Towards Efficient Detection and Optimal Response against Sophisticated Opponents

Tianpei Yang[1] , **Jianye Hao**[1*] , **Zhaopeng Meng**[1] , **Chongjie Zhang**[2] , **Yan Zheng**[1] and **Ze Zheng**[3]

[1]College of Intelligence and Computing, Tianjin University

[2]MMW, Tsinghua University

[3]Beifang Investigation, Design & Research CO.LTD

{tpyang, jianye.hao, mengzp, yanzheng}@tju.edu.cn, chongjie@tsinghua.edu.cn, zezheng0618@gmail.com

Abstract

Multiagent algorithms often aim to accurately predict the behaviors of other agents and find a best response accordingly. Previous works usually assume an opponent uses a stationary strategy or randomly switches among several stationary ones. However, an opponent may exhibit more sophisticated behaviors by adopting more advanced reasoning strategies, e.g., using a Bayesian reasoning strategy. This paper proposes a novel approach called Bayes-ToMoP which can efficiently detect the strategy of opponents using either stationary or higher-level reasoning strategies. Bayes-ToMoP also supports the detection of previously unseen policies and learning a best-response policy accordingly. We provide a theoretical guarantee of the optimality on detecting the opponent's strategies. We also propose a deep version of Bayes-ToMoP by extending Bayes-ToMoP with DRL techniques. Experimental results show both Bayes-ToMoP and deep Bayes-ToMoP outperform the state-of-the-art approaches when faced with different types of opponents in two-agent competitive games.

1 Introduction

In multiagent systems, the ideal behavior of an agent is contingent on the behaviors of coexisting agents. However, agents may exhibit different behaviors adaptively depending on the contexts they encounter. Hence, it is critical for an agent to quickly predict or recognize the behaviors of other agents, and make a best response accordingly [Powers and Shoham, 2005; Fernández et al., 2010; Albrecht and Stone, 2018; Hernandez-Leal et al., 2017a; Yang et al., 2019].

One efficient way of recognizing the strategies of other agents is to leverage the idea of Bayesian Policy Reuse (BPR) [2016], which was originally proposed to determine the best policy when faced with different tasks. Hernandez-Leal et al. [2016] proposed BPR+ by extending BPR to multiagent learning settings to detect the dynamic changes of an opponent's strategies. BPR+ also extends BPR with the ability

to learn new policies online against an opponent using previously unseen policies. However, BPR+ is designed for single-state repeated games only. Later, Bayes-Pepper [2017] is proposed for stochastic games by combing BPR and Pepper [Crandall, 2012]. However, Bayes-Pepper cannot handle an opponent which uses a previously unknown policy. There are also some deep multiagent RL algorithms investigating how to learn an optimal policy by explicitly taking other agents' behaviors into account. He et al. [2016] proposed DRON which incorporates the opponent's observation into deep Q-network (DQN) and uses a mixture-of-experts architecture to handle different types of opponents. However, it cannot guarantee the optimality against each particular type of opponents. Recently, Zheng et al. [2018] proposed Deep BPR+ by extending BPR+ with DRL techniques to achieve more accurate detection and better response against different opponents. However, all these approaches assume that an opponent randomly switches its policies among a class of stationary ones. In practice, an opponent can exhibit more sophisticated behaviors by adopting more advanced reasoning strategies. In such situations, higher-level reasonings and more advanced techniques are required for an agent to beat such kinds of sophisticated opponents.

The above problems can be partially addressed by introducing the concept of Theory of Mind (ToM) [Baker et al., 2011; de Weerd et al., 2013]. ToM is a kind of recursive reasoning technique [Hernandez-Leal et al., 2017a; Albrecht and Stone, 2018] describing a cognitive mechanism of explicitly attributing unobservable mental contents such as beliefs, desires, and intentions to other players. Previous methods often use nested beliefs and "simulate" the reasoning processes of other agents to predict their actions [Gmytrasiewicz and Doshi, 2005; Wunder et al., 2012]. However, these approaches show no adaptation to non-stationary opponents [Hernandez-Leal et al., 2017a]. Later, De Weerd et al. [2013] proposed a ToM model which enables an agent to predict the opponent's actions by building an abstract model of its opponent using recursive nested beliefs. Additionally, they adopt a confidence value to help an agent to adapt to different opponents. However, the main drawbacks of this model are: 1) it works only if an agent holds exactly one more layer of belief than its opponent; 2) it is designed for predicting the opponent's primitive actions instead of high-level strategies, resulting in slow adaptation to non-stationary opponents; 3)

---

it shows poor performance against an opponent using previously unseen strategies.

To address the above challenges, we propose a novel algorithm, named Bayesian Theory of Mind on Policy (Bayes-ToMoP), which leverages the predictive power of BPR and recursive reasoning ability of ToM, to compete against such sophisticated opponents. In contrast to BPR which is capable of detecting non-stationary opponents only, Bayes-ToMoP incorporates ToM into BPR to quickly and accurately detect not only non-stationary, and more sophisticated opponents and compute a best response accordingly. Theoretical guarantees are provided for the optimal detection of the opponent's strategies. Besides, Bayes-ToMoP also supports detecting whether an opponent is using a previously unseen policy and learning an optimal response against it. Furthermore, Bayes-ToMoP can be straightforwardly extended to DRL environment with a neural network as the value function approximator, termed as deep Bayes-ToMoP. Experimental results show that both Bayes-ToMoP and deep Bayes-ToMoP outperform the state-of-the-art approaches when faced with different types of opponents in two-agent competitive games.

2 Background

2.1 Bayesian Policy Reuse

Given a set of previously-solved tasks $\mathcal{T}$ and an unknown task τ^*, BPR[2016] was originally proposed as a framework for an agent to select the best policy π^* from the policy library Π within as small numbers of trials as possible. BPR uses the concept of *belief* β, which is a probability distribution over the task set $\mathcal{T}$, to measure the degree to which τ^* matches the known tasks based on the signal σ. A signal σ can be any information that is correlated with the performance of a policy (e.g., immediate rewards, episodic returns). BPR also involves performance models of policies on previously-solved tasks, each of which $P(U|\tau, \pi)$ is a probability distribution over the utility of a policy π on a task τ.

A number of BPR variants with exploration heuristics are proposed to select the best policy, e.g., probability of improvement (BPR-PI) heuristic and expected improvement (BPR-EI) heuristic. BPR-PI heuristic utilizes the probability with which a policy can achieve a hypothesized increase in performance (U^+) over the current best estimate $\bar{U} = \max_{\pi \in \Pi} \sum_{\tau \in \mathcal{T}} \beta(\tau) E[U|\tau, \pi]$. Formally, it chooses the policy that most likely to achieve the utility U^+: $\pi^* = \arg\max_{\pi \in \Pi} \sum_{\tau \in \mathcal{T}} \beta(\tau) P(U^+|\tau, \pi)$. However, it is not straightforward to determine the appropriate value of U^+, thus another way of avoiding this issue is BPR-EI heuristic, which selects the policy most likely to achieve any possible utilities of improvement $\bar{U} < U^+ < U^{\max}$: $\pi^* = \arg\max_{\pi \in \Pi} \int_{\bar{U}}^{U^{\max}} \sum_{\tau \in \mathcal{T}} \beta(\tau) P(U^+|\tau, \pi) dU^+$. BPR [2016] showed BPR-EI heuristic performs best among all BPR variants. Therefore, we choose BPR-EI heuristic for playing against different opponents.

2.2 Theory of Mind

ToM model [de Weerd *et al.*, 2013] is used to predict an opponent's action by building an abstract model of the opponent using recursive nested beliefs. ToM model is described in the context of a two-player competitive game where an agent and its opponent differ in their abilities to make use of ToM. The notion of ToM_k indicates an agent that has the ability to use ToM up to the k-th order, and we briefly introduce the first two orders of ToM models.

A zero-order ToM (ToM_0) agent holds its zero-order belief in the form of a probability distribution on the action set of its opponent. The ToM_0 agent then chooses the action that maximizes its expected payoff. A first-order ToM agent (ToM_1) keeps both zero-order belief $\beta^{(0)}$ and first-order belief $\beta^{(1)}$. The first-order belief $\beta^{(1)}$ is a probability distribution that describes what the ToM_1 agent believes its opponent believes about itself. The ToM_1 agent first predicts its opponent's action under its first-order belief. Then, the ToM_1 agent integrates its first-order prediction with the zero-order belief and uses this integrated belief in the final decision. The degree to which the prediction influences the agent's actions is determined by its first-order confidence $0 \le c_1 \le 1$, which is increased if the prediction is right while decreased otherwise.

3 Bayes-ToMoP

3.1 Motivation

Previous works [Hernandez-Leal *et al.*, 2016; Hernandez-Leal and Kaisers, 2017; Hernandez-Leal *et al.*, 2017b; Zheng *et al.*, 2018; He and Boyd-Graber, 2016] assume that an opponent randomly switches its policies among a number of stationary policies. However, a more sophisticated agent may change its policy in a more principled way. For instance, it first predicts the policy of its opponent and then best responds towards the estimated policy accordingly. If the opponent's policy is estimated by simply counting the action frequencies, it is then reduced to the well-known fictitious play [Shoham and Leyton-Brown, 2009]. However, in general, an opponent's action information may not be observable during interactions. One way of addressing this problem is using BPR, which uses Bayes' rule to predict the policy of the opponent according to the received signals (e.g., rewards), and can be regarded as the generalization of fictitious play.

Therefore, a question naturally arises: how an agent can effectively play against both simple opponents with stationary strategies and more advanced ones (e.g., using BPR)? To address this question, we propose a new algorithm called Bayes-ToMoP, which leverages the predictive power of BPR and recursive reasoning ability of ToM to predict the strategies of such opponents and behave optimally. We also extend Bayes-ToMoP to DRL scenarios with a neural network as the value function approximator, termed as deep Bayes-ToMoP. In the following descriptions, we do not distinguish whether a policy is represented in a tabular form or a neural network unless necessary.

We assume the opponent owns a class of candidate stationary strategies to select from periodically. Bayes-ToMoP needs to observe the reward of its opponent which is not an assumption since in competitive settings, e.g., zero-sum games, an agent's opponent's reward is always the opposite of its own. Bayes-ToMoP does not need to observe the actions of its opponent except for learning a policy against an

unknown opponent strategy. We use the notation of Bayes-ToMoP$_k$ to denote an agent with the ability of using Bayes-ToMoP up to the k-th order. Intuitively, Bayes-ToMoP$_i$ with a higher-order theory of mind could take advantage of any Bayes-ToMoP$_j$ with a lower-order one ($i > j$). De Weerd et al. [2013] showed that the reasoning levels deeper than 2 do not provide additional benefits, so we focus on Bayes-ToMoP$_0$ and Bayes-ToMoP$_1$. Bayer-ToMoP$_k$ ($k > 1$) can be naturally constructed by incorporating a higher-order ToM idea into our framework.

3.2 Bayes-ToMoP$_0$ Algorithm

We start with the simplest case: Bayes-ToMoP$_0$, which extends ToM$_0$ by incorporating Bayesian reasoning techniques to predict the strategy of an opponent. Bayes-ToMoP$_0$ holds a zero-order belief $\beta^{(0)}$ about its opponent's strategies $\{j | j \in \mathcal{J}\}$, each of which $\beta^{(0)}(j)$ is a probability that its opponent may adopt each strategy j: $\beta^{(0)}(j) \geq 0, \forall j \in \mathcal{J}$. $\sum_{j \in \mathcal{J}} \beta^{(0)}(j) = 1$. Given a utility U, a performance model $P_{self}(U|j, \pi)$ describes the probability of an agent using a policy $\pi \in \Pi$ against an opponent's strategy j.

For Bayes-ToMoP$_0$ agent equipped with a policy library Π against its opponent's with a strategy library $\mathcal{J}$, it first initializes its performance models $P_{self}(U|\mathcal{J}, \Pi)$ and zero-order belief $\beta^{(0)}$. Then, in each episode, given the current belief $\beta^{(0)}$, Bayes-ToMoP$_0$ agent evaluates the expected improvement utility defined following BPR-EI heuristic for all policies and then selects the optimal one. Next, Bayes-ToMoP$_0$ agent updates its zero-order belief using Bayes' rule [Rosman et al., 2016]. At last, Bayes-ToMoP$_0$ detects whether its opponent is using a previously unseen policy. If yes, it learns a new policy against its opponent. The new strategy detection and learning algorithm will be described in Section 3.4.

Finally, note that without the new strategy detection and learning phase, Bayes-ToMoP$_0$ agent is essentially equivalent to BPR and Bayes-Pepper since they both first predict the opponent's strategy (or taks type) and then select the optimal policy following BPR heuristic, each strategy of the opponent here can be regarded as a task in the original BPR. Besides, the full Bayes-ToMoP$_0$ is essentially equivalent to BPR+ since both can handle previously unseen strategies.

3.3 Bayes-ToMoP$_1$ Algorithm

Next, we move to Bayes-ToMoP$_1$ algorithm. Apart from its zero-order belief, Bayes-ToMoP$_1$ also maintains a first-order belief, which is a probability distribution that describes the probability that an agent believes his opponent believes it will choose a policy $\pi \in \Pi$. The overall strategy of Bayes-ToMoP$_1$ is shown in Algorithm 1. Given the policy library Π and $\mathcal{J}$, performance models $P_{self}(U|\mathcal{J}, \Pi)$ and $P_{oppo}(U|\Pi, \mathcal{J})$, zero-order belief $\beta^{(0)}$ and first-order belief $\beta^{(1)}$, Bayes-ToMoP$_1$ agent first predicts the policy $\hat{j}$ of its opponent assuming the opponent maximizes its own utility based on BPR-EI heuristic under its first-order belief (Line 2). However, this prediction may conflict with its zero-order belief. To address this conflict, Bayes-ToMoP$_1$ holds a first-order confidence $c_1 (0 \leq c_1 \leq 1)$ serving as the weighting

---

Algorithm 1 Bayes-ToMoP$_1$ algorithm

Initialize: Policy library Π and $\mathcal{J}$, performance models $P_{self}(U|\mathcal{J}, \Pi)$ and $P_{oppo}(U|\Pi, \mathcal{J})$, zero-order belief $\beta^{(0)}$, first-order belief $\beta^{(1)}$

1: **for** each episode **do**
2:    Compute the first-order opponent policy prediction $\hat{j}$:
$$\arg\max_{j \in \mathcal{J}} \int_{\underline{U}}^{U^{\max}} \sum_{\pi \in \Pi} \beta^{(1)}(\pi) P_{oppo}(U|\pi, j) dU$$
3:    Integrate $\hat{j}$ with $\beta^{(0)}$: $I(\beta^{(0)}, \hat{j}, c_1)$ (see Equation (1))
4:    Select the optimal policy π^*:
$$\arg\max_{\pi \in \Pi} \int_{\underline{U}}^{U^{\max}} \sum_{j \in \mathcal{J}} I(\beta^{(0)}, \hat{j}, c_1)(j) P_{self}(U|j, \pi) dU$$
5:    Play and receive the episodic return $\langle r_{self}, r_{oppo} \rangle$
6:    **for** each own policy $\pi \in \Pi$ **do**
7:       Update first-order belief $\beta^{(1)}$:
$$\beta^{(1)}(\pi) = \frac{P_{oppo}(r_{oppo}|\pi, \hat{j})\beta^{(1)}(\pi)}{\sum_{\pi' \in \Pi} P_{oppo}(r_{oppo}|\pi', \hat{j})\beta^{(1)}(\pi')}$$
8:    **end for**
9:    **for** each opponent strategy $j \in \mathcal{J}$ **do**
10:      Update zero-order belief $\beta^{(0)}$:
$$\beta^{(0)}(j) = \frac{P_{self}(r_{self}|j, \pi)\beta^{(0)}(j)}{\sum_{j' \in \mathcal{J}} P_{self}(r_{self}|j', \pi)\beta^{(0)}(j')}$$
11:   **end for**
12:   Update c_1 following Equation (2)
13:   Detect new opponent strategy
14: **end for**

---

factor to balance the influence between its first-order prediction and zero-order belief. Then, an integration function I is introduced to compute the final prediction results which is defined as the linear combination of the first-order prediction $\hat{j}$ and zero-order belief $\beta^{(0)}$ weighted by the confidence degree c_1 following Equation 1 [de Weerd et al., 2013] (Line 3).

$$I(\beta^{(0)}, \hat{j}, c_1)(j) = \begin{cases} (1 - c_1)\beta^{(0)}(j) + c_1 & \text{if } j = \hat{j} \\ (1 - c_1)\beta^{(0)}(j) & \text{otherwise} \end{cases} \tag{1}$$

Next, Bayes-ToMoP$_1$ agent computes the optimal policy based on the integrated belief (Line 4). At last, Bayes-ToMoP$_1$ updates its first-order belief and zero-order belief using Bayes' rule [Rosman et al., 2016] (Lines 6-11).

The next issue is how to update the first-order confidence degree c_1. The value of c_1 can be understood as the exploration rate of using first-order belief to predict the opponent's strategies. In previous ToM model [de Weerd et al., 2013], the value of c_1 is increased if the prediction is right while decreased otherwise based on the assumption that an agent can observe the actions of its opponent. However, in our settings, the prediction works on a higher level of behaviors (the policies), which information usually is not available (agents are not willing to reveal their policies to others to avoid being exploited in competitive environments). Therefore, we propose to use game outcomes as the signal to indicate whether our previous predictions are correct and adjust the first-order confidence degree accordingly. Specifically, in a competitive environment, we can distinguish game outcomes into three cases: win ($r_{self} > r_{oppo}$), lose or draw. Thus, the value of

c_1 is increased when the agent wins and decreased otherwise by an adjustment rate of λ. Following this heuristic, Bayes-ToMoP$_1$ can easily take advantage of Bayes-ToMoP$_0$ since it can well predict the policy of Bayes-ToMoP$_0$ in advance. However, this heuristic does not work with less sophisticated opponents, e.g., an opponent simply switching among several stationary policies without the ability of using ToM. This is due to the fact that the curve of c_1 becomes oscillating when it is faced with an agent who is unable to make use of ToM, thus fails to predict the opponent's behaviors accurately.

To this end, we propose an adaptive and generalized mechanism to adjust the value of c_1. We first introduce the concept of win rate $v_i = \frac{\sum_{i-l}^i r_{self}}{l}$ during a fixed length l of episodes. Since Bayes-ToMoP$_1$ agent assumes its opponent is Bayes-ToMoP$_0$ at first, the value of l controls the number of episodes before considering its opponent may switch to a less sophisticated type. If the average performance till the current episode is better than the previous episode's ($v_i \geq v_{i-1}$), we increase the weight of using first-order prediction, i.e., increasing the value of c_1 with an adjustment rate λ; if v_i is less than v_{i-1} but still higher than a threshold δ, it indicates the performance of the first-order prediction diminishes. Then Bayes-ToMoP$_1$ decreases the value of c_1 quickly with a decreasing factor $\frac{lgv_i}{lg(v_i-\delta)}$; if $v_i \leq \delta$, the rate of exploring first-order belief is set to 0 and only zero-order belief is used for prediction. Formally we have:

$$c_1 = \begin{cases} ((1-\lambda)c_1 + \lambda)\mathbf{F}(v_i) & \text{if } v_i \geq v_{i-1} \\ (\frac{lgv_i}{lg(v_i-\delta)}c_1)\mathbf{F}(v_i) & \text{if } \delta < v_i < v_{i-1} \\ \lambda\mathbf{F}(v_i) & \text{if } v_i \leq \delta \end{cases} \quad (2)$$

where δ is the threshold of the win rate v_i, which reflects the lower bound of the difference between its prediction and its opponent's actual behaviors. $\mathbf{F}(v_i)$ is an indicator function to control the direction of adjusting the value of c_1. Intuitively, Bayes-ToMoP$_1$ detects the switching of its opponent's strategies at each episode i and reverses the value of $\mathbf{F}(v_i)$ whenever its win rate v_i is no larger than δ (Equation 3).

$$\mathbf{F}(v_i) := \begin{cases} 1 & \text{if } (v_i \leq \delta \ \& \ \mathbf{F}(v_i) = 0) \\ 0 & \text{if } (v_i \leq \delta \ \& \ \mathbf{F}(v_i) = 1) \end{cases} \quad (3)$$

Finally, Bayes-ToMoP$_1$ learns a new optimal policy if it detects a new opponent strategy (detailed in next section).

3.4 New Opponent Detection and Learning

The new opponent detection and learning component is the same for all Bayes-ToMoP$_k$ agents ($k \geq 0$). Bayes-ToMoP$_k$ first detects whether its opponent is using a new kind of strategies. This is achieved by recording a fixed length of game outcomes and checking whether its opponent is using one of the known strategies at each episode. In details, Bayes-ToMoP$_k$ keeps a length h of memory recording the game outcomes at episode i, and uses the win rate $\theta_i = \frac{\sum_{i-h}^i r_{self}}{h}$ over the most recent h episodes as the signal indicating the average performance over all policies till the current episode i. If the win rate θ_i is lower than a given threshold δ ($\theta_i < \delta$), it indicates that all existing policies show poor performance against the

current opponent strategy, in this way Bayes-ToMoP$_k$ agent infers that the opponent is using a previously unseen policy outside the current policy library.

Since we can easily obtain the average win rate $\theta^{(\pi j)}$ of each policy π against each known opponent strategy j, the lowest win rate among the best-response policies ($\min_{\pi \in \Pi} \max_{j \in \mathcal{J}} \theta^{(\pi j)}$) can be seen as an upper bound of the value of δ. The value of h controls the number of episodes before considering whether the opponent is using a previously unseen strategy. Note that the accuracy of detection is increased with the increase of the memory length h, however, a larger value of h would necessarily increase the detection delay. The optimal memory length is determined empirically through extensive simulations (see the extended version[1]).

After detecting the opponent is using a new strategy, the agent begins to learn the best-response policy against it. Following previous work [Hernandez-Leal *et al.*, 2016], we adopt the same assumption that the opponent will not change its strategy during the learning phase (a number of rounds). Otherwise, the learning process may not converge. For tabular Bayes-ToMoP, we adopt the traditional model-based RL: R-max [Brafman and Tennenholtz, 2002] to compute the optimal policy. Specifically, once a new strategy is detected, R-max estimates the state transition function T and reward function R with $\hat{T}$ and $\hat{R}$. It also keeps a transition count $c(s, a, s')$ and total reward $t(s, a)$ for all state-action pairs. Each transition $\langle s, a, s', r \rangle$ results in an update for the transition count: $c(s, a, s') \leftarrow c(s, a, s') + 1$ and total reward: $t(s, a) \leftarrow t(s, a) + r$. The estimates $\hat{T}$ and $\hat{R}$ is updated using $c(s, a, s')$ and $t(s, a)$ with a given parameter n: $\hat{T}(s, a, s') = c(s, a, s')/n$, $\hat{R}(s, a) = t(s, a)/n$ if $\sum_{s'} c(s, a, s') \geq n$. R-max computes $\hat{Q}(s, a) = \hat{R}(s, a) + \gamma \sum_{s'} \hat{T}(s, a, s') \max_{a'} \hat{Q}(s', a')$ for all state-action pairs and selects the action that maximizes $\hat{Q}(s, a)$ according to ϵ-greedy mechanism.

For deep Bayes-ToMoP, we apply DQN [Mnih *et al.*, 2015] to do off-policy learning using the obtained interaction experience. DQN is a deep Q-learning method with experience replay, consisting of a neural network approximating $Q(s, a; \theta)$ and a target network approximating $Q(s, a; \theta^-)$. DQN draws samples (or minibatches) of experience $(s, a, s', r) \sim U(\mathcal{D})$ uniformly from a replay memory $\mathcal{D}$, and updates using the following loss function: $L(\theta) = \mathbf{E}_{(s,a,s',r),r \sim U(\mathcal{D})}[(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta))^2]$. DQN is not a requirement, actually, our learning framework is general in which other DRL approaches can be applied as well. However, most DRL algorithms suffer from the sample efficiency problems under some specific settings. This can be addressed by incorporating sample efficient DRL methods in the future. To generate new performance models, we use a neural network to estimate the policy of the opponent based on the observed state-action history of the opponent using supervised learning techniques [Zheng *et al.*, 2018; Foerster *et al.*, 2018].

After the above learning phase, new performance models

---

[1]http://arxiv.org/abs/1809.04240

are generated using rewards obtained from a number of simulations of the agent's policy against the opponent's estimated strategy. These values are modeled as a Gaussian distribution to obtain the performance models. Finally, it adds the new policy π_{new} and the estimated opponent policy to its policy library Π and its opponent's policy library $\mathcal{J}$ respectively.

We provide a theoretical analysis about Bayes-ToMoP and the proof is given in the extended version[1].

Theorem 1 (*Optimality on Strategy Detection*) *If the opponent plays a strategy from the known policy library, Bayes-ToMoP can detect the strategy w.p.1 and selects an optimal response policy accordingly.*

4 Simulations

In this section, we present experimental results of Bayes-ToMoP compared with state-of-the-art tabular approaches (BPR+ [Hernandez-Leal *et al.*, 2016] and Bayes-Pepper [Hernandez-Leal and Kaisers, 2017]). For deep Bayes-ToMoP, we compare with the following four baseline strategies: 1) BPR+, 2) Bayes-Pepper (BPR+ and Bayes-Pepper use a neural network as the value function approximator), 3) DRON [He and Boyd-Graber, 2016] and 4) deep BPR+ [Zheng *et al.*, 2018]. The network structure and details of parameter settings are in the extended version[1].

4.1 Game Settings

We evaluate the performance of Bayes-ToMoP on the following testbeds: soccer [Littman, 1994; He and Boyd-Graber, 2016] and thieves and hunters [Goodrich *et al.*, 2003; Crandall, 2012]. Soccer (Figure 1) and Thieves and hunters (Figure 2) are two stochastic games both on a 7×7 grid. Two players, A and B, start at one of starting points in the left and right respectively and can choose one of the following 5 actions: go left, right, up, down and stay. Any action that goes to grey-slash grids or beyond the border is invalid. 1) In soccer, the ball (circle) is randomly assigned to one player initially. The possession of the ball switches when two players move to the same grid. A player scores one point if it takes the ball to its opponent's goals; 2) in thieves and hunters, player A scores one point if two players move to one goal simultaneously (A catches B), otherwise, it loses one point if player B moves to one goal without being caught. If neither player gets a score within 50 steps, the game ends with a tie.

We consider two versions of the above games in both tabular and deep representations. For the tabular version, the state space includes few numbers of discrete states, in which Q-values can be represented in a tabular form. For the deep version, each state consists of different dimensions of information: for example, states in soccer includes coordinates of two agents and the ball possession. In this case, we evaluate the performance of deep Bayes-ToMoP. We manually design six kinds of policies for the opponent in soccer (differentiated by the directions of approaching the goal) and twenty-four kinds of policies for the opponent in thieves and hunters (differentiated by the orders of achieving the goal). A policy library of best-response policies are trained using Q-learning and DQN for Bayes-ToMoP and deep Bayes-ToMoP.
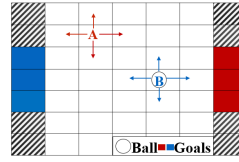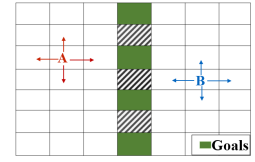

Figure 1: The soccer game.   Figure 2: Thieves and hunters

Opponents / Methods	O_{ToMoP_0}	O_{ns}	O_{ToMoP_0}-s
BPR_s	49.78±1.71%	99.37±0.72%	66.31±0.57%
DRON	74.75±0.19%	76.54±0.16%	75.35±0.18%
Deep BPR+	71.57±1.26%	99.49±0.51%	78.88±0.76%
Bayes-ToMoP$_1$	**99.82±0.18%**	98.21±0.37%	**98.48±0.54%**

Table 1: Average win rates with std.dev.($\pm$) in soccer.

4.2 Performance against Different Opponents

Three kinds of opponents are considered: 1) a Bayes-ToMoP$_0$ opponent (O_{ToMoP_0}); 2) an opponent that randomly switches its policy among stationary strategies and lasts for an unknown number of episodes (O_{ns}) and 3) an opponent switching its strategy between stationary strategies and Bayes-ToMoP$_0$ (O_{ToMoP_0}-s). We assume an opponent only selects a policy from the known policy library. Thus Bayes-Pepper is functionally equivalent to BPR+ in our setting and we use BPR_s to denote both strategies. Due to the space limitation, we only give experiments on the tabular form of Thieves and hunters and a deep version of soccer.

Figure 3 (a) shows that only Bayes-ToMoP$_1$ can quickly detect the opponent's strategies and achieve the highest average rewards. In contrast, BPR_s fails against O_{ToMoP_0}. Similar comparison results can be found for deep Bayes-ToMoP$_1$ (Figure 3 (b)). This is because Bayes-ToMoP$_1$ explicitly considers two-orders of belief to do recursive reasoning first and then derives an optimal policy against its opponent. However, BPR_s is essentially equivalent to Bayes-ToMoP$_0$, i.e., it is like Bayes-ToMoP$_0$ is under self-play. Therefore, neither BPR_s nor Bayes-ToMoP$_0$ could take advantage of each other and the winning percentages are expected to be approximately 50%. Average win rates shown in Table 1 also confirm our hypothesis. The results for Bayes-ToMoP$_1$ under self-play can be found in the extended version[1]. Deep BPR+ incorporates previous opponent's behaviors into BPR, however, their model only considers which kind of stationary strategy the opponent is using, thus is not enough to detect the policy of opponent O_{ToMoP_0}. Figure 3 (b) shows
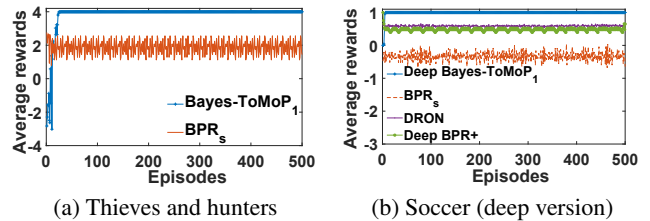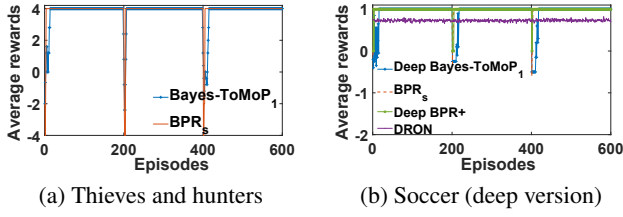

(a) Thieves and hunters   (b) Soccer (deep version)

Figure 3: Against an opponent O_{ToMoP_0} on different games.

(a) Thieves and hunters (b) Soccer (deep version)

Figure 4: Against an opponent O_{ns} on different games.

(a) Thieves and hunters (b) Soccer (deep version)

Figure 5: Against an opponent O_{ToMoP_0-s} on different games.

(a) Thieves and hunters (b) Soccer (deep version)

Figure 6: Against an opponent using a new policy on two games.

DRON performs better than Deep BPR+ and BPR$_s$. However, it fails to achieve the highest average rewards because the dynamic adjustment of DRON cannot guarantee that the optimality against a particular type of opponents.

Next, we present the results of playing against O_{ns} that switches its policy among stationary strategies and lasts for 200 episodes. Figure 4 (a) shows the comparison of Bayes-ToMoP$_1$ with BPR$_s$, where both methods can quickly and accurately detect which stationary strategy the opponent is using and derive the optimal policy against it. We observe that Bayes-ToMoP$_1$ requires longer time to detect than BPR$_s$ and similar comparison is found in Figure 4 (b) that deep Bayes-ToMoP$_1$ requires longer time to detect than BPR$_s$ and deep BPR+. This happens because 1) Bayes-ToMoP$_1$ needs longer time to determine that the opponent is not using a BPR-based strategy; 2) some stochastic factors such as the random initialization of belief. This phenomenon is consistent with the slightly lower win rate of Bayes-ToMoP$_1$ than BPR$_s$ and deep BPR+ (Table 1). However, DRON only achieves the average rewards of 0.7 since it uses end-to-end trained response subnetworks, which cannot guarantee that each response is good enough against a particular type of opponent.

Finally, we consider the case of playing against O_{ToMoP_0}-s to show the robustness of Bayes-ToMoP. Figure 5 (a) shows that only Bayes-ToMoP$_1$ can quickly and accurately detect the strategies of both opponent O_{ToMoP_0} and non-stationary opponent. In contrast, BPR$_s$ fails when its opponent's strategy switches to Bayes-ToMoP$_0$. A similar comparison can be found in soccer shown in Figure 5 (b) in which both BPR$_s$ and deep BPR+ fail to detect and respond to O_{ToMoP_0}-s opponent. Figure 5 (b) also shows DRON performs poorly against both kinds of opponents due to the similar reason described above. Average win rates are summarized in Table 1 which are consistent with the results in Figure 5 (b).

4.3 New Opponent Detection and Learning

In this section, we evaluate Bayes-ToMoP$_1$ against an opponent who may use previously unknown strategies. We manually design a new opponent in soccer and thieves and hunters that stars with one of the known strategies and switches to the new one at the 200th episode. Figure 6 shows the average rewards of different approaches on two games respectively.

Figure 6 (a) shows that when the opponent switches to an unknown strategy, both Bayes-ToMoP$_1$ and BPR+ can quickly detect this change, and finally learn an optimal policy. However, Bayes-Pepper fails. This is because Bayes-Pepper predicts the strategy of its opponent from a known policy library, which makes it fail to well respond to a previously un-
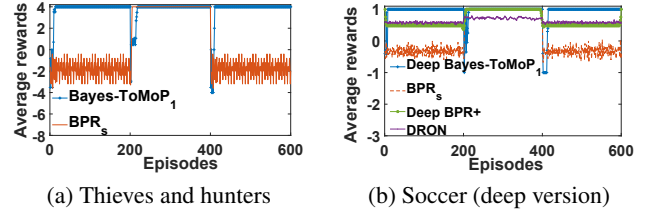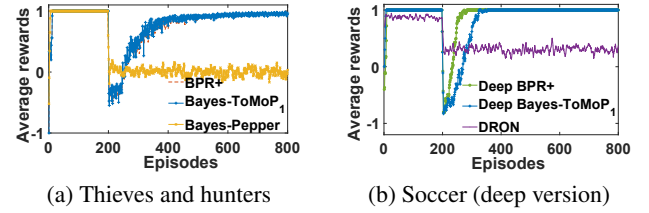
seen strategy. The learning curve of Bayes-ToMoP$_1$ is closer to BPR+ since both methods learn from scratch. Similar results can be found for deep Bayes-ToMoP$_1$ and BPR+ in Figure 6 (b). DRON fails against the unknown opponent strategy due to the fact that the number of expert networks is fixed and thus unable to handle such a case. Deep BPR+ performs better than the other three methods since it uses policy distillation to transfer knowledge from similar previous policies to accelerate online learning, which can be readily applied to our Bayes-ToMoP to accelerate online learning as future work.

5 Conclusion and Future Work

This paper presents a novel algorithm called Bayes-ToMoP to handle not only switching, non-stationary opponents and also more sophisticated ones (e.g., BPR-based). Bayes-ToMoP also enables an agent to learn a new optimal policy when encountering a previously unseen strategy. Theoretical guarantees are provided for the optimal detection of the opponent's strategies. Extensive simulations show Bayes-ToMoP outperforms state-of-the-art approaches both in tabular and deep learning environments. As future work, it is worth investigating how to accelerate the online new policy learning phase and how to extend Bayes-ToMoP to multi-opponent scenarios. Furthermore, higher orders of Bayes-ToMoP are worth investigating to handle more sophisticated opponents and apply to large scale, real scenarios.

References

[Albrecht and Stone, 2018] Stefano V. Albrecht and Peter Stone. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95, 2018.

[Baker *et al.*, 2011] Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. Bayesian theory of mind: Modeling joint belief-desire attribution. In *Proceedings of Annual Meeting of the Cognitive Science Society*, 2011.

[Brafman and Tennenholtz, 2002] Ronen I. Brafman and Moshe Tennenholtz. R-MAX - A general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2002.

[Crandall, 2012] Jacob W. Crandall. Just add pepper: extending learning algorithms for repeated matrix games to repeated markov games. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems*, pages 399–406, 2012.

[de Weerd *et al.*, 2013] Harmen de Weerd, Rineke Verbrugge, and Bart Verheij. How much does it help to know what she knows you know? an agent-based simulation study. *Artificial Intelligence*, 199:67–92, 2013.

[Fernández *et al.*, 2010] Fernando Fernández, Javier García, and Manuela M. Veloso. Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems*, 58(7):866–871, 2010.

[Foerster *et al.*, 2018] Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of International Conference on Autonomous Agents and MultiAgent Systems*, pages 122–130, 2018.

[Gmytrasiewicz and Doshi, 2005] Piotr J. Gmytrasiewicz and Prashant Doshi. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24:49–79, 2005.

[Goodrich *et al.*, 2003] Michael A. Goodrich, Jacob W. Crandall, and Jeffrey L. Stimpson. Neglect tolerant teaming: Issues and dilemmas. In *AAAI Spring Sympo. on Human Interaction with Autonomous Systems in Complex Environments*, 2003.

[He and Boyd-Graber, 2016] He He and Jordan L. Boyd-Graber. Opponent modeling in deep reinforcement learning. In *Proceedings of International Conference on Machine Learning*, pages 1804–1813, 2016.

[Hernandez-Leal and Kaisers, 2017] Pablo Hernandez-Leal and Michael Kaisers. Towards a fast detection of opponents in repeated stochastic games. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems*, pages 239–257, 2017.

[Hernandez-Leal *et al.*, 2016] Pablo Hernandez-Leal, Matthew E. Taylor, Benjamin Rosman, Luis Enrique Sucar, and Enrique Munoz de Cote. Identifying and tracking switching, non-stationary opponents: A bayesian approach. In *Multiagent Interaction without Prior Coordination, Papers from the 2016 AAAI Workshop*, 2016.

[Hernandez-Leal *et al.*, 2017a] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A survey of learning in multiagent environments: Dealing with non-stationarity. *CoRR*, abs/1707.09183, 2017.

[Hernandez-Leal *et al.*, 2017b] Pablo Hernandez-Leal, Yusen Zhan, Matthew E. Taylor, Luis Enrique Sucar, and Enrique Munoz de Cote. Efficiently detecting switches against non-stationary opponents. *Autonomous Agents and Multi-Agent Systems*, 31(4):767–789, 2017.

[Littman, 1994] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of International Conference on Machine Learning*, pages 157–163, 1994.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Powers and Shoham, 2005] Rob Powers and Yoav Shoham. Learning against opponents with bounded memory. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 817–822, 2005.

[Rosman *et al.*, 2016] Benjamin Rosman, Majd Hawasly, and Subramanian Ramamoorthy. Bayesian policy reuse. *Machine Learning*, 104(1):99–127, 2016.

[Shoham and Leyton-Brown, 2009] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[Wunder *et al.*, 2012] Michael Wunder, John Robert Yaros, Michael Kaisers, and Michael L. Littman. A framework for modeling population strategies by depth of reasoning. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 947–954, 2012.

[Yang *et al.*, 2019] Tianpei Yang, Jianye Hao, Zhaopeng Meng, Yan Zheng, Chongjie Zhang, and Ze Zheng. Bayestomop: A fast detection and best response algorithm towards sophisticated opponents. In *Proceedings of International Conference on Autonomous Agents and MultiAgent Systems*, pages 2282–2284, 2019.

[Zheng *et al.*, 2018] Yan Zheng, Zhaopeng Meng, Jianye Hao, Zongzhang Zhang, Tianpei Yang, and Changjie Fan. A deep bayesian policy reuse approach against non-stationary agents. In *Advances in Neural Information Processing Systems*, pages 954–964, 2018.