Limited Information Opponent Modeling

Yongliang Lv [*1] **Yan Zheng** [*1] **Jianye Hao** [*1]

Abstract

The goal of opponent modeling is to model the opponent policy to maximize the reward of the main agent. Most prior works fail to effectively handle scenarios where opponent information is limited. To this end, we propose a Limited Information Opponent Modeling (LIOM) approach that extracts opponent policy representations across episodes using only self-observations. LIOM introduces a novel policy-based data augmentation method that extracts opponent policy representations offline via contrastive learning and incorporates them as additional inputs for training a general response policy. During online testing, LIOM dynamically responds to opponent policies by extracting opponent policy representations from recent historical trajectory data and combining them with the general policy. Moreover, LIOM ensures a lower bound on expected rewards through a balance between conservative and exploitation. Experimental results demonstrate that LIOM is able to accurately extract opponent policy representations even when the opponent's information is limited, and has a certain degree of generalization ability for unknown policies, outperforming existing opponent modeling algorithms.

1. Introduction

Opponent Modeling (He et al., 2016; Hong et al., 2017; Zheng et al., 2018; Foerster et al., 2017) is an important branch of Multi-Agent Reinforcement Learning aimed at utilizing opponent information to model the opponent in order to maximize self (i.e., the main agent's) reward, particularly when the opponent policy is non-stationary. However, existing opponent modeling approaches heavily rely on the completeness of opponent information, rendering them limited in accuracy and effectiveness when opponent observations and actions are unknown. To address this issue, we propose the Limited Information Opponent Modeling (LIOM) algorithm, which models the opponent policy solely based on the main agent's historical observations. We divide the problem into two stages: offline training and online testing.

During the offline training stage, we aim to learn a powerful general policy on a given set of opponent policies, which can approximate the optimal response policy of any known or unknown opponent as closely as possible. This requires an accurate and generalizable representation of opponent policies as support. To this end, we propose a novel policy-based data augmentation approach that interacts with opponent policies with various augmentation policies to generate trajectories for constructing positive and negative samples. We then extract cross-episode trajectory representations self-supervisedly via contrastive learning as opponent policy representations. Then, the opponent policy representations are further inputted into the reinforcement learning algorithm for learning the general policy. The advantage of the general policy lies in its ability to generalize to an infinite number of opponent policies, implying no need for relearning against new individual opponent policies during online testing.

During the online testing stage, we encode the recent historical trajectories into opponent policy representations and use them as additional inputs to the general policy. This dynamic response allows us to handle non-stationary opponent policies. Furthermore, to deal with difficult-to-generalize opponent policies, we dynamically update a weight based on the reward, which selects between a conservative Nash equilibrium policy and an exploitative general policy. This ensures the lower bound of expected returns.

This paper presents two innovative contributions: (1) opponent modeling is performed only based on self-observation, which renders our approach adaptable to almost any environment. (2) a novel policy-based data augmentation technique is proposed, which allows for independent extraction of policy representations without being influenced by policies it interacts with. Moreover, we introduce a classic algorithm EXP3 to address the trade-off between conservative and

[*]Equal contribution [1]College of Intelligence and Computing, Tianjin University, Tianjin, China. Correspondence to: Jianye Hao <jianye.hao@tju.edu.cn>, Yongliang Lv <lvyongliang@tju.edu.cn>.

Workshop on New Frontiers in Learning, Control, and Dynamical Systems at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023.

exploitation during online testing.

We compared the performance of LIOM with multiple algorithms in two classic reinforcement learning benchmarks, Kuhn Poker and Soccer. Our results demonstrate that LIOM outperforms existing opponent modeling algorithms in terms of performance against both "seen" and "unseen" opponents.

2. Related Work

2.1. Opponent Modeling

Early opponent modeling research primarily focused on simple environments with fixed opponent policies. With the introduction of non-stationary environments, existing opponent modeling can be categorized into two approaches: implicit modeling and explicit modeling.

Implicit opponent modeling refers to extracting opponent information for representation learning during training. He et al. (2016) proposed an end-to-end training approach by merging the opponent's observation with the agent's observation using a deep neural network. Hong et al. (2017) further incorporated opponent action information and fitted the opponent policy through a neural network. Considering that the opponent may also have learning behaviors, Foerster et al. (2017) leveraged recurrent reasoning to estimate the parameters of the opponent policy network and maximize the agent's reward. Raileanu et al. (2018) took a different perspective, considering the opponent policy network parameters of the agent's own policy and using opponent observations to make decisions.

Explicit opponent modeling refers to the explicit modeling of opponent policies, dividing opponent types, and online detection and response during the interaction process. Rosman et al. (2016) first proposed bayesian policy reuse for multi-task learning, while Hernandez-Leal et al. (2016) extended this to multi-agent systems by using MDPs to model opponents and adding a detection mechanism for unknown opponent policies. In more complex environments, Zheng et al. (2018) used neural networks to model opponents and introduced a rectified belief model to improve opponent detection accuracy and speed. Building on this work, Yang et al. (2018) introduced the Theory of Mind approach to defeat opponents using higher-level decision-making methods in cases where the opponent is also using an opponent modeling method.

2.2. Contrastive Learning

As the most prevalent self-supervised learning algorithm in recent years, contrastive learning aims to learn common features among similar instances while distinguishing dissimilar instances. Oord et al. (2018) initially proposed InfoNCE loss, which encodes time-series data. By segregating positive and negative samples, it can extract data-specific representations. Following this approach, He et al. (2020) achieved high performance in image classification by enhancing the similarity between the query vector and its corresponding key vector, while reducing similarity with the key vectors of other images. From the perspective of data augmentation, Chen et al. (2020) applied various transformations such as random cropping, inversion, grayscale, etc., on images, and extracted invariant representations through contrastive learning. Subsequent works have made further improvements, achieving performance levels comparable to supervised learning algorithms on certain tasks.

3. Methodology

We introduce our main algorithm LIOM, the offline training in Sec. 3.1 and the online testing in Sec. 3.2.

3.1. Offline training

This section first introduces a novel policy data augmentation method. Then, we introduce the encoder training based on contrastive learning, where the opponent policy representation can be obtained from historical trajectory data. The policy representation will assist in training the general response policy and solving online execution policies.

Data augmentation is widely used for representation extraction, where it can extract important features while disregarding irrelevant information. In contrastive learning, data augmentation is mainly used to create positive and negative samples. However, the opponent's policy representation cannot be learned independently, it can only be extracted from interaction trajectories. We thus define a policy that interacts with an opponent's policy as an *augmentation policy*. We use some pure policies as augmentation policies since they take deterministic actions and have a minor effect on the trajectory representations' distribution.

By treating the interaction with an augmentation policy as a form of data augmentation, we can define a policy-based contrastive loss. For a given N opponent policy set, we randomly choose two augmentation policies and to interact with the opponent policies. Then, the obtained trajectories are encoded to form the set of trajectory representations $\{\mu_1, \mu_2, ..., \mu_{2N}\}$, where μ_{2k-1} and μ_{2k} are generated by the same opponent policy. The loss function can be defined as:

$$L = \frac{1}{2N} \sum_{k=1}^{N} [l(2k-1, 2k) + l(2k, 2k-1)], \quad (1)$$

where:

$$l(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbf{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}. \quad (2)$$

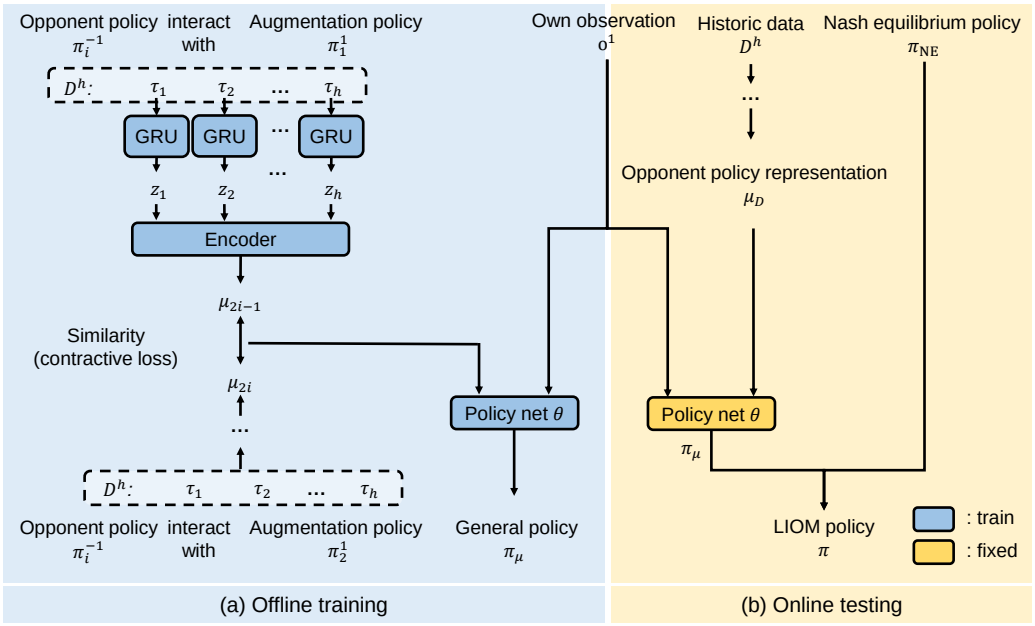

Figure 1. LIOM consists of two stages: (a) Offline training and (b) Online testing.

Here $s_{i,j}$ represents the cosine similarity between trajectory representation μ_i and μ_j, and τ is the temperature coefficient.

Due to the unknown observation information of the opponent, inferring the current opponent policy from a single-episode trajectory is challenging. Therefore, it is necessary to extract cross-episode opponent policy representations. The specific procedure is illustrated in Figure 1(a). Let h denote the number of game episodes required to learn the policy representations. For a N opponent policy set $\{\pi_1^{-1}, \pi_2^{-1}, ..., \pi_N^{-1}\}$, we randomly select two augmented policies π_1^1 and π_1^2. Let opponent policy π_i^{-1} interact with augmented policy π_1^1, and randomly sample a trajectory set $D^h = \{\tau_1, \ldots, \tau_h\}$. For each trajectory, we use GRU to extract the features:

$$z_t = f_{\text{GRU}}(\tau_t). \tag{3}$$

It is necessary to further encode and aggregate the representations z_t obtained from trajectories. In this paper, we employ the mean operation for aggregation due to simplify the expression:

$$\mu_{2i-1} = \frac{1}{h}\sum_{t=1}^{h} f_{\text{MLP}}(z_t). \tag{4}$$

Similarly, opponent policies interact with the augmented policy π_1^2 to construct a set of trajectory representations $\{\mu_1, \ldots, \mu_{2N}\}$, based on extracted features. By minimizing contrast loss (Eq. 1), we can obtain the opponent policy encoder $f_\phi(\cdot)$ in offline stage:

$$\mu_D = f_\phi(D). \tag{5}$$

After that, the opponent policy representation μ is regarded as labeled information, which assists the agent to make optimal responsive decisions against different opponent policies. This is particularly important in scenarios where the opponent's observations are unknown.

The core idea is to select an opponent policy π_k^{-1} and interact with it, storing the trajectory information τ in the historical trajectory set D_k. Before each episode of the game starts, the historical trajectory D_i^h for the current episode is selected from D_k. The opponent policy representation μ is generated based on D_i^h and a pre-trained opponent policy encoder $f_\phi(\cdot)$, and then it is combined with the observation o_t^1 of the main agent as a new observation to the critic and policy networks for training, and the network parameters are updated using the SAC algorithm. The main advantage of training a general policy lies in enabling the opponent policy encoder to obtain relatively accurate representations of unknown opponent policies, improving generalization.

3.2. Online testing

This section first introduces the representation extraction method for online opponent policies. Then, we discuss the choice between conservative and exploitative policies and

Algorithm 1 LIOM(testing)

Require: Nash equilibrium policy π_{NE}, general policy π_μ, opponent policy encoder $f_\phi(\cdot)$, opponent policy set Π^{test}, number of testing episodes T, trajectory length H, number of steps behind h, EXP3 parameters $p = [0.5, 0.5]$, $s = [0, 0]$, $\rho = 1$, $\eta = 0.1$.

for testing episode $i = 0 \cdots T-1$ **do**
  Initialize history trajectory set $D = \emptyset$
  if $i < h$ **then**
    Choose to interact with opponent using π_{NE}, generating trajectory τ_i.
    $D \leftarrow D \cup \tau_i$.
  else
    Cut out historical trajectory D_i^h for current episode from D.
    Generate opponent policy representation $\mu = f_\phi(D_i^h)$.
    Update probability distribution p according to Eq. (6).
    Choose policy π_i for this episode based on probability distribution p, generating trajectory τ_i.
    Update score s based on chosen policy and Eq. (7).
    if $\pi_i = \pi_\mu$ **then**
      $D \leftarrow D \cup \tau_i$.
    end if
  end if
end for

present a dynamic response policy that maximizes expected rewards.

Using the opponent policy encoder f_ϕ obtained in the offline training phase and the general policy π_μ, LIOM can respond to the opponent policy in online testing. For opponents with continuously changing policies, the agent fits the opponent policy representation μ using the recent h episodes of historical trajectory D^h. The real-time calculated μ is then as the input of the general policy π_μ to continuously adjust the currently used policy, achieving the optimal response to the current opponent. However, for unknown opponent policies, although the general policy and policy representation theoretically have certain generalization capabilities, the agent still cannot guarantee to respond to any unknown opponent policy. A conservative and stable policy is, therefore, necessary to handle this situation.

To maximize expected return during online testing, the algorithm needs to balance between the conservative policy π_{NE} and the exploitative policy π_μ. This scenario can be modeled as a classic Multi-Armed Bandit (MAB) problem, that is, how to quickly converge to the higher expected return policy when the return distribution for choosing policies π_{NE} and π_μ is uncertain. We choose to use the EXP3 (Auer et al., 2002) algorithm to solve the problem. The EXP3 algorithm dynamically maintains an action probability distribution p, and the probability of choosing action a in the ith selection is given by:

$$p_i(a) = (1-\eta)\frac{(1+\rho)^{s_i(a)}}{\sum_{j=1}^{K}(1+\rho)^{s_j(a)}} + \frac{\eta}{K}, \tag{6}$$

where K is the number of actions, and ρ and η are hyperparameters, s represents the score of each action, which is also dynamically maintained:

$$s_{i+1}(a) = s_i(a) + \frac{\eta r}{K p_i(a)}. \tag{7}$$

Here, a is selected based on the distribution p, and r is the reward obtained in this selection. In theory, the EXP3 algorithm's regret R_n^* has a lower bound:

$$R_n^* \geq c\sqrt{nK}, \tag{8}$$

where n is the total number of selections, and c is a constant.

Combined with the policy representation estimation and policy selection algorithm EXP3, we can obtain the online part of the LIOM algorithm. As shown in Algorithm 1, at the beginning of testing, LIOM uses the Nash equilibrium policy for a period of interaction. On the one hand, the algorithm needs enough data to construct the historical trajectory set D. On the other hand, using the Nash equilibrium policy for exploration is a more stable approach when there is less information about the opponent. When there is enough data, only the trajectory data of interactions between π_μ and the opponent will be kept in D to avoid the influence of trajectory data with different distributions on the performance of π_μ. It is worth noting that the exploration factor η should not be set too large or too small. Due to the possible changes in the opponent's policy, the exploration of policy selection should be ensured as much as possible.

4. Experiments

The purpose of our experimental research is to compare the performance of various methods when facing online adversarial scenarios with known, unknown, and non-stationary opponent policies. Additionally, we analyze the effectiveness of policy representation and the balance between conservative and exploitation.

4.1. Experimental Setup

Environments. We employed two classic multi-agent adversarial environments, Kuhn Poker (Fu et al., 2022) and Soccer (Zheng et al., 2018). Kuhn Poker is a simplified version of Texas Hold'em environment where each player chooses one card from J, Q, and K, and subsequently selects pass or bet in a turn-based manner. The rules is presented in

Table 1. Kuhn Poker rules

PLAYER 1	PLAYER 2	PLAYER1	REWARD
PASS	PASS		± 1
PASS	BET	PASS	$(-1, 1)$
PASS	BET	BET	± 2
BET	BET		± 2
BET	PASS		$(1, -1)$

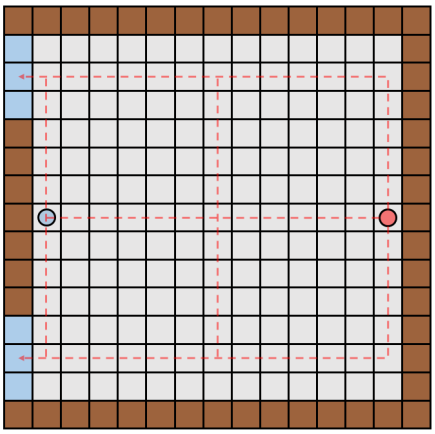

Figure 2. Soccer configuration.

Table 1, where ± 1 represents +1 for the player with a higher card value, and -1 otherwise. Soccer is a partially observable environment, as illustrated in Figure 2, where the attacker (red) moves towards the goal along any path while the defender (blue) tries to stop them, and can only observe the position of the opponent when they are nearby. The winner receives a reward of 1, while the loser receives -1. We selected player 1 in Kuhn Poker and the defender in Soccer as our main agents to control. Due to the unknown hand cards or positions of the opponent, this presents a multi-agent competition problem with limited opponent information.

Opponent policies. For each environment, we have designed six opponent policies with distinct styles denoted by $\{\pi_0, \pi_1, \cdots, \pi_5\}$. Among them, policies $\{\pi_0, \pi_1, \pi_2\}$ constitute the visible policies and are used to form the training set Π^{train}, while policies $\{\pi_3, \pi_4, \pi_5\}$ serve as invisible policies. During online testing, we evaluate the effectiveness of various algorithms on three categories of opponent policy sets: "seen", "unseen", and "mix".

Comparing methods. The following policies will be compared in LIOM:

- NE: Nash equilibrium policy, a conservative policy that can be solved through self-play and other methods.
- ORACLE: Oracle policy refers to a policy that is aware of the opponent policy type and trains separate policy networks for each type of opponent policy. It can be considered as the best response policy.
- DRON: An implicit opponent modeling algorithm that uses opponent information as an additional input to the network during offline training.
- Deep BPR+: An explicit opponent modeling algorithm that selects the best response from the offline-trained policy library using a Bayesian belief model. It can learn new response policies and update the policy library by detecting unknown opponents online.
- LIOM w/o EXP3: Only use the general policy π_μ to combat the opponent, where μ is jointly calculated based on historical trajectories within several episodes.

4.2. Online Testing with Fixed Opponents

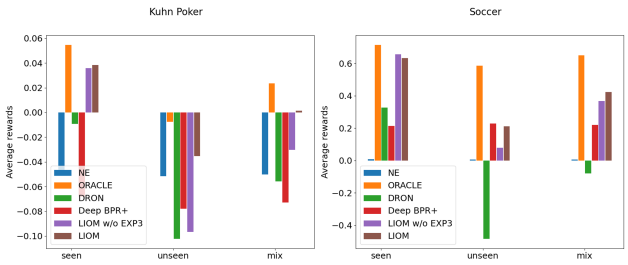

Figure 3. The average rewards of different methods when facing three categories of opponent policy sets in online testing.

In this experiment, we demonstrate the average rewards of different methods in online testing against three categories of opponent policy sets ("seen", "unseen" and "mix").

Based on the average rewards on Kuhn Poker presented in Figure 3, LIOM exhibits performance inferior only to the best response policy across all types of opponent policy sets. In the "seen" setting, LIOM performs similarly to LIOM w/o EXP3, indicating that when facing known opponent policies, LIOM tends to select strategies with stronger exploitation to achieve best responses. Moreover, LIOM outperforms DRON significantly, demonstrating the effectiveness of offline opponent policy representation extraction across episodes. In the "unseen" setting, LIOM w/o EXP3 outperforms DRON, suggesting that the general policy π_μ possesses better generalization capability, while LIOM surpasses LIOM w/o EXP3, owing to the lower bound on LIOM's performance guaranteed by EXP3, which can handle difficult-to-generalize opponent policies. In the "mix" setting, LIOM approaches the ORACLE policy for the known opponent type, indicating that the opponent policy representation μ inferred from limited opponent information

Table 2. Average rewards in the interaction with independent opponent policies.

	π_0	π_1	π_2
NE	-0.046 ± 1.308	-0.042 ± 1.274	-0.058 ± 1.143
ORACLE	0.068 ± 1.498	0.065 ± 1.673	0.031 ± 1.295
DRON	0.069 ± 1.509	0.025 ± 1.466	-0.122 ± 1.301
DEEP BPR+	-0.106 ± 1.507	-0.063 ± 1.43	-0.035 ± 1.307
LIOM W/O EXP3	0.057 ± 1.505	0.031 ± 1.49	0.02 ± 1.297
LIOM	0.055 ± 1.457	0.04 ± 1.456	0.018 ± 1.282
	π_3	π_4	π_5
NE	-0.061 ± 1.11	-0.051 ± 1.175	-0.043 ± 1.206
ORACLE	0.026 ± 1.335	-0.039 ± 1.113	-0.01 ± 1.42
DRON	-0.156 ± 1.258	-0.093 ± 1.337	-0.058 ± 1.377
DEEP BPR+	-0.039 ± 1.31	-0.098 ± 1.427	-0.097 ± 1.379
LIOM W/O EXP3	0.0 ± 1.326	-0.137 ± 1.349	-0.153 ± 1.364
LIOM	0.008 ± 1.291	-0.065 ± 1.163	-0.049 ± 1.197

by contrastive learning can effectively describe both known and unknown opponents, thereby assisting the general policy π_μ in making decisions.

Based on the average rewards on Soccer shown in Figure 3, we arrive at conclusions similar to those on Kuhn Poker. The only difference is that in the "unseen" setting, Deep BPR+ performs slightly better than LIOM due to its capability of online learning against unknown opponent policies. However, the existing opponent modeling methods heavily rely on the completeness of opponent information, leading to a significant degradation in modeling accuracy when the opponent information is limited. Therefore, considering all settings, LIOM demonstrates superior performance.

4.3. Analysis of Opponent Policy Representations

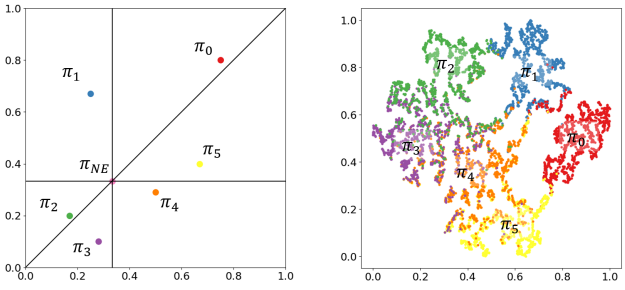

Figure 4. The true coordinates (left) and representation distributions (right) of different opponent policies in Kuhn Poker.

In this experiment, we further analyse the effectiveness and generalization of offline-extracted opponent policy representations on Kuhn Poker, as it offers a straightforward parameterized approach to define opponent policies. Figure 4 shows the true distribution of six opponent policies, as well as their representation distribution in the representation space extracted by offline contrastive learning. Note that, the training of the opponent policy encoder only use $\Pi^{train} = \{\pi_0, \pi_1, \pi_2\}$.

Figure 4 shows that the distribution of opponent policy representations, as depicted, closely aligns with the true distribution. This indicates the accuracy and generalization of opponent policy representations extracted through contrastive learning. As policies cannot be directly represented, Figure 4 displays the distribution of trajectory representations obtained from interacting with a randomly augmentation policy. In fact, opponent policy representations computed by different augmentation policies exhibit similar distributions, and the distribution of opponent policy representations is almost independent of the choice of augmentation policies. This is because, through contrastive learning, we only extract the portions of the trajectory representation that are relevant to opponent policies.

To further illustrate, we show in Table 2 the average returns of various algorithms interacting with specific opponent policies. Compared to DRON, LIOM w/o EXP3 performs better when facing unknown opponent policy π_3, indicating that the learned opponent policy representation has certain generalization ability. However, LIOM w/o EXP3 performs worse on opponent policies π_4 and π_5 than LIOM, which is consistent with the proximity relationships between different opponent policies as depicted in Figure 4. This also indicates the necessity of introducing the EXP3 to balance conservative and exploitative policies.

4.4. Online Testing with Non-stationary Opponents

In this experiment, we present the reward curve against non-stationary opponents. As shown in Figure 5, LIOM is capable of inferring the current opponent's policy representation μ based on the recent historical trajectory data D, which serves as an additional input to the general pol-

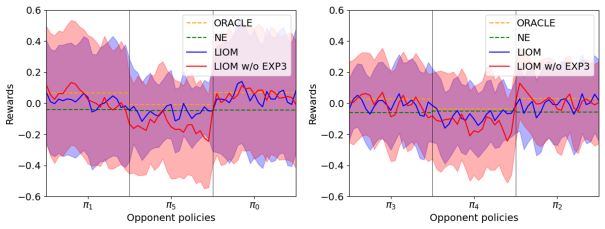

Figure 5. The reward curve of online testing with non-stationary opponent policies in Kuhn Poker.

icy π_μ. Thus, it can dynamically respond to non-stationary opponent policies, and its average performance is comparable to the best response policy ORACLE. Additionally, LIOM selects between conservative Nash equilibrium policies and exploitative general policies using EXP3, so that the expected reward falls between the two, but guaranteeing a lower bound on the reward when facing unknown opponents.

5. Conclusion and Future Work

In this paper, we propose a policy-based data augmentation method that achieves offline cross-episode opponent policy representation extraction through contrastive learning. Our approach does not rely on the completeness of opponent information and can infer the current opponent's policy representation through limited information, which is not achievable by many existing opponent modeling works. By introducing additional policy representations, a general policy is trained offline and used as an exploitative policy. We also use EXP3 to balance between conservative and exploitation. Results from online testing demonstrate that LIOM exhibits high expected returns and strong generalization ability when facing unknown opponents.

Future work may involve exploring the relationship between constructing offline policy sets and the generalization ability of general policies, as well as developing methods for constructing augmentation policies in complex environments. We aim to build a diverse and complete policy set for both opponent and augmentation policies. However, as problem scales increase, it becomes difficult for humans to define "diversity" in the policy representation space, which may require further decomposition of complex policies to extend our algorithms to complex environments.

References

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Foerster, J. N., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*, 2017.

Fu, H., Tian, Y., Yu, H., Liu, W., Wu, S., Xiong, J., Wen, Y., Li, K., Xing, J., Fu, Q., et al. Greedy when sure and conservative when uncertain about the opponents. In *International Conference on Machine Learning*, pp. 6829–6848. PMLR, 2022.

He, H., Boyd-Graber, J., Kwok, K., and Daumé III, H. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, pp. 1804–1813. PMLR, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Hernandez-Leal, P., Taylor, M. E., Rosman, B. S., Sucar, L. E., and Munoz de Cote, E. Identifying and tracking switching, non-stationary opponents: A bayesian approach. 2016.

Hong, Z.-W., Su, S.-Y., Shann, T.-Y., Chang, Y.-H., and Lee, C.-Y. A deep policy inference q-network for multi-agent systems. *arXiv preprint arXiv:1712.07893*, 2017.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Raileanu, R., Denton, E., Szlam, A., and Fergus, R. Modeling others using oneself in multi-agent reinforcement learning. In *International conference on machine learning*, pp. 4257–4266. PMLR, 2018.

Rosman, B., Hawasly, M., and Ramamoorthy, S. Bayesian policy reuse. *Machine Learning*, 104:99–127, 2016.

Yang, T., Meng, Z., Hao, J., Zhang, C., Zheng, Y., and Zheng, Z. Towards efficient detection and optimal response against sophisticated opponents. *arXiv preprint arXiv:1809.04240*, 2018.

Zheng, Y., Meng, Z., Hao, J., Zhang, Z., Yang, T., and Fan, C. A deep bayesian policy reuse approach against non-stationary agents. *Advances in neural information processing systems*, 31, 2018.