
Comparison of different model settings (Agent vs. Adversaries). The values are the normalized average episode advantage scores.

Ag vs. Ads	TDOM-AC	ROMMEO	PR2	Mean
TDOM-AC	0.967	1.000	0.999	**0.989**
ROMMEO	0.674	0.997	0.981	0.884
PR2	0.000	0.722	0.313	0.345
Mean	**0.547**	0.906	0.764	N/A

5.3. Predator and prey

Predator and Prey is a challenging four-agent mixed cooperative-competitive task. There are three slower cooperating adversaries that try to chase the faster agent in a randomly generated environment with two large landmarks impeding the way. The cooperative adversaries are rewarded for every collision with the agent, while the agent is being penalized for any such collision. All agents can observe the relative positions and velocities of other agents and the positions of the landmarks.

For this task, we train all the algorithms for 0.6 M steps and compare the normalized average episode advantage score (the sum of agent's rewards in an episode - the sum of adversaries' rewards in an episode [15,10]. We evaluate the performance of the different algorithms by letting the cooperative adversaries trained by one algorithm play against an agent trained by another algorithm and vice versa. A higher score means the agent (prey) performs better than the cooperative adversaries (predators), while a lower score means that the cooperative adversaries have a superior policy over the agent. Table 2 shows that the TDOM-AC performs best on both prey (0.999) and predator (0.547) side.

6. Conclusion

In this work, we propose a novel time dynamical opponent model called TDOM. It supports mixed cooperative-competitive tasks with a low computational cost. Furthermore, we introduce the TDOM-AC algorithm and demonstrate the superior performance compared to other state-of-the-art methods on multiple challenging benchmarks. In the future, we plan to omit the centralized training and instead also model opponent Q-function parameters as time dynamical latent variables, thereby relying exclusively on past opponent actions for training. Also, we would like to evaluate the proposed approach on more complicated tasks, such as SMAC [36] and investigate the capability on partially observable environments where the agent does not share its observation space with all opponents.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Yuan Tian reports financial support was provided by Swiss National Science Foundation. Olga Fink reports a relationship with Swiss National Science Foundation that includes: funding grants.

Acknowledgement

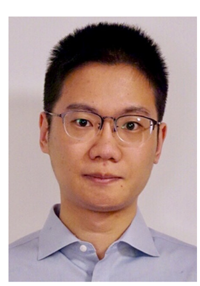
The contributions of Yuan Tian and Olga Fink were funded by the Swiss National Science Foundation (SNSF) Grant No. PP00P2_176878.

References

[1] Y. Du, B. Liu, V. Moens, Z. Liu, Z. Ren, J. Wang, X. Chen, H. Zhang, Learning correlated communication topology in multi-agent reinforcement learning, in: Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems, 2021, pp. 456–464.
[2] O. Vinyals, I. Babuschkin, W.M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D.H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., Grandmaster level in starcraft ii using multi-agent reinforcement learning, Nature 575 (7782) (2019) 350–354.
[3] N. Brown, T. Sandholm, Superhuman ai for multiplayer poker, Science 365 (6456) (2019) 885–890.
[4] OpenAI, Openai five,https://blog.openai.com/openai-five/ (2018).
[5] Y. Yin, X. Bu, P. Zhu, W. Qian, Point-to-point consensus tracking control for unknown nonlinear multi-agent systems using data-driven iterative learning, Neurocomputing.
[6] S. Yuan, C. Yu, P. Wang, Suboptimal linear quadratic tracking control for multi-agent systems, Neurocomputing.
[7] M. Hüttenrauch, S. Adrian, G. Neumann, et al., Deep reinforcement learning for swarm systems, J. Mach. Learn. Res. 20 (54) (2019) 1–31.
[8] P. Hernandez-Leal, M. Kaisers, T. Baarslag, E.M. de Cote, A survey of learning in multiagent environments: Dealing with non-stationarity, arXiv preprint arXiv:1707.09183.
[9] R.S. Sutton, A.G. Barto, R.J. Williams, Reinforcement learning is direct adaptive optimal control, IEEE Control Syst. Mag. 12 (2) (1992) 19–22.
[10] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, I. Mordatch, Multi-agent actor-critic for mixed cooperative-competitive environments, arXiv preprint arXiv:1706.02275.
[11] G.W. Brown, Iterative solution of games by fictitious play, Activity analysis of production and allocation 13 (1) (1951) 374–376.
[12] Z. Tian, Y. Wen, Z. Gong, F. Punakkath, S. Zou, J. Wang, A regularized opponent model with maximum entropy objective, arXiv preprint arXiv:1905.08087.
[13] S.H. Strogatz, Nonlinear dynamics and chaos with student solutions manual: With applications to physics, biology, chemistry, and engineering, CRC Press, 2018.
[14] J.N. Foerster, Y.M. Assael, N. De Freitas, S. Whiteson, Learning to communicate with deep multi-agent reinforcement learning, arXiv preprint arXiv:1605.06676.
[15] Y. Wen, Y. Yang, R. Luo, J. Wang, W. Pan, Probabilistic recursive reasoning for multi-agent reinforcement learning, arXiv preprint arXiv:1901.09207.
[16] R. Kamdar, P. Paliwal, Y. Kumar, A state of art review on various aspects of multi-agent system, Journal of Circuits, Syst. Comput. 27 (11) (2018) 1830006.
[17] K. Zhang, Z. Yang, T. Başar, Multi-agent reinforcement learning: A selective overview of theories and algorithms, Handbook of Reinforcement Learning and Control (2021) 321–384.
[18] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, S. Whiteson, Counterfactual multi-agent policy gradients, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018.
[19] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, J. Wang, Mean field multi-agent reinforcement learning, in: International Conference on Machine Learning, PMLR, 2018, pp. 5571–5580.
[20] Y. Yang, Y. Wen, J. Wang, L. Chen, K. Shao, D. Mguni, W. Zhang, Multi-agent determinantal q-learning, in: International Conference on Machine Learning, PMLR, 2020, pp. 10757–10766.
[21] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, S. Whiteson, Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning, in: International Conference on Machine Learning, PMLR, 2018, pp. 4295–4304.
[22] T. Zhang, Y. Li, C. Wang, G. Xie, Z. Lu, Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning, in: International Conference on Machine Learning, PMLR, 2021, pp. 12491–12500.
[23] K. Son, D. Kim, W.J. Kang, D.E. Hostallero, Y. Yi, Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning, in: International Conference on Machine Learning, PMLR, 2019, pp. 5887–5896.
[24] P. Sunehag, G. Lever, A. Gruslys, W.M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J.Z. Leibo, K. Tuyls, et al., Value-decomposition networks for cooperative multi-agent learning, arXiv preprint arXiv:1706.05296.
[25] Z. Tian, S. Zou, I. Davies, T. Warr, L. Wu, H.B. Ammar, J. Wang, Learning to communicate implicitly by actions, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 7261–7268.
[26] Y. Zheng, Z. Meng, J. Hao, Z. Zhang, T. Yang, C. Fan, A deep bayesian policy reuse approach against non-stationary agents, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, 2018, pp. 962–972.
[27] S.V. Albrecht, P. Stone, Autonomous agents modelling other agents: A comprehensive survey and open problems, Artif. Intell. 258 (2018) 66–95.
[28] M.I. Jordan, Z. Ghahramani, T.S. Jaakkola, L.K. Saul, An introduction to variational methods for graphical models, Mach. Learn. 37 (2) (1999) 183–233.
[29] J.N. Foerster, R.Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, I. Mordatch, Learning with opponent-learning awareness, arXiv preprint arXiv:1709.04326.
[30] M.L. Littman, Markov games as a framework for multi-agent reinforcement learning, in: Machine learning proceedings 1994, Elsevier, 1994, pp. 157–163.
[31] L.S. Shapley, Stochastic games, Proceedings of the national academy of sciences 39 (10) (1953) 1095–1100.
[32] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning nature 518 (7540) (2015) 529–533.
[33] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, in: International conference on machine learning, PMLR, 2018, pp. 1861–1870.
[34] D.P. Kingma, M. Welling, Auto-encoding variational bayes (2014). arXiv:1312.6114.
[35] E. Wei, D. Wicke, D. Freelan, S. Luke, Multiagent soft q-learning, in: 2018 AAAI Spring Symposium Series, 2018.
[36] M. Samvelyan, T. Rashid, C.S. De Witt, G. Farquhar, N. Nardelli, T.G. Rudner, C.-M. Hung, P.H. Torr, J. Foerster, S. Whiteson, The starcraft multi-agent challenge, arXiv preprint arXiv:1902.04043.

Yuan Tian is currently a Ph.D. student at Intelligent Maintenance Systems, ETH Zürich. His research mainly focuses on reinforcement learning and its application, especially in the maintenance domain. Before his Ph.D., he received his bachelor's degree from Beijing Institute of Technology University in 2017, and his master's degree from TU Delft in 2019.

Klaus-Rudolf Kladny received his BSc in computer science at TU Dresden. He is currently pursuing an MSc in data science at ETH Zürich. He is writing his Master's thesis at the Max Planck Institute for Intelligent Systems supervised by Michael Mühlebach and Bernhard Schölkopf.



Qin Wang received his bachelor's degree from Tsinghua University in 2015, and his master's degree and Doctor of Science degree from ETH Zürich in 2018 and 2022. He is currently a postdoc researcher at ETH Zurich. His research mainly focuses on domain adaptation.

Zhiwu Huang received his Ph.D. degree at the University of Chinese Academy of Sciences. He obtained his master and bachelor degrees from Xiamen University and Huaqiao University, respectively. He is currently an Assistant Professor of Computer Science at Singapore Management University. Before taking this professorship, he was a Postdoctoral Researcher at ETH Zurich. His research studies autonomous vision that aims for making machines to learn the visual world, all by themselves. His current focus is on visual deepfake, affective and behavior computing through automated machine learning on data, label, feature, neuron and task.

Olga Fink has been an assistant professor of intelligent maintenance and operations systems at EPFL since March 2022. Olga is also a research affiliate at the Massachusetts Institute of Technology and an Expert of the Innosuisse in the field of ICT. Olga's research focuses on Hybrid Algorithms Fusing Physics-Based Models and Deep Learning Algorithms, Hybrid Operational Digital Twins, Transfer Learning, Self-Supervised Learning, Deep Reinforcement Learning, and Multi-Agent Systems for Intelligent Maintenance and Operations of Infrastructure and Complex Assets. Before joining the EPFL faculty, Olga was an assistant professor of intelligent maintenance systems at ETH Zurich from 2018 to 2022, being awarded the prestigious professorship grant of the Swiss National Science Foundation (SNSF).