An Open-Ended Learning Framework for Opponent Modeling

Yuheng Jing[1,2], Kai Li[1,2†], Bingyun Liu[1,2], Haobo Fu[6], Qiang Fu[6], Junliang Xing[5], Jian Cheng[1,3,4†]
[1]Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
[3]School of Future Technology, University of Chinese Academy of Sciences
[4]AiRiA
[5]Tsinghua University
[6]Tencent AI Lab
kai.li@ia.ac.cn, jian.cheng@ia.ac.cn

Abstract

Opponent **M**odeling (OM) aims to enhance decision-making by modeling other agents in multi-agent environments. Existing works typically learn opponent models against a predesignated fixed set of opponents during training. However, this will cause poor generalization when facing unknown opponents during testing, as previously unseen opponents can exhibit *out-of-distribution* (OOD) behaviors that the learned opponent models cannot handle. To tackle this problem, we introduce a novel **O**pen-**E**nded **O**pponent **M**odeling (OEOM) framework, which continuously generates opponents with diverse strengths and styles to reduce the possibility of OOD situations occurring during testing. Founded on population-based training and information-theoretic trajectory space diversity regularization, OEOM generates a dynamic set of opponents. This set is then fed to any OM approaches to train a potentially generalizable opponent model. Upon this, we further propose a simple yet effective OM approach that naturally fits within the OEOM framework. This approach is based on in-context reinforcement learning and learns a Transformer that dynamically recognizes and responds to opponents based on their trajectories. Extensive experiments in cooperative, competitive, and mixed environments demonstrate that OEOM is an approach-agnostic framework that improves generalizability compared to training against a fixed set of opponents, regardless of OM approaches or testing opponent settings. The results also indicate that our proposed approach generally outperforms existing OM baselines.

1 Introduction

Developing *autonomous agents* capable of modeling *other agents* (including adversaries, teammates, *etc*., collectively referred to as "***opponents***") in a multi-agent environment is a long-standing research topic commonly known as **O**pponent **M**odeling (**OM**).[1] This topic aims to model the behaviors, goals, beliefs, and other properties of opponents to reduce the uncertainty of the self-agent regarding the environment and enhance its decision-making capabilities (Albrecht and Stone 2018; Nashed and Zilberstein 2022; Lu et al. 2022; Zhao et al. 2022). The OM domain typically adopts a two-stage learning paradigm: (1) **Training**: An opponent model is trained on *a set of training opponent policies* (termed as Π^{train}), where the model learns to respond to opponents based on their information. (2) **Testing**: The trained opponent model is evaluated on *an unknown set of testing opponent policies* (termed as Π^{test}), where the model needs to quickly adapt to non-stationary opponents and generalize to handle unseen opponents.

Existing works typically assume that Π^{train} is a predesignated, fixed set of policies, either consisting of handcrafted heuristic scripts or obtained through naive self-play methods. However, this results in the opponent models learned from training having poor generalization ability during testing, making it particularly difficult to handle unseen opponents, as unseen opponents can exhibit many *out-of-distribution* (OOD) behaviors never encountered.

To tackle this issue, we argue that using a *dynamic* Π^{train}, which continuously grows in terms of *strength* and *style*, can endow opponent models with stronger generalizabilities compared to a *static* Π^{train}. This argument stems from the following intuitions: (1) The more diverse the opponents that the opponent model encounters during training, the less likely it is to face OOD opponent behaviors during testing. (2) A good diversity of Π^{train} should consider not only *game performance* (strength) but also *behavioral patterns* (style).

Based on the above observations, our first contribution is proposing the **O**pen-**E**nded **O**pponent **M**odeling (**OEOM**) framework to encode the idea of generating a dynamic Π^{train}. OEOM is a general framework that continuously generates opponents with diverse strengths and styles for any OM approach to training on, thereby enhancing its generalizability during testing. The diversity in strength within OEOM is built upon *Population-Based Training* (PBT) (Jaderberg et al. 2017), which uses *Reinforcement Learning* (RL) objectives to conduct cross-play training and natural selection. This iterative process continuously generates policies with increasing strength. The diversity in style within OEOM is established through a novel information-theoretic *Trajectory Space Diversity* (TSD) regularization. This regularization maximizes the mutual information between the policy index and the observation-action sequences (*i.e*., trajectories). The style diversity of policies should be reflected in the difference in trajectories they visit. By maximizing TSD, different policies are driven to visit as different trajectories as possible, thus generating diverse styles of policies. OM ap-

[0]The symbol † denotes the corresponding authors.

[1]We refer to *autonomous agent* as *self-agent* in our paper.

proaches trained with the diversified opponents produced by OEOM can encounter much fewer OOD situations during testing, thereby enhancing their generalizabilities.

After generating Π^{train} using OEOM, this set can be fed to any OM approach to train opponent models. Although various approaches exist in OM, such as those based on representation learning (Grover et al. 2018; Papoudakis et al. 2021), Bayesian learning (Zheng et al. 2018; Fu et al. 2022), and shaping opponents' learning (Willi et al. 2022; Letcher et al. 2019), they always rely on complex and cumbersome methodologies. We offer a new perspective that OM is essentially a *sequence-to-sequence* (Seq2Seq) problem, which constitutes our second contribution. The input sequence of this Seq2Seq problem consists of the historical data generated from interactions with the opponent, while the output sequence represents the optimal sequence of actions that the self-agent needs to take. Specifically, we contribute a new **I**n-Context-RL-based **OM** (**IOM**) approach, which conditions on the opponent's trajectory and learns a Transformer that dynamically recognizes and responds to opponents.

We conduct extensive comparative analyses and ablation studies in competitive, cooperative, and mixed environments to validate the rationale of OEOM compared to training opponent models against a fixed Π^{train}. OEOM is empirically shown to be an approach-agnostic framework that enhances generalization ability regardless of the OM approach or the testing opponent settings. Additionally, experimental results support the effectiveness of our IOM approach, as it generally outperforms existing OM baselines when facing various opponents during both training and testing stages of OM.

2 Related Work

Opponent Modeling. Existing OM approaches can be divided into two major categories based on their focus: (1) those *focus on training* and (2) those *focus on testing*.

The first category focuses on learning high-level latent knowledge about opponents during training and generalizing it to testing. Some adopt the idea of representation learning (He et al. 2016; Grover et al. 2018; Zintgraf et al. 2021; Papoudakis et al. 2021), aiming to learn high-quality representations of opponent policies during training to assist in policy optimization. Some others employ non-meta-gradient meta-learning methods (Zintgraf et al. 2021; Jing et al. 2023), attempting to use recurrent architectures to learn the internal structure of each opponent's policy and the differences between them during training. Our work provides a unified, open-ended framework for this category and contributes a novel perspective: improving generalization ability by generating a more reasonable set of opponent policies.

The second category focuses on model updating or online inference during testing to specifically adapt to the current opponent. Some employ Bayesian learning methodologies (Zheng et al. 2018; DiGiovanni and Tewari 2021; Fu et al. 2022; Lv et al. 2023), attempting to detect or infer the opponent's policy during testing and generate the corresponding responses. Some others adopt meta-gradient-based meta-learning concepts (Al-Shedivat et al. 2018; Kim et al. 2021; Wu et al. 2022), leveraging the well-initialized solutions obtained during training in the parameter space to fine-tune and quickly adapt to the test opponents. Additionally, work on shaping opponents' learning (Foerster et al. 2018a,b; Letcher et al. 2019) and recursive reasoning (Wen et al. 2019; Yu et al. 2022) falls into this category. Theoretically, this category can also benefit from the OEOM framework, as their adaptive capabilities during testing still rely on the learning status of the training.

Open-Ended Learning. The open-ended learning domain aims to continuously discover and solve new problems. In single-agent RL, many studies focus on continually generating increasingly challenging environments to train generally capable agents (Wang et al. 2020; Team et al. 2021; Bauer et al. 2023). In multi-agent RL, existing work respectively focuses on two types of games. For zero-sum games, they study the generation of adaptive objectives to expand the frontiers of strategies (Balduzzi et al. 2019; McAleer et al. 2020; Liu et al. 2021). For cooperative games, they aim to continuously evaluate and identify the cooperative abilities of strategies to generate more adaptive collaborators (Xue et al. 2022; Li et al. 2023). In contrast, our work is not limited by the type of game, which continually provides new opponents as objectives for OM training, resulting in opponent models with gradually enhanced generalizability.

Diversity for Decision Making. Learning diversity for decision-making problems is a growing research area. In single-agent RL, the focus is usually on leveraging diversity to encourage exploration. These studies formalize diversity as a function of action distribution (Cohen et al. 2019; Parker-Holder et al. 2020), state distribution (Song et al. 2019; Kumar et al. 2020), or state-action distribution (Gangwani, Liu, and Peng 2018) for optimization. Additionally, Eysenbach et al. (2019) explores unsupervised learning of multiple skills that are distinguishable by a classifier based on state distributions. In multi-agent RL, optimizing diversity serves various purposes. Perez-Nieves et al. (2021); Yao et al. (2024) target accelerating convergence to Nash equilibria in zero-sum games. Additionally, Lupu et al. (2021); Zhao et al. (2023) focuses on learning a diverse population of strategies to enable zero-shot coordination in cooperative games. It is worth mentioning that our work shares a similar goal with Lupu et al. (2021); Zhao et al. (2023), namely improving generalization ability during testing.

In-Context RL. Algorithmically, *In-Context RL* (ICRL) can be considered as taking a more agnostic approach by learning the RL algorithm itself (Duan et al. 2016; Mishra et al. 2018). Laskin et al. (2023); Lee et al. (2023) propose supervised pretraining to empirically demonstrate ICRL abilities in decision-making. Lin, Bai, and Mei (2024) further introduce a theoretical analysis framework to explain the principles and working conditions of ICRL. Unlike existing work focusing on single-agent settings, our work explores the empirical effects of an ICRL approach within the framework of OEOM on multi-agent settings.

3 Preliminaries

We use an n-agent *Partially-Observable Stochastic Game* (POSG) $\langle \mathcal{S}, \{\mathcal{O}^i\}_{i=1}^n, \{\mathcal{A}^i\}_{i=1}^n, \mathcal{P}, \{R^i\}_{i=1}^n, \{\Omega^i\}_{i=1}^n, T \rangle$ to

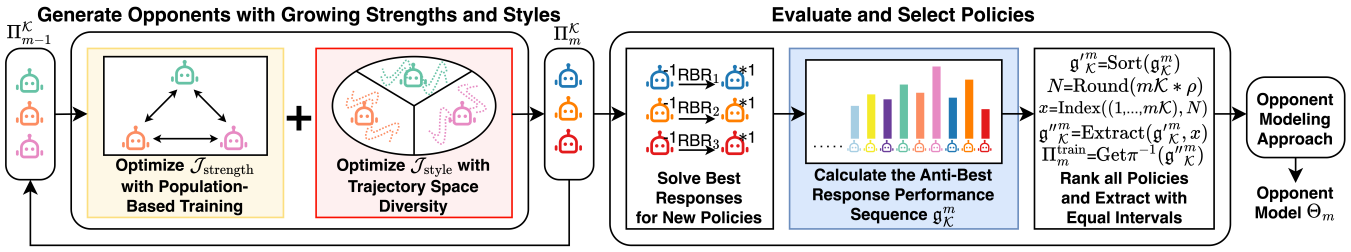

Figure 1: Iteration process of OEOM.

formalize the multi-agent environment (Yang and Wang 2020). $\mathcal{S}$ is the state space. $\mathcal{O}^i$ is the observation space of agent $i \in [n]$, $\mathcal{O} = \prod_{i=1}^n \mathcal{O}^i$ is the joint observation space. $\mathcal{A}^i$ is agent i's action space, $\mathcal{A} = \prod_{i=1}^n \mathcal{A}^i$ is the joint action space. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the transition dynamics. $R^i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function of agent i. $\Omega^i : \mathcal{S} \times \mathcal{A} \times \mathcal{O}^i \to [0, 1]$ is the agent i's observation function. T is the episodic horizon.

We mark the agent under our control, *i.e.*, *self-agent*, with superscript 1, while the other $n-1$ agents, *i.e.*, *opponents*, are marked with superscript -1. The joint policy of opponents is $\pi^{-1}(a^{-1}|o^{-1}) = \prod_{j \neq 1} \pi^j(a^j|o^j)$, where a^{-1} is joint actions of opponents and o^{-1} is their joint observations. Let the opponents' trajectory at timestep t be denoted as $\tau_t^{-1} = (o_0^{-1}, a_0^{-1}, r_0^{-1}, ..., o_t^{-1}, a_t^{-1}, r_t^{-1})$ and their complete trajectory as $\tau^{-1} := \tau_{T-1}^{-1} + (o_T^{-1})$, where r^{-1} is opponents' joint rewards. At any episode e and any timestep t, *opponent historical trajectories* $\mathfrak{T}_t^{-1(e)} := (\tau^{-1(0)}, ..., \tau^{-1(e-1)}, \{a_{t'}^{-1(e)}\}_{t'=0}^{t-1})$ is available to self-agent.

In OM, the policy of the self-agent can be typically represented as $\pi^1(a^1|o^1, D)$, which adjusts based on the *opponent information data* D . The data D can either be directly derived from a subset of $\mathfrak{T}^{-1}$ or obtained by learning a representation from $\mathfrak{T}^{-1}$. Leveraging the training stage, the objective of the self-agent is to maximize its expected *return* (*i.e.*, cumulative reward) during testing:

$$\mathop{\mathbb{E}}_{\substack{\mathcal{P},\{\Omega^i\}_{i=1}^n,\pi^1(\cdot|\cdot,D),D\leftarrow\mathfrak{T}^{-1},\\ \pi^{-1}\sim\Pi^{\text{test}},\pi^1\leftarrow\text{Training}(\Pi^{\text{train}})}} \left[\sum_{t=0}^{T-1} R_t^1\right]. \tag{1}$$

4 Methodology

In this section, Sec. 4.1 introduces how we construct OEOM based on **P**opulation-**B**ased **T**raining (**PBT**) and a novel **T**rajectory **S**pace **D**iversity (**TSD**) to optimize policy strength and policy style, respectively. In Sec. 4.2, we propose an OM approach based on **I**n-**C**ontext **RL** (**ICRL**), which can seamlessly fit within the OEOM framework to maximize its effectiveness. In Sec. 4.3, we provide a theoretical analysis of the OEOM framework.

4.1 Open-Ended Opponent Modeling Framework

From Eq. (1), it can be seen that the objective of OM heavily relies on the *set of training opponent policies* Π^{train}. Existing work typically uses scripted or simplistic methods to construct Π^{train}, which results in weak generalization when facing the *unknown set of testing opponent policies* Π^{test}. To address this problem, we propose OEOM, which continually generates opponents with increasing diversity in strength and style, for training any OM approach.

The overview of the OEOM framework is shown in Fig. 1, and its iteration mainly consists of three steps: (1) Optimize policy strength and policy style, (2) Evaluate and select policies, and (3) Train the OM approach with the generated Π^{train}. Following, we will introduce each step in detail.

Optimize policy strength and policy style. Our core objective is to generate a *dynamic* Π^{train}. Here, dynamic not only means that the number of policies in Π^{train} gradually increases but also that Π^{train} needs to exhibit progressively *growing diversity*. This is because a reasonable open-ended system should be capable of continuously producing new creations that are distinct from previous ones. We argue that for Π^{train}, good diversity needs to consider two aspects: *strength diversity*, determined by game performance, and *style diversity*, determined by behavior patterns. Based on this reasoning, we design OEOM to explicitly optimize the objectives of *policy strength* and *policy style* to generate a Π^{train} with progressively increasing diversity in both aspects.

First, OEOM optimizes **policy strength** based on PBT. PBT collects samples through cross-play, optimizes each policy using the RL objective, and applies natural selection. It has been shown to be effective in optimizing game performance in decision-making problems (Jaderberg et al. 2019; Long et al. 2023). Assuming a population $\Pi^{\mathcal{K}}$ of size $\mathcal{K}$, where the k-th *joint policy* is π_k, PBT in each iteration samples two policies π_j and π_k from $\Pi^{\mathcal{K}}$ according to a uniform distribution $\text{U}[\mathcal{K}]$. π_j^1 and π_k^{-1} are used as the self-agent and the opponent, respectively, to interact and collect samples, and to maximize the expected returns for both policies. The objective for optimizing *opponent policies* through PBT is:

$$\mathcal{J}_{\text{strength}} = \mathbb{E}_{(j,k)\sim\text{U}[\mathcal{K}]}\left[\sum_{t=0}^{T-1} R_t^{-1} \mid \pi_j^1, \pi_k^{-1}\right]. \tag{2}$$

After completing the policy updates, PBT conducts a natural selection process to eliminate the weakest policies. It evaluates all policies in $\Pi^{\mathcal{K}}$ by pairing them as both self-agent and opponent, identifying the weakest policy π_{worst} and the strongest policy π_{best}. For the *opponent policies*, we replace the weakest with the strongest, *i.e.*, $\pi_{\text{worst}}^{-1} \leftarrow \pi_{\text{best}}^{-1}$. By continuously using PBT for cross-play and natural selection, we can generate increasingly stronger opponents in strength.

Secondly, OEOM optimizes **policy style** based on a novel TSD. We argue that policies should be distinguishable based on the trajectories they visit. In other words, to generate a diverse set of policies, the policies within the set must partition the entire trajectory space as distinctly as possible. We incorporate a TSD regularization term into the PBT objective, encoding this idea by maximizing the mutual information between the policy index and the observation-action sequences. Assuming that the random variables for the opponent's observations and actions are O^{-1} and A^{-1}, respectively, and K is the random variable representing the *opponent policy index*, the TSD objective is to maximize:

$$I((O^{-1}\times A^{-1})^T; K) := I(O_0^{-1}, A_0^{-1}, ..., O_{T-1}^{-1}, A_{T-1}^{-1}; K). \tag{3}$$

However, Eq. (3) is intractable because it requires integration over all possible trajectories. To make optimizing TSD tractable, we optimize the variational lower bound of Eq. (3):

$$\begin{aligned}\mathcal{J}_{\text{style}} = \mathbb{E}_{k\sim p(k)}[\sum_{t=0}^{T-1}(\log q(k|h_t^{-1}) \\ - \log q(k|h_{t-1}^{-1})) \mid \pi_k^{-1}].\end{aligned} \tag{4}$$

$p(k)$ is the distribution of opponent policy index k, which we assume to be a uniform distribution $\text{U}[\mathcal{K}]$. $h_t^{-1} = (o_0^{-1}, a_0^{-1}, \ldots, o_t^{-1}, a_t^{-1})$ is the *opponent's observation-action sequence*. q is a variational distribution used to approximate the posterior distribution of $p(k|h_t^{-1})$. We provide the proof that Eq. (4) is a lower bound of Eq. (3) in Sec. 4.3.

By combining the objectives of optimizing policy strength and policy style, we obtain the final optimization objective:

$$\mathcal{J}_{\text{oeom}} = \alpha \cdot \mathcal{J}_{\text{strength}} + \beta \cdot \mathcal{J}_{\text{style}} \tag{5a}$$

$$= \mathbb{E}_{(j,k)\sim \text{U}[K], t\sim \text{U}[T-1]} \tag{5b}$$

$$\left[\alpha \cdot R_t^{-1} + \beta \cdot (\log q(k|h_t^{-1}) - \log q(k|h_{t-1}^{-1})) \mid \pi_j^1, \pi_k^{-1}\right].$$

Here, α, β are hyperparameters used to balance the two objectives. Optimizing TSD can be viewed as adding a pseudo-reward to the original RL objective in PBT, driving different policies to visit different trajectories. The optimization of OEOM does not impose any specific requirements on which RL algorithm to use. Specifically, we use PPO (Schulman et al. 2017) to optimize Eq. (5).

In each iteration of OEOM, we repeat the following process multiple times to generate $\mathcal{K}$ new opponent policies through PBT and TSD: sample two policies from $\Pi^{\mathcal{K}}$, have them interact to generate data, and maximize Eq. (5).

Evaluate and select policies. Theoretically, OEOM can continue running indefinitely, generating infinite opponents. However, considering computational feasibility, we terminate OEOM at a certain iteration and evaluate the generated policies to select representative ones for downstream training. In experiments in Sec. 5, we will see that even this finite set of opponents has proven effective in enhancing the generalization abilities of OM approaches.

Assuming we have reached the m-th iteration, after completing optimizations using Eq. (5), we obtain $\mathcal{K}$ *new opponent policies* $\Pi_m^{\mathcal{K}} = \{\pi_k^{-1^{(m)}}\}_{k=1}^{\mathcal{K}}$. Including the policies generated in previous iterations $\Pi_1^{\mathcal{K}}, \ldots, \Pi_{m-1}^{\mathcal{K}}$, there are a total of $m\mathcal{K}$ opponent policies. To select the most representative ones from them, we use their *Anti-Best Response Performance* (ABRP), *i.e.*, their performance in the worst-case scenario, as the evaluation criterion for all policies. Let the *Best Response* (BR) to an opponent policy π_k^{-1} be denoted as π_k^{*1}, we measure the ABRP of π_k^{-1} by the expected return $g_k = \mathbb{E}\left[\sum_t R_t^{-1}\right]$ when π_k^{-1} plays against π_k^{*1}.

If a new BR is trained from scratch for each iteration to evaluate ABRP, it would be a highly resource-intensive process. Therefore, instead of training from scratch, we train $\mathcal{K}$ *Running Best Responders* (RBRs) to solve the BR for each of the $\mathcal{K}$ new policies generated in each iteration and store the resulting BRs. The RBRs are shared across different iterations. After solving for the BRs of the opponent policies generated in the m-th iteration, we get the *ABRP sequence* $\mathfrak{g}_{\mathcal{K}}^m = [g_1{}^{(1)}, \ldots, g_{\mathcal{K}}{}^{(1)}, \ldots, g_1{}^{(m)}, \ldots, g_{\mathcal{K}}{}^{(m)}]$.

To obtain a representative set that characterizes $m\mathcal{K}$ generated opponents, we select opponents at equal intervals based on ABRP ranking. We (1) rank $\mathfrak{g}_{\mathcal{K}}^m$ in ascending order to get $\mathfrak{g}'^m_{\mathcal{K}}$, (2) determine number of policies to select N based on proportion of policies to be chosen ρ, (3) get N indexes x from $(1, ..., m\mathcal{K})$ at equal intervals, (4) extract a subsequence $\mathfrak{g}''^m_{\mathcal{K}}$ from $\mathfrak{g}'^m_{\mathcal{K}}$ by x, and (5) get all corresponding opponent policies of $\mathfrak{g}''^m_{\mathcal{K}}$ to form the m-th set Π_m^{train}.

Train OM approach with generated Π^{train}. After obtaining Π_m^{train} in the m-th iteration, we treat the downstream OM approach as a black box, directly using Π_m^{train} as input. The OM approach then undergoes several training steps and outputs the learned *opponent model* Θ_m. To fully utilize the computations during the optimation step of the OEOM, we also input the set of BRs corresponding to each opponent policy in Π_m^{train}, denoted as Π_m^{BR}, into the OM approach to facilitate their learning process.

4.2 ICRL-based Opponent Modeling Approach

Though existing work develops diverse OM approaches, their methodologies tend to be complex and cumbersome. We reconsider the problem from its essence, treating OM as a *sequence-to-sequence* (Seq2Seq) problem. Here, the input consists of historical trajectory sequences of both the opponent and the self-agent, and the output is the optimal action sequence for the self-agent in response to the opponent.

Building upon the above idea, we propose a **I**CRL-based **OM** (**IOM**) approach. IOM directly uses the *opponent observation-action sequence* $h_{T-1}^{-1} = (o_0^{-1}, a_0^{-1}, \ldots, o_{T-1}^{-1}, a_{T-1}^{-1})$ to extract the *opponent information data* D for recognizing the opponent policies. Then, IOM supervised learns on the BRs against the opponent policies to learn the optimal action sequences for responding to them. Thanks to the opponents generated by OEOM with diverse strengths and styles, as reflected in their trajectories h_{T-1}^{-1}, by training on such data, IOM can gain a better ability to handle a wide range of unknown opponents.

IOM uses a simple yet reasonable *Construction Strategy for D* (CSD): Assuming current episode is the e-th episode, we randomly sample a consecutive sequence of length H of

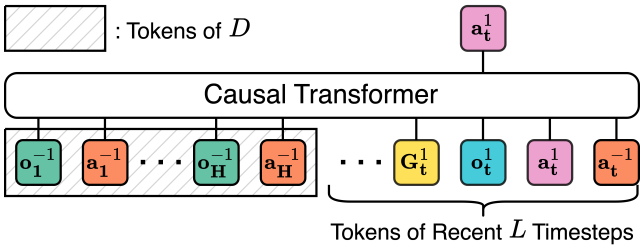

Figure 2: The architecture of IOM.

(o^{-1}, a^{-1}) from the last episode's sequence $h_{T-1}^{-1\ (e-1)}$.

During the training stage, IOM samples an opponent policy π^{-1} from Π^{train} and obtains the corresponding BR π^{*1} for that opponent from Π^{BR}. Then, it interacts with π^{-1} using π^{*1} for several rounds. It constructs D using the aforementioned CSD, and then conditions on D and the self-agent's *reward-to-go* $G_t^1 = \sum_{t'=t}^{T-1} R_{t'}^1$ to imitate π^{*1}'s policy. Assume that the model for IOM is denoted as Θ, the optimization objective of IOM is as follows:

$$\mathcal{J}_{\text{iom}} = \mathbb{E}_{\pi^{*1} \leftarrow \Pi^{\text{BR}}(\pi^{-1}), \pi^{-1} \sim \Pi^{\text{train}}} \left[\log \Theta(a^1 | o^1, G^1, D) \mid \pi^{*1}, \pi^{-1} \right]. \tag{6}$$

During the testing stage, IOM uses the CSD above to construct D to recognize the current opponent policy, specifies the G^1 desired to achieve, and then generates appropriate actions on the fly as it interacts with testing opponents.

IOM adopts a Transformer-based architecture, as shown in Fig. 2. This model maintains a self-agent in-episode historical trajectory sequence of length L. We also incorporate opponent in-episode historical action sequence $\{a_{t'}^{-1}\}_{t'=0}^{t-1}$ to fully leverage the information obtained from interactions with the opponent, which benefits IOM's learning.

4.3 Theoretical Analysis

Theorem 4.1. *$I((O^{-1} \times A^{-1})^T; K) \geq \mathcal{J}_{style}$. '=' holds if and only if $\forall k \in [K], h_t^{-1}, p(k \mid h_t^{-1}) = q(k \mid h_t^{-1})$.*

Proof. Define $q(k|h_{-1}^{-1}) \equiv 1$, we obtain:

$$\mathcal{J}_{\text{style}} = \mathbb{E}_{k \sim p(k)} \left[\log q(k | h_{T-1}^{-1}) \mid \pi_k^{-1} \right]. \tag{7}$$

Expanding $I((O^{-1} \times A^{-1})^T; K)$, we obtain:

$$I((O^{-1} \times A^{-1})^T; K) = H(K) - H(K \mid (O^{-1} \times A^{-1})^T). \tag{8}$$

H represents Shannon entropy. Since $H(K) \geq 0$, we have:

$$I((O^{-1} \times A^{-1})^T; K) \geq -H(K \mid (O^{-1} \times A^{-1})^T). \tag{9}$$

Based on Eqs. (7) to (9), to prove $I((O^{-1} \times A^{-1})^T; K) \geq \mathcal{J}_{\text{style}}$, we only need to prove that:

$$\mathbb{E}_{k \sim p(k)} \left[\log p(k | h_{T-1}^{-1}) - \log q(k | h_{T-1}^{-1}) \mid \pi_k^{-1} \right] \geq 0. \tag{10}$$

Since $D_{KL}(p(k|h_t^{-1}) || q(k|h_t^{-1})) \geq 0$, we have:

$$D_{KL}(p(k|h_t^{-1}) || q(k|h_t^{-1})) = \mathbb{E}_{p(k|h_t^{-1})} \left[\log \frac{p(k|h_t^{-1})}{q(k|h_t^{-1})} \right] = \mathbb{E}_{p(k|h_t^{-1})} \left[\log p(k|h_t^{-1}) - \log q(k|h_t^{-1}) \right] \geq 0. \tag{11}$$

Taking the expectation of Eq. (11) with respect to $p(h_t^{-1})$, *i.e.*, the distribution of h_t^{-1}, we obtain:

$$\mathbb{E}_{p(k, h_t^{-1})} \left[\log p(k|h_t^{-1}) - \log q(k|h_t^{-1}) \right] \geq 0. \tag{12}$$

Based on conditional entropy, Eq. (12) is equal to:

$$\mathbb{E}_{k \sim p(k)} \left[\log p(k|h_t^{-1}) - q(k|h_t^{-1}) \mid \pi_k^{-1} \right] \geq 0. \tag{13}$$

For Eq. (13), when $t = T - 1$, Eq. (10) holds. □

Thm. 4.1 ensures the TSD optimization objective, *i.e.*, Eq. (4), is a lower bound of Eq. (3). With this foundation, TSD can ensure that OEOM drives different opponent policies to visit different parts of the trajectory space.

5 Experiments

In this section, Sec. 5.1 presents our experimental environments, baselines, and evaluation protocols. Sec. 5.2 poses a series of questions and provides empirical results to answer them, aiming to analyze the effectiveness of the OEOM framework and the IOM approach.

5.1 Experimental Setup

Environments. We consider three sparse-reward evaluation environments for OM:

- `Predator Prey (PP)` is a competitive environment with a continuous state space. The self-agent is a prey who aims to avoid being hit by three predators as much as possible. The challenge of `PP` lies in the need to model three opponents simultaneously and handle their cooperation.
- `Level-Based Foraging (LBF)` is a mixed environment. The self-agent aims to eat as many apples as possible. The challenge of `LBF` is that cooperation with the opponent is necessary to eat apples of a higher level than the self-agent's. `LBF` represents a typical social dilemma.
- `OverCooked (OC)` is a cooperative environment using high-dimensional images as states. The self-agent aims at collaborating with the opponent to serve dishes as much as possible. The challenge of `OC` lies in the high-intensity coordination required between the two agents to complete a series of sub-tasks to serve a dish successfully.

Opponent Generation (OG) Baselines. We consider commonly used methods in OM and two variants of OEOM:

- **Script**: Rule-based handcrafted script policies designed heuristically according to the characteristics of the environment. This method has been used in He et al. (2016); Zintgraf et al. (2021); Zheng et al. (2018) and *etc*.
- **S**elf-**P**lay (**SP**): Use only *one* single *joint policy* to collect samples, and each agent optimizes its own RL objective. This method has been used in Papoudakis et al. (2021); Al-Shedivat et al. (2018); Kim et al. (2021) and *etc*.
- **OEOM**$_{-TSD}$: An ablation of OEOM (equivalent to the original PBT), where we set β to 0 to disable TSD.
- **OEOM**$_{-PBT}$: An ablation of OEOM (equivalent to optimizing only TSD), where we set α to 0 to disable the original RL objective of PBT.

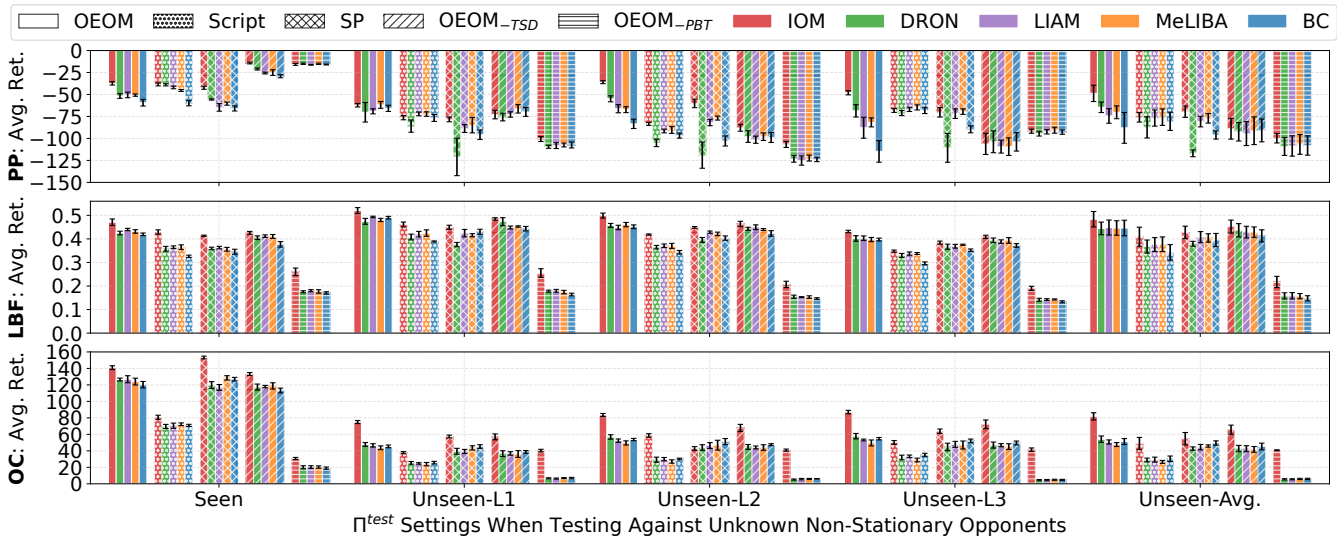

Figure 3: Average results of testing all the OM approaches trained with Π^{train} generated by different OG methods.

OM Baselines. We consider representative approaches *focused on training*, where *IOM also belongs to this category*.

- **DRON** (He et al. 2016): Encode hand-crafted features of opponents using a Mixture-of-Expert network while also predicting opponents' actions to model the opponents, which is the most performant version in their paper.
- **LIAM** (Papoudakis et al. 2021): Use the observations and actions of the self-agent to reconstruct those of the opponent through an auto-encoder, thereby embedding the opponent policy into a latent space.
- **MeLIBA** (Zintgraf et al. 2021): Use variational auto-encoder (VAE) to reconstruct the opponent's future actions and condition on the embedding generated by this VAE to learn a Bayesian meta-policy.
- **B**ehavioral **C**loning (**BC**): Imitate the BRs in Π^{BR} by maximizing the log likelihood of the label action a^1 given the observation o^1. BC serves as a non-OM baseline used to ablate the effectiveness of the OM approaches.

OG Protocols. For each OG method, we construct a Π^{train} of size 30. For OEOM, OEOM$_{-TSD}$, and OEOM$_{-PBT}$, we set the population size $\mathcal{K} = 6$, the number of iterations $m = 25$, and the selection ratio $\rho = 0.2$ to generate 30 opponent policies. For SP, we run 30 different seeds, using the same total training steps as OEOM, to generate 30 opponent policies. For Script, we manually create 30 different hard-coded opponent policies for each environment.

OM Protocols. During the training stage, each OM approach is trained for 30000 steps using the given Π^{train}. For the testing stage, we construct four different Π^{test} settings: (1) *Seen*, (2) *Unseen-L1*, (3) *Unseen-L2*, and (4) *Unseen-L3*. *Seen* uses Π^{train} as Π^{test}, while *Unseen-L1* to *Unseen-L3* are constructed using three *Levels* of opponent policies never appeared in Π^{train}, where the *higher the level*, the *stronger* the opponents are *in strength*. We assume the test opponents are *unknown* and *non-stationary*, with their true policies unknown to the self-agent, and sample a policy from Π^{test} every 10 episodes. All OM approaches use the final checkpoint from training to play 900 episodes against the unknown non-stationary opponents, who switch policies a total of 90 times.

All bar and line charts report the *average* and *standard deviation* of the mean results over 5 random seeds.

5.2 Empirical Analysis

Question 1. *Can OEOM effectively improve the generalization ability of OM approaches than existing OG methods?*

In Fig. 3, we show the testing results for all OM approaches trained with Π^{train} generated by different OG methods. The "*Unseen-Avg*" represents the average of the results under the *Unseen-L1* to *Unseen-L3* settings, indicating the overall performance against unseen opponents. Fig. 3 shows that OEOM can generally enhances the performance of OM approaches compared to other OG baselines, especially when opponents are unseen. *e.g.*, in `PP` and `OC`, most OM approaches trained with OEOM achieve the best *Unseen-Avg* results. This indicates that, compared to existing OG methods, OEOM improves the generalization abilities of OM approaches. Additionally, OEOM generally improves the performance of OM approaches compared to OEOM$_{-TSD}$ and OEOM$_{-PBT}$. This validates our hypothesis that *diverse opponents* in terms of *strength* and *behavior* are key to improving the generalization ability of OM.

Question 2. *Can IOM better exploit advantage of OEOM?*

Continuing to observe Fig. 3, we find that IOM generally outperforms other OM approaches across all settings, regardless of the OG methods used. When we focus on OEOM, we further notice that the improvement of IOM over other OM baselines is particularly notable when trained with OEOM. *e.g.*, in *Unseen-L3* of `PP`, IOM trained with other OG methods performs similarly to other baselines, but when trained with OEOM, it effectively outperforms the baselines.

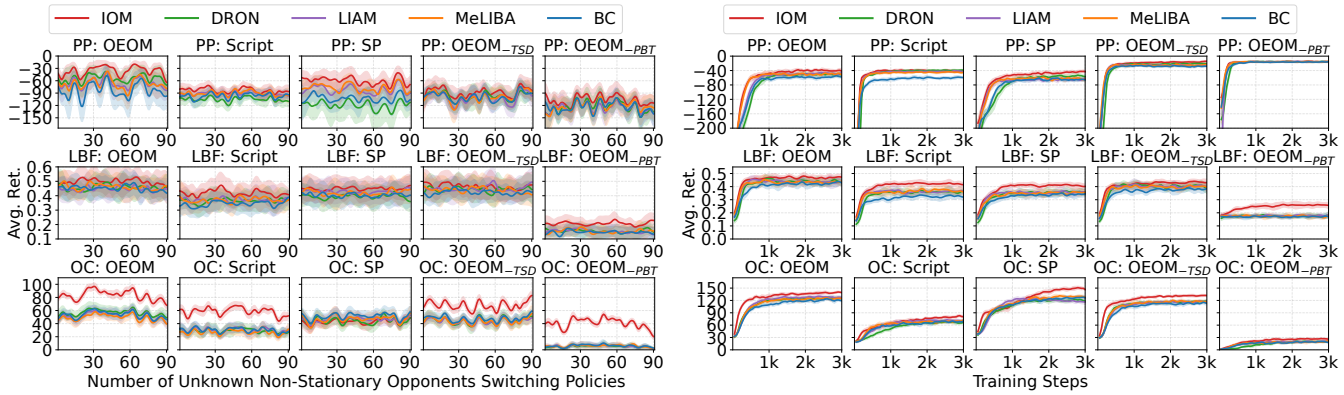

Figure 4: *Left*: Average results against each opponent policy during testing, where Π^{test} adopts the setting of *Unseen-L2*. *Right*: Average performances during training against Π^{train}, where Π^{train} is generated by different OG methods.

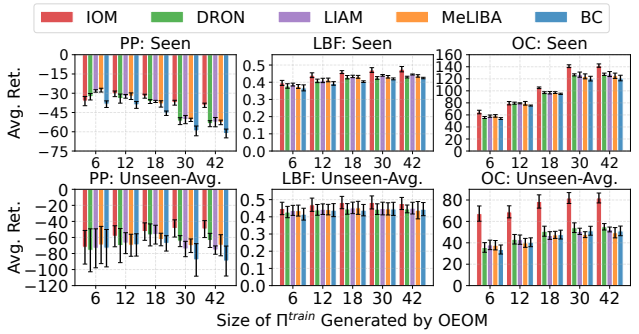

Figure 5: Average results of testing all the OM approaches trained with different sizes of Π^{train} generated by OEOM.

This validates the effectiveness of our perspective that *OM* can be essentially *considered as a Seq2Seq problem*.

Question 3. *Can OM approaches trained with OEOM better adapt to each policy in the unseen* Π^{test}*?*

The *Left* of Fig. 4 shows the results of all OM approaches, trained with Π^{train} generated by different OG methods when tested against each specific opponent policy under *Unseen-L2* setting. These results align with those in Fig. 3 and support similar conclusions: OEOM, compared to other OG methods, better enhances the performance of OM approaches when facing every unseen opponents, thereby improving their generalizabilities. Also, IOM always outperforms other OM baselines, regardless of the OG methods used, further validating the conclusion in **Question 2**.

Question 4. *Can IOM learn how to model opponents more effectively during the OM training stage?*

The *Right* of Fig. 4 shows the average performance curves of all OM approaches during training with Π^{train} generated by different OG methods. Compared to other OM baselines, IOM learns to model various opponents more effectively during training. Combining insights from Fig. 3, we find that other OM approaches, while better than BC during training, often perform worse than BC when facing unseen opponents during testing. In contrast, IOM consistently outperforms BC during both training and testing, indicating that its modeling reliably yields positive results.

Question 5. *How does the size of OEOM's* Π^{train} *(determined by number of iterations* m*) affect OM approaches?*

Fig. 5 provides the test results of all OM approaches trained with different sizes of Π^{train} generated by OEOM. As we are interested in how the sizes of Π^{train} affect the *generalization ability* of OM approaches, we focus on *Unseen-Avg* setting. As the sizes of Π^{train} gradually increase, the performance of OM approaches generally improves. This suggests that OEOM is a robust open-ended system capable of producing opponent models with progressively stronger generalizability. Interestingly, *as the sizes of* Π^{train} *increase*, IOM's performance *steadily improves*, demonstrating good scalability; in contrast, other OM baselines *sometimes perform worse* in PP, indicating they have certain limitations.

6 Discussion

Summary. We propose a novel OEOM framework, which continuously generates opponents with growing strength and style. This framework can train any OM approach, enhancing their generalization ability when adapting to unknown opponents compared to using a fixed set of opponents for training. The foundation of OEOM lies in optimizing policy strength through PBT and policy style through a novel TSD. Building on OEOM, we further propose an ICRL-based OM approach that naturally fits within this framework and can better leverage its advantages. Extensive empirical analysis validates the effectiveness of the OEOM framework and the IOM approach in enhancing generalizability.

Limitations and future work. While OEOM generates a growing set of opponents for training, each policy in this set remains fixed. OM approaches trained using this set may find it difficult to adapt when facing opponents who continuously learn or infer during testing. Generating evolving opponents for OM approaches to enhance their adaptability presents a challenging future research direction.

Acknowledgments

This work is supported in part by the National Science and Technology Major Project (2022ZD0116401); the Natural Science Foundation of China under Grant 62076238, Grant 62222606, and Grant 61902402; the Jiangsu Key Research and Development Plan (No. BE2023016); and the China Computer Federation (CCF)-Tencent Open Fund.

References

Al-Shedivat, M.; Bansal, T.; Burda, Y.; Sutskever, I.; Mordatch, I.; and Abbeel, P. 2018. Continuous Adaptation via Meta-Learning in Nonstationary and Competitive Environments. In *International Conference on Learning Representations*.

Albrecht, S. V.; and Stone, P. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258: 66–95.

Balduzzi, D.; Garnelo, M.; Bachrach, Y.; Czarnecki, W.; Perolat, J.; Jaderberg, M.; and Graepel, T. 2019. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, 434–443. PMLR.

Bauer, J.; Baumli, K.; Behbahani, F.; Bhoopchand, A.; Bradley-Schmieg, N.; Chang, M.; Clay, N.; Collister, A.; Dasagi, V.; Gonzalez, L.; et al. 2023. Human-timescale adaptation in an open-ended task space. In *International Conference on Machine Learning*, 1887–1935. PMLR.

Cohen, A.; Qiao, X.; Yu, L.; Way, E.; and Tong, X. 2019. Diverse exploration via conjugate policies for policy gradient methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3404–3411.

DiGiovanni, A.; and Tewari, A. 2021. Thompson sampling for Markov games with piecewise stationary opponent policies. In *Uncertainty in Artificial Intelligence*, 738–748.

Duan, Y.; Schulman, J.; Chen, X.; Bartlett, P. L.; Sutskever, I.; and Abbeel, P. 2016. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*.

Eysenbach, B.; Ibarz, J.; Gupta, A.; and Levine, S. 2019. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019*.

Foerster, J.; Chen, R. Y.; Al-Shedivat, M.; Whiteson, S.; Abbeel, P.; and Mordatch, I. 2018a. Learning with opponent-learning awareness. In *International Conference on Autonomous Agents and MultiAgent Systems*, 122–130.

Foerster, J.; Farquhar, G.; Al-Shedivat, M.; Rocktäschel, T.; Xing, E.; and Whiteson, S. 2018b. DiCE: The Infinitely Differentiable Monte Carlo Estimator. In *International Conference on Machine Learning*, 1524–1533.

Fu, H.; Tian, Y.; Yu, H.; Liu, W.; Wu, S.; Xiong, J.; Wen, Y.; Li, K.; Xing, J.; Fu, Q.; et al. 2022. Greedy when Sure and Conservative when Uncertain about the Opponents. In *International Conference on Machine Learning*, 6829–6848.

Gangwani, T.; Liu, Q.; and Peng, J. 2018. Learning Self-Imitating Diverse Policies. In *International Conference on Learning Representations*.

Grover, A.; Al-Shedivat, M.; Gupta, J.; Burda, Y.; and Edwards, H. 2018. Learning policy representations in multiagent systems. In *International Conference on Machine Learning*, 1802–1811.

He, H.; Boyd-Graber, J.; Kwok, K.; and Daumé III, H. 2016. Opponent modeling in deep reinforcement learning. In *International Conference on Machine Learning*, 1804–1813.

Jaderberg, M.; Czarnecki, W. M.; Dunning, I.; Marris, L.; Lever, G.; Castaneda, A. G.; Beattie, C.; Rabinowitz, N. C.; Morcos, A. S.; Ruderman, A.; et al. 2019. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 364(6443): 859–865.

Jaderberg, M.; Dalibard, V.; Osindero, S.; Czarnecki, W. M.; Donahue, J.; Razavi, A.; Vinyals, O.; Green, T.; Dunning, I.; Simonyan, K.; et al. 2017. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*.

Jing, Y.; Li, K.; Liu, B.; Zang, Y.; Fu, H.; FU, Q.; Xing, J.; and Cheng, J. 2023. Towards Offline Opponent Modeling with In-context Learning. In *The Twelfth International Conference on Learning Representations*.

Kim, D. K.; Liu, M.; Riemer, M. D.; Sun, C.; Abdulhai, M.; Habibi, G.; Lopez-Cot, S.; Tesauro, G.; and How, J. 2021. A policy gradient algorithm for learning to learn in multiagent reinforcement learning. In *International Conference on Machine Learning*, 5541–5550.

Kumar, S.; Kumar, A.; Levine, S.; and Finn, C. 2020. One solution is not all you need: Few-shot extrapolation via structured maxent rl. *Advances in Neural Information Processing Systems*, 33: 8198–8210.

Laskin, M.; Wang, L.; Oh, J.; Parisotto, E.; Spencer, S.; Steigerwald, R.; Strouse, D.; Hansen, S. S.; Filos, A.; Brooks, E.; Gazeau, M.; Sahni, H.; Singh, S.; and Mnih, V. 2023. In-context Reinforcement Learning with Algorithm Distillation. In *International Conference on Learning Representations*.

Lee, J. N.; Xie, A.; Pacchiano, A.; Chandak, Y.; Finn, C.; Nachum, O.; and Brunskill, E. 2023. Supervised Pretraining Can Learn In-Context Reinforcement Learning. *arXiv preprint arXiv:2306.14892*.

Letcher, A.; Foerster, J.; Balduzzi, D.; Rocktäschel, T.; and Whiteson, S. 2019. Stable Opponent Shaping in Differentiable Games. In *International Conference on Learning Representations*.

Li, Y.; Zhang, S.; Sun, J.; Du, Y.; Wen, Y.; Wang, X.; and Pan, W. 2023. Cooperative open-ended learning framework for zero-shot coordination. In *International Conference on Machine Learning*, 20470–20484. PMLR.

Lin, L.; Bai, Y.; and Mei, S. 2024. Transformers as Decision Makers: Provable In-Context Reinforcement Learning via Supervised Pretraining. In *The Twelfth International Conference on Learning Representations*.

Liu, X.; Jia, H.; Wen, Y.; Hu, Y.; Chen, Y.; Fan, C.; Hu, Z.; and Yang, Y. 2021. Towards unifying behavioral and response diversity for open-ended learning in zero-sum games. *Advances in Neural Information Processing Systems*, 34: 941–952.

Long, W.; Hou, T.; Wei, X.; Yan, S.; Zhai, P.; and Zhang, L. 2023. A Survey on Population-Based Deep Reinforcement Learning. *Mathematics*, 11(10): 2234.

Lu, C.; Willi, T.; De Witt, C. A. S.; and Foerster, J. 2022. Model-free opponent shaping. In *International Conference on Machine Learning*, 14398–14411.

Lupu, A.; Cui, B.; Hu, H.; and Foerster, J. 2021. Trajectory diversity for zero-shot coordination. In *International conference on machine learning*, 7204–7213. PMLR.

Lv, Y.; Yu, Y.; Zheng, Y.; Hao, J.; Wen, Y.; and Yu, Y. 2023. Limited Information Opponent Modeling. In *International Conference on Artificial Neural Networks*, 511–522. Springer.

McAleer, S.; Lanier, J. B.; Fox, R.; and Baldi, P. 2020. Pipeline psro: A scalable approach for finding approximate nash equilibria in large games. *Advances in neural information processing systems*, 33: 20238–20248.

Mishra, N.; Rohaninejad, M.; Chen, X.; and Abbeel, P. 2018. A Simple Neural Attentive Meta-Learner. In *International Conference on Learning Representations*.

Nashed, S.; and Zilberstein, S. 2022. A survey of opponent modeling in adversarial domains. *Journal of Artificial Intelligence Research*, 73: 277–327.

Papoudakis, G.; Christianos, F.; Albrecht, S.; and et al. 2021. Agent modelling under partial observability for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 19210–19222.

Parker-Holder, J.; Pacchiano, A.; Choromanski, K. M.; and Roberts, S. J. 2020. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 18050–18062.

Perez-Nieves, N.; Yang, Y.; Slumbers, O.; Mguni, D. H.; Wen, Y.; and Wang, J. 2021. Modelling behavioural diversity for learning in open-ended games. In *International conference on machine learning*, 8514–8524. PMLR.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Song, Y.; Wang, J.; Lukasiewicz, T.; Xu, Z.; and Xu, M. 2019. Diversity-driven extensible hierarchical reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 4992–4999.

Team, O. E. L.; Stooke, A.; Mahajan, A.; Barros, C.; Deck, C.; Bauer, J.; Sygnowski, J.; Trebacz, M.; Jaderberg, M.; Mathieu, M.; et al. 2021. Open-ended learning leads to generally capable agents. *arXiv preprint arXiv:2107.12808*.

Wang, R.; Lehman, J.; Rawal, A.; Zhi, J.; Li, Y.; Clune, J.; and Stanley, K. 2020. Enhanced poet: Open-ended reinforcement learning through unbounded invention of learning challenges and their solutions. In *International conference on machine learning*, 9940–9951. PMLR.

Wen, Y.; Yang, Y.; Luo, R.; Wang, J.; and Pan, W. 2019. Probabilistic Recursive Reasoning for Multi-Agent Reinforcement Learning. In *International Conference on Learning Representations*.

Willi, T.; Letcher, A. H.; Treutlein, J.; and Foerster, J. 2022. COLA: consistent learning with opponent-learning awareness. In *International Conference on Machine Learning*, 23804–23831.

Wu, Z.; Li, K.; Xu, H.; Zang, Y.; An, B.; and Xing, J. 2022. L2E: Learning to exploit your opponent. In *International Joint Conference on Neural Networks*, 1–8.

Xue, K.; Wang, Y.; Yuan, L.; Guan, C.; Qian, C.; and Yu, Y. 2022. Heterogeneous multi-agent zero-shot coordination by coevolution. *arXiv preprint arXiv:2208.04957*.

Yang, Y.; and Wang, J. 2020. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*.

Yao, J.; Liu, W.; Fu, H.; Yang, Y.; McAleer, S.; Fu, Q.; and Yang, W. 2024. Policy space diversity for non-transitive games. *Advances in Neural Information Processing Systems*, 36.

Yu, X.; Jiang, J.; Zhang, W.; Jiang, H.; and Lu, Z. 2022. Model-based opponent modeling. In *Advances in Neural Information Processing Systems*, 28208–28221.

Zhao, R.; Song, J.; Yuan, Y.; Hu, H.; Gao, Y.; Wu, Y.; Sun, Z.; and Yang, W. 2023. Maximum entropy population-based training for zero-shot human-ai coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 6145–6153.

Zhao, S.; Lu, C.; Grosse, R. B.; and Foerster, J. 2022. Proximal Learning With Opponent-Learning Awareness. In *Advances in Neural Information Processing Systems*, 26324–26336.

Zheng, Y.; Meng, Z.; Hao, J.; Zhang, Z.; Yang, T.; and Fan, C. 2018. A deep bayesian policy reuse approach against non-stationary agents. In *Advances in Neural Information Processing Systems*, 962–972.

Zintgraf, L.; Devlin, S.; Ciosek, K.; Whiteson, S.; and Hofmann, K. 2021. Deep Interactive Bayesian Reinforcement Learning via Meta-Learning. In *International Conference on Autonomous Agents and MultiAgent Systems*, 1712–1714.