Safe adaptation in multiagent competition

Macheng Shen\textsuperscript{1} and Jonathan P. How\textsuperscript{2}

Abstract—Achieving the capability of adapting to ever-changing environments is a critical step towards building fully autonomous robots that operate safely in complicated scenarios. In multiagent competitive scenarios, agents may have to adapt to new opponents with previously unseen behaviors by learning from the interaction experiences between the ego-agent and the opponent. However, this adaptation is susceptible to opponent exploitation. As the ego-agent updates its own behavior to exploit the opponent, its own behavior could become more exploitable as a result of overfitting to this specific opponent’s behavior. To overcome this difficulty, we developed a safe adaptation approach in which the ego-agent is trained against a regularized opponent model, which effectively avoids overfitting and consequently improves the robustness of the ego-agent’s policy. We evaluated our approach in the Mujoco domain with two competing agents. The experiment results suggest that our approach effectively achieves both adaptation to the specific opponent that the ego-agent is interacting with and maintaining low exploitability to other possible opponent exploitation.

I. INTRODUCTION

One critical step towards building fully autonomous intelligent robots is to enable the capability of continual adaptation to new environments. In multiagent scenarios, besides the changing environment dynamics, agents must also adapt to the novel/evolving behaviors of other agents, which may not have been seen during training. There has been a lot of recent progress on fast adaptation to new tasks via meta-learning in both single-agent reinforcement learning (RL) \cite{Mnih2015, Mnih2013, Vinyals2017, Chua2018, Tymofiyenko2019}, and multiagent reinforcement learning (MARL) \cite{Schaul2015, Hessel2018, Houthooft2016}. During the meta-training phase, the agent meta-learns from tasks sampled from a task distribution how to quickly learn from a new task, which enables fast learning during the meta-testing phase. One important assumption within the meta-learning framework is that the task distribution is stationary \cite{Hessel2018}. As a result, the tasks encountered during the meta-testing phase are sampled from the same distribution as those encountered during the meta-training phase. However, in multiagent competitive settings, assuming access to the task distribution (sampling from the unknown opponent agent’s policy distribution) is often unrealistic. Furthermore, as the ego-agent adapts to the opponent, the opponent may also adapt to the ego-agent concurrently, leading to non-stationarity from the ego-agent’s perspective \cite{Houthooft2016, 8Shen2018}. Achieving effective adaptation requires knowledge about the opponent’s learning dynamics \cite{Recht2012, Danner2018}, which is a very strong assumption in competitive scenarios.

In addition to the lack of knowledge about the opponent, another challenge of adaptation in competitive scenarios is to avoid being exploited by the opponent. In two-player zero-sum games, to exploit the opponent, the ego-agent has to deviate from the Nash-Equilibrium \cite{Nash1951}, leading to increased exploitability. Previous works demonstrated search-based safe exploitation in extensive-form games \cite{He2019}, where the ego-agent updates its strategy via subgame-resolving leveraging a model of the opponent’s strategy without substantially increasing the exploitability \cite{He2019}. However, these approaches are specialized for extensive-form games with knowledge of the whole game tree, so it is unclear how to extend these approaches to handle more general multiagent settings, potentially with continuous dynamics, within the model-free reinforcement learning setting.

This paper focuses on safe adaptation in two-player zero-sum scenarios, where the ego-agent needs to update its policy based on the limited amount of interaction experience with an opponent agent. The goal of adaptation is to achieve high competitiveness (measured by cumulative reward) against this specific opponent that the ego-agent is interacting with, while the safety requirement is that the ego-agent must also maintain high competitiveness against any other possible opponent (with either a stationary or evolving policy) during the whole adaptation phase. We investigate this problem under the framework of Markov game \cite{Hennion2015} and multiagent reinforcement learning in a model-free setting without explicit assumptions on the state or action space. As such, the main contributions of this paper are:

1) We present a novel Bayesian formulation of the safe adaptation problem within the MARL framework, which bridges the connection between robust MARL and safe adaptation.

2) We proposed an optimization objective for modeling the opponent, with a behavior cloning term for adaptation and a novel ensemble-regularization term to achieve low exploitability, which is derived from the Bayesian formulation.

3) We demonstrated that our approach achieves adaptation by learning from a limited amount of interaction experience with the opponent while maintaining low exploitability against a second opponent that actively co-adapts to exploit the ego-agent.

*This work is supported by ARL-DCIST (Cooperative Agreement Number W911NF-17-2-0181).
1Macheng Shen is with the Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA macshen@mit.edu
2Jonathan P. How is with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA jhow@mit.edu
II. PRELIMINARIES

A. Markov Games

A Markov game for \( N \) agents is defined by a set of states \( S \), describing the possible configurations of all agents, a set of actions \( A_1, \ldots, A_N \), and a set of observations \( O_1, \ldots, O_N \) for each agent. Each agent has a stochastic policy \( \pi_i : O_i \times A_i \rightarrow [0,1] \), and a reward function \( r_i : S \times A_i \rightarrow \mathbb{R} \).

B. Multiagent reinforcement learning

The objective of each agent is to maximize its own cumulative reward, \( R_i = \sum_{t=0}^T \gamma^t r_i(s_t, a_t) \), with discount factor \( \gamma \) and time horizon \( T \). As a result, the learning problem is formulated as finding a joint policy \( \pi = \{\pi_i\}_{i=1:N} \), where each policy maximizes its own reward,

\[
J_i = \mathbb{E}_{s \sim p^s, a \sim \pi_i, \ldots, a \sim \pi_N} [R_i(s, a)],
\]

with \( p \) being the transition dynamics induced by the joint policy \( \pi \) and the subscript \( -i \) denotes the set \( \{j|j \neq i, j = 1,2,\ldots,N\} \).

C. MARL with policy distribution

Various empirical studies [18]–[24] and a recent theoretical study [25] suggest a more general formulation of MARL in which each agent samples its policy from a policy distribution. Therefore, we consider the following objective function that learns a distribution of policies for each agent,

\[
J_i = \mathbb{E}_{s \sim p^s, a \sim \pi_i, \ldots, a \sim \pi_N} [R_i(s, a)],
\]

where \( \mathcal{P} \) is a joint distribution over the joint policy space \( \Pi = \Pi_1 \times \Pi_2 \ldots \times \Pi_N \). Each agent is learning its own policy distribution \( \Pi_i \) to optimize its objective \( J_i \) subject to the joint distribution \( \Pi \).

Note that the feasibility set of Eq. 2 contains that of Eq. 1, which is analogous to the relationship between a mixed-strategy Nash Equilibrium and a pure-strategy Nash Equilibrium [26]. This relationship also suggests that Eq. 2 is a more appropriate learning objective than Eq. 1.

III. OUR APPROACH

As we focus on two-player zero-sum games, we rewrite Eq. 2 from the ego-agent’s perspective as:

\[
J_{ego} = \mathbb{E}_{x_{ego} \sim p(\Pi_{ego}), \pi_{ego} \sim p(\Pi_{ego})} [\mathbb{E}_{s \sim p^s, a \sim \pi} [R_{ego}(s, a)]] ,
\]

where the ego-agent optimizes its policy distribution \( \Pi_{ego} \), subject to a given opponent policy distribution \( \Pi_{oppo} \). Since the optimal \( \Pi_{ego} \) with respect to Eq. 3 depends on the opponent policy distribution \( \Pi_{oppo} \), determining \( \Pi_{oppo} \) is critical. Here, we discuss some common choices for the opponent policy distribution \( \Pi_{oppo} \):

1) Oracle policy distribution: Suppose we know the true policy distribution of the opponent, we can optimize the ego-agent policy distribution against the opponent policy distribution to obtain \( \Pi_{ego}^{oracle} \). However, there are two problems with this approach: 1) Feasibility: In competitive scenarios, it is unlikely to get access to the policy of the opponent. 2) Robustness: As the ego-agent over-fits its policy to the opponent, the resulting \( \Pi_{ego}^{oracle} \) may not perform well (or even poorly as shown in [27]) against an adversarial opponent that is trained against \( \Pi_{ego}^{oracle} \) to exploit its weakness.

2) Learned opponent policy distribution [28]–[31]: As the ego-agent interacts with the opponent, the ego-agent can learn an internal model of the opponent policy \( \Pi_{oppo}^{model} \) distribution from the interaction experience as an approximation of the true opponent policy distribution. Many previous works [28]–[30], assume access to the opponent’s observation and action for this model learning, which is not a strong assumption in robotic domains with full-observability over the state space since the opponent’s observation and action can be deduced from the state observation. Besides, [31] also demonstrates the possibility of learning an opponent model from the ego-agent’s observation alone via variational inference over a hidden space that models the opponent’s private information. However, this opponent modeling approach also suffers from the robustness problem mentioned earlier.

3) Nash Equilibrium policy distribution: Another way to model the opponent is to solve for the Nash Equilibrium policy distribution \( \Pi_{oppo}^{nash} \) (or equivalently, the minimax solution [32] in two-player zero-sum games [26]), with no prior knowledge of the opponent’s policy distribution. The corresponding optimal policy distribution for the ego-agent is also the Nash Equilibrium \( \Pi_{ego}^{nash} \), which is the least exploitable policy. However, this approach does not attempt to adapt to the opponent that the ego-agent is interacting with, leading to sub-optimal performance against an opponent with exploitability.

We argue that a better approach to modeling the opponent’s policy distribution should leverage the available interaction experience for adaptation as well as stay close to the equilibrium distribution for robustness against adversarial exploitation. We derive this approach by rephrasing the sub-problem within Eq. 3 of modeling the opponent policy distribution \( \pi_{oppo} \sim p(\Pi_{oppo}) \) into a Bayesian inference problem over the space of policy distributions given the interaction experience \( D \) between the ego-agent and the opponent:

\[
\pi_{oppo} \sim p(\Pi_{oppo}|D) \propto p(\Pi_{oppo}|\emptyset) \times p(D|\Pi_{oppo}) ,
\]

where \( \emptyset \) denotes the empty set, so \( p_{prior}(\Pi_{oppo}) = p(\Pi_{oppo}|\emptyset) \) is the prior distribution over the opponent’s policy space before obtaining any interaction experience. We argue that the Nash Equilibrium policy distribution is a sensible choice for this prior, i.e. \( p_{prior}(\Pi_{oppo}) = p^{NE}(\Pi_{oppo}) \), since with no information about the opponent, the best choice is to minimize the ego-agent’s exploitability. As the ego-agent receives more interaction experience with the opponent, the posterior distribution \( p(\Pi_{oppo}|D) \) is updated...
through the likelihood term $p(D|\Pi_{oppo})$ while regularized by the prior term, which ensures adaptation to the opponent while maintaining low exploitability.

However, the posterior inference problem Eq. 4 is challenging for two reasons: 1) Solving for the Nash Equilibrium policy distribution $\Pi^{\text{NE}}$ ($\Pi_{oppo}$) is a challenging problem; 2) representing and parameterizing policy distribution is challenging. Therefore, we apply the following two approximations,

1) We approximate the Nash Equilibrium policy distribution via an ensemble of policies generated via Alg. 1, which has been shown in [18], [23], [27] to produce robust agent behaviors that are much less exploitable than policies generated without ensembling.

2) Instead of modeling the posterior distribution, we seek for a single opponent policy that approximates the maximum a posteriori probability (MAP) estimate of Eq. 4.

With these two approximations, we propose an alternative formulation for the estimated opponent policy as the optimization problem in Eq. 5.

$$\hat{\pi}_{oppo} = \arg\min_{\pi}\mathbb{D}(\Pi_{oppo}^\text{ensemble} | \pi) + \lambda_1 L_{\text{likelihood}}(D | \pi) + \lambda_2 L_{\text{RL}}(\Pi_{ego}^\text{ensemble}, \pi),$$

(5)

where $\lambda_1$ and $\lambda_2$ are hyper-parameters. The first term $\mathbb{D}(\Pi_{oppo}^\text{ensemble} | \pi)$ denotes a distance metric between the opponent policy ensemble generated via Alg. 1 and the estimated opponent model, which regularizes the opponent policy to stay close to the robust ensemble policy distribution. This term corresponds to the prior term $p^{\text{NE}}(\Pi_{oppo})$ in Eq. 4. The second term $L_{\text{likelihood}}(D | \pi)$ is the likelihood of observing the interaction experience $D$ given the opponent policy, which corresponds to the likelihood term $p(D|\Pi_{oppo})$ in Eq. 4. The third term $L_{\text{RL}}(\Pi_{ego}^\text{ensemble}, \pi)$ is the reinforcement learning loss that optimizes the opponent policy against the ego-agent’s policy ensemble.

$$L_{RL}(\Pi_{ego}^\text{ensemble}, \pi) = -\mathbb{E}_{\pi_{ego} \sim \Pi_{ego}^\text{ensemble}, \pi_{oppo} \sim \pi_{oppo}}[\mathbb{E}_{s \sim p^\pi, a \sim \pi}[R_{oppo}(s, a)]],$$

(6)

where the minus sign ensures that minimizing this loss results in maximization of the reward. This term does not correspond to any term in Eq. 4. Intuitively, this term enables the opponent policy to continually evolve, as we update the ego-agent’s policy ensemble to adapt to the opponent via optimizing its reinforcement learning objective in Eq. 7.

$$J_{\text{ego}} = \mathbb{E}_{\pi_{ego} \sim \Pi_{ego}^\text{ensemble}, \pi_{oppo} = \pi_{oppo}}[\mathbb{E}_{s \sim p^\pi, a \sim \pi}[R_{\text{ego}}(s, a)]],$$

(7)

We include $L_{\text{RL}}$ because we found that although this term does not make much difference when adapting to a stationary opponent, it could be critical for achieving high competitiveness against an evolving opponent as we show later in the experiment section.

Now we discuss our choice for the first two terms in Eq. 5. The first term $\mathbb{D}(\Pi_{oppo}^\text{ensemble} | \pi)$ measures the discrepancy between the policy ensemble $\Pi_{oppo}^\text{ensemble}$ and our estimated opponent policy $\pi$. There are several closed-form metrics to measure the discrepancy between two policies, including KL-divergence discrepancy [33], total variation distance [34] and maximum mean discrepancy [35]. However, it is unclear how to select one metric over another given a specific application domain, and whether the selected metric can optimally discriminate between two policies. To resolve this ambiguity and achieving optimal discriminative power, we choose to learn the discrepancy metric via adversarial learning following the paradigm of generative adversarial imitation learning (GAIL) [36], where we train a discriminator $D_w(o, a): \mathcal{O}_{oppo} \times \mathcal{A}_{oppo} \rightarrow [0, 1]$ to minimize the following discrimination loss,

$$\mathbb{E}_{\tau_{\text{imit}}}[\log D_w(o, a)] + \mathbb{E}_{\tau_{\Pi_{oppo}^\text{ensemble}}}[\log(1 - D_w(o, a))],$$

(8)

such that

$$\mathbb{D}(\Pi_{oppo}^\text{ensemble} | \pi) = -\mathbb{E}_{\tau_{\text{imit}}}[\log D_w(o, a)],$$

(9)

where the shorthand notation $\tau_{\text{imit}}$ denotes the trajectory distribution when the ego-agent follows policy $\pi_{ego} \sim p(\Pi_{ego})$, while the opponent follows policy $\cdot$. This loss function is minimized by maximizing an imitation reward $r_{\text{imit}} = \log D_w(o, a)$.

The second term in Eq. 5 is the log-likelihood of observing the experience $D$ given the opponent policy $\pi$. Since the opponent policy can only affect the probability of the

---

**Algorithm 1 Ensemble training**

**Require:** Ensemble size $N$, number of training iterations $K$

1: Randomly initialize policy ensembles: $\Pi_{ensemble}^\text{ensemble} = \{\pi_{ego}^i\}_{i=1:N}$, $\Pi_{oppo}^\text{ensemble} = \{\pi_{oppo}^i\}_{i=1:N}$

2: for $k = 1:K$

3: Randomly sample policy index: $j \sim \{1, \ldots, N\}, l \sim \{1, \ldots, N\}$

4: Environment_rollout($\pi_{ego}^j, \pi_{oppo}^l$)

5: Update $\pi_{ego}^j$ and $\pi_{oppo}^l$ to optimize objective Eq. 2

6: end for

7: return $\Pi_{ego}^\text{ensemble}$ and $\Pi_{oppo}^\text{ensemble}$

---

**Algorithm 2 Safe adaptation**

**Require:** Policy ensembles $\Pi_{ensemble}^\text{ensemble}$ and $\Pi_{ensemble}^\text{oppo}$, interaction experience $D$, number of iterations $K$

1: freeze $\Pi_{oppo}^\text{ensemble}$

2: Initialize $\hat{\pi}_{oppo}^0$ as random policy

3: $\Pi_{ensemble}^\text{oppo} \leftarrow \Pi_{oppo}^\text{ensemble}$

4: for $k = 1:K$

5: $\hat{\pi}_{oppo}^k \leftarrow \text{update_opponent}(\hat{\pi}_{oppo}^{k-1}, \Pi_{oppo}^\text{ensemble})$

6: $\Pi_{ego}^\text{ensemble}, k \leftarrow \text{update_ego_agent}(\hat{\pi}_{ego}^k, \Pi_{ego}^{\text{ensemble}, k-1})$

7: $\text{one gradient step of Eq. 5}$

8: end for

9: return $\Pi_{ego}^\text{ensemble}, K$
Fig. 1: Mujoco environments (left: swimmer; right: ant), where the blue part of the body is controlled by the ego-agent and the red part of the body is controlled by the opponent agent. The white circles are the joints where agents can apply torques on. The ego-agent is rewarded for moving forward while the opponent agent is rewarded for moving backward.

Formally, given an opponent’s taken action, this term can be reduced to behavior cloning loss,

$$p(D | \pi) = \sum_{(a, o) \in D} \log \pi(a | o).$$  \hspace{1cm} (10)

In practice, we use mini-batch to calculate the gradient of this loss.

IV. RESULTS AND DISCUSSION

A. Experiment setting

We evaluate our safe adaptation approach on the Multiagent Mujoco domain [37], where each robot is decomposed into parts that are controlled by individual agents as illustrated in Fig. 1. We use a zero-sum reward where the ego-agent tries to maximize the reward for moving forward and the opponent tries to minimize this reward.

To evaluate the capability of safe adaptation to a previously unseen opponent, we describe the following procedure to set up the evaluation:

**Off-line training phase:**

1) Alternating for $K_1$ iterations, between one-step ($K_1 = 1$) policy ensemble training of size $N = 5$ for both agents, $\Pi_{\text{ensemble}}$ via Alg. 1 and one-step training of exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$ via Alg. 3.

2) Freeze $\Pi_{\text{ensemble}}$ via Alg. 1 and one-step training of exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$ via Alg. 3.

3) Collect interaction experience $D$ between $\Pi_{\text{ensemble}}$ via Alg. 1 and one-step training of exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$.

4) Unfreeze $\Pi_{\text{ensemble}}$.

5) Alternating for $K_2$ iterations, between one-step exploitation for $\Pi_{\text{ensemble}}$ and one-step training of exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$.

6) Return $\pi^{\text{exp}}_{\text{oppo}}$.

During the off-line training phase, we alternate between ensemble training $\Pi_{\text{ensemble}}$ and $\pi^{\text{exp}}_{\text{oppo}}$ to mitigate the well-known problem of training imbalance [23], [38] in competitive/adversarial training, so that the exploiter opponent can always catch up with the ego-agent before the ego-agent becomes too strong. The additional training of the exploiter opponent ensures that the exploiter is sufficiently trained to exploit the ego-agent, which motivates the ego-agent to adapt to this exploiter in the adaptation phase. During the whole off-line training phase, the ego-agent does not collect experience against the exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$. As a result, this exploiter is a previously unseen opponent in the ego-agent’s perspective.

During the on-line adaptation phase, the ego-agent policy ensemble $\Pi_{\text{ensemble}}$ adapts to the exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$ given a fixed size interaction experience $D$, with regularization from $\Pi_{\text{ensemble}}$. Concurrently, the second exploiter $\pi^{\text{exp}}_{\text{oppo}}$ is trained to exploit the $\Pi_{\text{ensemble}}$. As a result, the reward against the first exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$ during the adaptation phase measures the capability of adaptation against a stationary opponent, while the reward against the second exploiter opponent $\pi^{\text{exp}}_{\text{oppo}}$ measures robustness/safety against an evolving adversarial exploiter.

We use stochastic policy with Gaussian distribution, and two fully-connected hidden layers, each with 128 hidden units followed by ReLU activate layer, as the policy and critic network architecture. We use a pytorch implementation\(^1\) of Soft Actor-Critic [39] (SAC) with dual critic networks and automatic tuning of the entropy parameter to train the ensemble networks and the two exploiter opponents in both the off-line training phase and the on-line adaptation phase. We use a replay buffer size of one million for the off-line training phase but reduce that to half a million for the on-line adaptation phase. We use a learning rate of 0.001 for both training and adaptation. In both environments, the number of steps per episode is fixed at 500. Each agent can observe the joints/bodies position and

\(^1\)https://github.com/p-christ/Deep-Reinforcement-Learning-Algorithms-with-PyTorch
\(^2\)https://github.com/Khrylx/PyTorch-RL

![Algorithm 3 Train exploiter opponent](https://github.com/Khrylx/PyTorch-RL)

**Algorithm 3 Train exploiter opponent**

**Require:** Ego-agent ensemble $\Pi_{\text{ensemble}}$, exploiter opponent policy $\pi^{\text{exp}}_{\text{oppo}}$, number of training iterations $K$

1: freeze $\Pi_{\text{ego}}$
2: for $k = 1:K$ do
3: Train $\pi^{\text{exp}}_{\text{oppo}}$ against $\Pi_{\text{ego}}$ by gradient decent on $L_{\text{RL}}(\Pi_{\text{ego}}, \pi^{\text{exp}}_{\text{oppo}})$
4: end for
5: Unfreeze $\Pi_{\text{ensemble}}$
6: return $\pi^{\text{exp}}_{\text{oppo}}$
velocity of its own and its opponent's, so the agents have full observability.

In the off-line training phase, we use $K_1 = 10000$ iterations (episodes), and $K_2 = 5000$ iterations the swimmer environment and $K_2 = 2000$ for the ant environment. In the on-line adaptation phase, we use different $K_3$ for the two environments until the average rewards become steady. We collect 10 episodes of interaction experience for adaptation, which corresponds to $|\mathcal{D}| = 5000$ environment steps. We tune the hyper-parameters $\lambda_1$ and $\lambda_2$ independently for each environment. We select from the following values: $\{0.1, 0.5, 1.0, 5.0\}$, and manually choose the best one by looking at the adaptation rewards against both of the two exploiter opponents. The selected hyper-parameters are: $\lambda_1 = 1.0$, $\lambda_2 = 1.0$ for the swimmer environment, and $\lambda_1 = 0.1$, $\lambda_2 = 1.0$ for the ant environment.

**B. Results**

We show the exploiter opponent rewards during the adaptation phase in Fig. 2, which includes the following settings: 1) oracle opponent, where the ego-agent is trained against the first exploiter directly. This setting is unrealistic since the first exploiter's policy is unknown to the ego-agent; 2) ensemble opponent, where the ego-agent is trained against the opponent ensemble policy generated from the off-line training phase; 3) reg_bc_rl: our proposed approach including all the three terms in Eq. 5; 4) bc_rl: ablation of our approach without the behavior cloning term; 5) reg_rl: ablation of our approach without the ensemble regularization term; 6) reg_bc: ablation of our approach without the RL term.

Fig. 2a shows that the oracle opponent setting (reference) achieves the best adaptation against the first exploiter opponent, while all the settings with the behavior cloning loss achieve comparable adaptation performance as the reference. For all the settings, the opponent reward decreases due to
the fact that the opponent policy is fixed but the ego-agent policy is updating. However, there remains a gap between those settings without interaction experience with the first exploiter opponent (ensemble opponent and reg_{rl}) and the other settings.

Fig. 2b shows that those settings without the ensemble regularization term (oracle opponent and bc_{rl}) are unable to achieve robustness against the second exploiter opponent which actively exploits the ego-agent’s policy. Combining Fig. 2a and 2b, we can conclude that our approach (reg_{bc_{rl}}) strikes a better trade-off between adaptation and safety, compared with the two reference approaches (oracle opponent: good adaptation but poor robustness; ensemble opponent: good robustness but poor adaptation).

Fig. 2c and 2d show consistent results as those in Fig. 2a and 2b: the ego-agent’s policies that are adapting to the first exploiter opponent are also more susceptible to be exploited by the second exploiter opponent. Besides, Fig. 2d shows that the setting without the RL loss (reg_{bc}) is also exploited after the second exploiter is sufficiently trained. Our conjectured reason for this observation is that the RL term enables the estimated opponent policy \( \hat{\pi}_{opp} \) to discover the weakness of the ego-agent’s policy, which also helps reduce the ego-agent’s exploitability because the ego-agent is trained against the estimated opponent policy.

To quantitatively measure the adaptation and robustness of different settings, we calculate the area under curve (AUC) metrics of both the first exploiter’s (measures adaptation) and the second exploiter’s (measures robustness) reward curves normalized by the two reference settings (oracle opponent and ensemble opponent), as shown in Table I. We calculate the AUC using reward curves from steps 1e7 to 2e7 for the swimmer environment and from steps 1e7 to 3.5e7 for the ant environment. These results are consistent with our hypothesis that learning from the interaction experience with the opponent enables adaptation, but without regularization from the ensemble policy, this adaptation could be highly exploitable. The regularization term is effective for achieving safe adaptation. As a result, our approach achieves the best overall metric which combines adaptation and robustness.

From Table I, we can also see that the adaptation metric and the robustness metric tend to be negatively correlated. To further analyze the relationship between adaptation (exploitation) and robustness (exploitability), we show the normalized area between curves (ABC) in Table II, which is the gap between the reward against the second exploiter opponent and the first exploiter opponent. Lower ABC indicates less sensitivity to opponent exploitation. This result, together with the result shown in Table I, verifies the well-known trade-off between exploitation and exploitability [13]: the settings with both the ensemble regularization term and the behavior cloning terms (reg_{bc_{rl}} and reg_{bc}) are slightly more exploitable than their counterpart without the behavior cloning term (reg_{rl}), which is an inevitable consequence of exploiting the interaction experience against the first exploiter opponent.

V. CONCLUSIONS

This paper investigates safe adaptation which is an important problem in competitive MARL. In contrast to the widely-studied fast adaptation problem, our focus is on maintaining low exploitability during the adaptation. Our key innovation is the derivation of a novel ensemble regularization term from a Bayesian formulation of the MARL objective function. We show empirically that our proposed approach is effective both at adaptation to a previously unseen opponent given experience from a few interaction episodes and at maintaining low exploitability against an adversarial opponent that actively exploits the weakness of the ego-agent. Our ablation study and analysis reveal the effect of each term in our proposed loss function, as well as verify the well-known trade-off between exploitation and exploitability. Our work contributes to an important step towards building reliable intelligent robots that are able to operate safely in competitive multiagent scenarios against ever-changing adversarial opponents.

| Settings/Metrics | Adaptation | Robustness | Overall (A+R) |
|------------------|------------|------------|---------------|
| Oracle opponent  | 1.0        | 0.0        | 1.0           |
| Ensemble opponent| 0.0        | 1.0        | 1.0           |
| Reg_{bc_{rl}}    | 0.50       | 1.01       | 1.51          |
| Bc_{rl}          | 0.67       | -0.92      | -0.25         |
| Reg_{rl}         | -0.16      | 0.69       | 0.53          |
| Reg_{bc}         | 0.52       | 1.07       | 1.59          |

| Settings/Metrics | Adaptation | Robustness | Overall (A+R) |
|------------------|------------|------------|---------------|
| Oracle opponent  | 1.0        | 0.0        | 1.0           |
| Ensemble opponent| 0.0        | 1.0        | 1.0           |
| Reg_{bc_{rl}}    | 0.60       | 0.79       | 1.39          |
| Bc_{rl}          | 0.76       | -0.60      | 0.16          |
| Reg_{rl}         | 0.06       | 0.89       | 0.95          |
| Reg_{bc}         | 0.48       | 0.25       | 0.73          |

| Settings | Normalized ABC |
|----------|----------------|
| Oracle opponent | 1.0 |
| Ensemble opponent | 0.0 |
| Reg_{bc_{rl}} | 0.27 |
| Bc_{rl} | 1.23 |
| Reg_{rl} | 0.05 |
| Reg_{bc} | 0.25 |

| Settings | Normalized ABC |
|----------|----------------|
| Oracle opponent | 1.0 |
| Ensemble opponent | 0.0 |
| Reg_{bc_{rl}} | 0.37 |
| Bc_{rl} | 1.07 |
| Reg_{rl} | 0.11 |
| Reg_{bc} | 0.58 |

TABLE I: Adaptation and robustness metrics of different settings, where the oracle opponent and the ensemble opponent settings are taken as references. The best two settings are highlighted.

TABLE II: Normalized ABC (area between curves) between the second exploiter reward and the first exploiter reward. Lower ABC score indicates that the ego-agent’s policy is less sensitive to opponent exploitation.
REFERENCES

[1] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, “RL2: Fast reinforcement learning via slow reinforcement learning,” arXiv preprint arXiv:1611.02779, 2016.
[2] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in International conference on machine learning. PMLR, 2017, pp. 1126–1135.
[3] A. Nagabandi, C. Finn, and S. Levine, “Deep online learning via meta-learning: Continual adaptation for model-based rl,” arXiv preprint arXiv:1812.07671, 2018.
[4] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn, “Learning to adapt in dynamic, real-world environments through meta-reinforcement learning,” arXiv preprint arXiv:1803.11347, 2018.
[5] A. Nichol and J. Schulman, “Reptile: a scalable metalearning algorithm,” arXiv preprint arXiv:1803.02999, vol. 2, no. 3, p. 4, 2018.
[6] M. Al-Shedivat, T. Bansal, Y. Burda, I. Sutskever, I. Mordatch, and P. Abbeel, “Continuous adaptation via meta-learning in nonstationary and competitive environments,” arXiv preprint arXiv:1710.03641, 2017.
[7] J. N. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, and I. Mordatch, “Learning with opponent-learning awareness,” arXiv preprint arXiv:1709.04326, 2017.
[8] D. K. Kim, M. Liu, M. D. Riemer, C. Sun, M. Abdulkhai, G. Habibi, S. Lopez-Cot, G. Tesauro, and J. How, “A policy gradient algorithm for learning to adapt in multiagent reinforcement learning,” in International Conference on Machine Learning. PMLR, 2021, pp. 5541–5550.
[9] K. Khetarpal, M. Riemer, I. Rish, and D. Precup, “Towards continual reinforcement learning: A review and perspectives,” arXiv preprint arXiv:2012.13490, 2020.
[10] —, “Towards continual reinforcement learning: A review and perspectives,” arXiv preprint arXiv:2012.13490, 2020.
[11] C. A. Holt and A. E. Roth, “The nash equilibrium: A perspective,” Proceedings of the National Academy of Sciences, vol. 101, no. 12, pp. 3999–4002, 2004.
[12] R. Cressman, C. Ansell, and K. Binmore, Evolutionary dynamics and extensive form games. MIT Press, 2003, vol. 5.
[13] S. Ganzfried and T. Sandholm, “Safe opponent exploitation,” ACM Transactions on Economics and Computation (TEAC), vol. 3, no. 1, pp. 1–28, 2015.
[14] M. Liu, C. Wu, Q. Liu, Y. Jing, J. Yang, P. Tang, and C. Zhang, “Safe opponent-exploitation subgame refinement,” 2022. [Online]. Available: https://openreview.net/forum?id=VwSHZgruNEc.
[15] N. Brown and T. Sandholm, “Safe and nested subgame solving for imperfect-information games,” Advances in neural information processing systems, vol. 30, 2017.
[16] —, “Safe and nested subgame solving for imperfect-information games,” Advances in neural information processing systems, vol. 30, 2017.
[17] M. L. Littman, “Markov games as a framework for multi-agent reinforcement learning,” in Machine learning proceedings 1994. Elsevier, 1994, pp. 157–163.
[18] R. Lowe, Y. I. Wu, A. Tamari, J. Harb, O. Pieter Abbeel, and I. Mordatch, “Multi-agent actor-critic for mixed cooperative-competitive environments,” Advances in neural information processing systems, vol. 30, 2017.
[19] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., “Mastering the game of go with deep neural networks and tree search,” nature, vol. 529, no. 7587, pp. 484–489, 2016.
[20] M. Jaderberg, W. Czarnecki, I. Dunning, L. Marris, G. Lever, A. Cataneda, B. Beattie, N. Rabinowitz, A. Morcos, A. Ruderman, et al., “Human-level performance in 3D multiplayer games with population-based reinforcement learning,” Science, vol. 364, no. 6443, pp. 859–865, 2019.
[21] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chang, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., “Grandmaster level in starcraft ii using multi-agent reinforcement learning,” Nature, vol. 575, no. 7782, pp. 350–354, 2019.
[22] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolet, D. Silver, and T. Graepel, “A unified game-theoretic approach to multiagent reinforcement learning,” arXiv preprint arXiv:1711.00032, 2017.
[23] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, “Emergent complexity via multi-agent competition,” arXiv preprint arXiv:1710.03748, 2017.
[24] E. Vintsyuk, Y. Du, K. Parvate, K. Jang, P. Abbeel, and A. Bayen, “Robust reinforcement learning using adversarial populations,” arXiv preprint arXiv:2008.01825, 2020.
[25] W. M. Czarnecki, G. Gidel, B. Tracey, K. Tuyls, S. Omidsfahani, D. Balduzzi, and M. Jaderberg, “Real world games look like spinning tops,” arXiv preprint arXiv:2004.09468, 2020.
[26] J. F. Nash et al., “Equilibrium points in n-person games,” Proceedings of the national academy of sciences, vol. 56, no. 1, pp. 586–591, 1950.
[27] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell, “Adversarial policies: Attacking deep reinforcement learning,” arXiv preprint arXiv:1905.10615, 2019.
[28] H. He, J. Boyd-Graber, K. Kwok, and H. Daumé III, “Opponent modeling in deep reinforcement learning,” in International conference on machine learning. PMLR, 2016, pp. 1804–1813.
[29] B. Raileanu, E. Denton, A. Szlam, and R. Fergus, “Modeling others using oneself in multi-agent reinforcement learning,” in International conference on machine learning. PMLR, 2018, pp. 4257–4266.
[30] A. Grover, M. Al-Shedivat, J. Gupta, Y. Burda, and H. Edwards, “Learning policy representations in multiagent systems,” in International conference on machine learning. PMLR, 2018, pp. 1802–1811.
[31] G. Papoudakis and S. V. Albrecht, “Variational autoencoders for opponent modeling in multi-agent systems,” arXiv preprint arXiv:2001.10829, 2020.
[32] S. Li, Y. Wu, X. Cui, H. Dong, F. Fang, and S. Russell, “Robust multi-agent reinforcement learning via minmax deep deterministic policy gradient,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, 2019, pp. 4213–4220.
[33] C. Zhang, Y. Li, and J. Li, “Policy search by target distribution learning for continuous control,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 04, 2020, pp. 6770–6777.
[34] J. Queeney, I. Paschalidis, and C. Cassandras, “Generalized proximal policy optimization with sample reuse,” Advances in Neural Information Processing Systems, vol. 34, 2021.
[35] M. A. Masood and F. Doshi-Velez, “Diversity-inducing policy gradient: Using maximum mean discrepancy to find a set of diverse policies,” arXiv preprint arXiv:1906.00088, 2019.
[36] J. Ho and S. Ermon, “Generative adversarial imitation learning,” Advances in neural information processing systems, vol. 29, 2016.
[37] B. Peng, T. Rashid, C. Schroder de Witt, P.-A. Kamienne, P. Torr, W. Böhmer, and S. Whiteson, “Facmac: Factored multi-agent centralised policy gradients,” Advances in Neural Information Processing Systems, vol. 34, 2021.
[38] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” Advances in neural information processing systems, vol. 27, 2014.
[39] T. Haarnoja, A. Zhou, P. Abbeel, and 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.
[40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.