Deep FTRL-ORW: An Efficient Deep Reinforcement Learning Algorithm for Solving Imperfect Information Extensive-Form Games

Linjian Meng[1], **Zhenxing Ge**[1], **Pinzhuo Tian**[2], **Bo An**[3], **Yang Gao**[1*]

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[2] School of Computer Engineering and Science, Shanghai University, Shanghai, China
[3] School of Computer Science and Engineering, Nanyang Technological University, Singapore
{menglinjian, zhenxingge}@smail.nju.edu.cn, pinzhuo@shu.edu.cn, boan@ntu.edu.sg, gaoy@nju.edu.cn

Abstract

One of the most popular methods for learning Nash equilibrium (NE) in large-scale imperfect information extensive-form games (IIEFGs) is the neural variants of *counterfactual regret minimization (CFR)*. CFR is a special case of *Follow-The-Regularized-Leader (FTRL)*. At each iteration, the neural variants of CFR update the agent's strategy via the estimated counterfactual regrets. Then, they use neural networks to approximate the new strategy, which incurs an approximation error. These approximation errors will accumulate since the counterfactual regrets at iteration t are estimated using the agent's past approximated strategies. Such accumulated approximation error causes poor performance. To address this accumulated approximation error, we propose a novel FTRL algorithm called *FTRL-ORW*, which does not utilize the agent's past strategies to pick the next iteration strategy. More importantly, FTRL-ORW can update its strategy via the trajectories sampled from the game, which is suitable to solve large-scale IIEFGs since sampling multiple actions for each information set is too expensive in such games. However, it remains unclear which algorithm to use to compute the next iteration strategy for FTRL-ORW when only such sampled trajectories are revealed at iteration t. To address this problem and scale FTRL-ORW to large-scale games, we provide a model-free method called *Deep FTRL-ORW*, which computes the next iteration strategy using model-free *Maximum Entropy Deep Reinforcement Learning*. Experimental results on two-player zero-sum IIEFGs show that Deep FTRL-ORW significantly outperforms existing model-free neural methods and *OS-MCCFR*.

Introduction

Imperfect information extensive-form games (IIEFGs) are a standard class of games that can model multiple agents, imperfect information, and random events. IIEFGs have been widely used for modeling real-world situations, such as medical treatment (Sandholm 2015), security games (Lisỳ, Davis, and Bowling 2016), cybersecurity games (Chen et al. 2017), and recreational games (Brown and Sandholm 2018, 2019). For these games, the common goal is to find Nash equilibrium (NE) since it prescribes a notion of rational behavior in which no player can benefit from deviating unilaterally from the equilibrium.

[*]Corresponding Author.

To learn NE in IIEFGs, one of the most powerful methods is the *counterfactual regret minimization (CFR) (Zinkevich et al. 2007)* algorithm. CFR is a special case of the *Follow-The-Regularized-Leader (FTRL)* (Shalev-Shwartz and Singer 2007) algorithm which utilizes the *FD dilated distance generating function (FD dilated DGF)* (Liu et al. 2022). CFR learns NE by iteratively traversing the entire game and minimizing the regret on each information set (infoset). Due to the large-scale state space in most real-world scenarios, it is impossible to traverse the entire game tree and use tables to represent strategies. To sidestep the issue, many neural variants of CFR have been proposed (Brown et al. 2019; Li et al. 2019; Gruslys et al. 2020; Hennes et al. 2020; Steinberger, Lerer, and Brown 2020; Fu et al. 2021; McAleer et al. 2022). At each time, they estimate the counterfactual regrets and update the strategy using the estimated regrets. Then, they approximates the new strategy via neural networks. However, each approximation incurs an approximation error. Since the counterfactual regrets at iteration t are estimated with the agent's previous approximated strategies, these errors accumulate. Due to this accumulated approximation error, these methods have poor performance. The counterfactual regrets are related to the FD dilated DGF. Unfortunately, the FD dilated DGF at iteration t is dependent on the agent's past strategies, which is the essential reason why the neural variants of CFR suffer from such accumulated approximation error.

To address the accumulated approximation error, we propose to combine neural networks with FTRL cases whose DGF at iteration t is independent of the agent's previous strategies. To achieve this goal, we provide our first key contribution – a novel DGF called *Opponent Related Dilated DGF (ORD-DGF)*, which is unrelated to the agent's past strategies. More importantly, the value of ORD-DGF for a strategy can be estimated via the trajectories sampled from the game. It allows the *FTRL with ORD-DGF (FTRL-ORW)* algorithm to update its strategy via such sampled trajectories. This property enables FTRL-ORW to be suitable for solving large-scale games since sampling multiple actions for each information set is too expensive in such games. However, it remains unclear how to compute the next iteration strategy for FTRL-ORW if we only obtain such sampled trajectories at iteration t. To address this problem and scale FTRL-ORW to large-scale games, we make our second key contribution – the *Deep FTRL-ORW* algorithm. Deep FTRL-ORW employs the model-free *Maximum Entropy Deep Reinforcement Learning (MEDRL)* algorithms (Haarnoja et al. 2018) to compute the next iteration strategy of FTRL-ORW. Deep FTRL-ORW is a model-free method, which learns from the sampled trajectories. Third, we prove that FTRL-ORW and Deep FTRL-ORW converge to an $O(\sqrt{\log T/T^{2/3}})$-NE and $O(\sqrt{\log T/T^{2/3}} + E)$-NE in two-player zero-sum IIEFGs with perfect recall respectively, where T is the number of iterations and E is the solution error bound of MEDRL. Note that the solution error at iteration t only has a one-time impact on convergence performance to NE, not affecting the subsequent updates. In contrast, the approximation error of the neural variants of CFR at iteration t affects all subsequent updates. In other words, it impacts the convergence performance multiple times. Finally, experimental results on three standard IIEFG benchmarks, *i.e.*, Kuhn Poker, Leduc Poker, and Goofspiel, demonstrate that FTRL-ORW achieves competitive performance compared with CFR, and Deep FTRL-ORW has the lowest exploitability compared with other model-free neural methods and OS-MCCFR (Lanctot et al. 2009). In experiments on larger games, such as phantom tic tac toe and dark hex, Deep FTRL-ORW achieves higher utility than existing model-free neural methods.

Related Work

In this paper, we focus on learning NE in two-player zero-sum IIEFGs with perfect recall. Many methods have been proposed to learn an approximate NE in such games. These methods usually need to traverse the entire game tree. However, due to the large size in most real-world scenarios, it is impossible to traverse the entire game. Therefore, many stochastic methods have been proposed, which sample actions at each infoset to reduce the computation overhead.

Population Learning Methods These methods are the variants of *Policy-Space Response Oracles (PSRO)* (Lanctot et al. 2017). They store past policies and a meta-distribution over them, incrementally adding new policy to the policy set by computing the best response to a mixture of the past policies. *Neural Fictitious Self-Play (NFSP)* (Heinrich and Silver 2016) can be seen as a special case of PSRO where the meta-distribution is the uniform distribution. These methods can scale to large-scale games since they only require simulation of the policies and aggregating data (Vinyals et al. 2019). However, their rate of convergence is usually associated with the size of the strategy space of the game. Thus, they may have poor performance in IIEFGs with a large branch factor and long horizon since the strategy space of such games is too large.

Tabular Regret Minimization Methods These methods (Lanctot et al. 2009; Farina, Kroer, and Sandholm 2020; Farina, Schmucker, and Sandholm 2021; Farina and Sandholm 2021; Kozuno et al. 2021; Bai et al. 2022) estimate the current loss gradient and feed the estimated loss gradient to a full feedback regret minimizer, such as CFR (Zinkevich et al. 2007) and OMD (Duchi, Hazan, and Singer 2011). These methods converge to an $O(\sqrt{1/T})$-NE with probability $1-p$ ($p \in (0,1)$), where T is the number of iterations. However, they are tabular methods, and the expensive memory overhead limits their application in large-scale games.

Neural Variants of CFR These methods are also regret minimization methods. Although tabular regret minimization methods reduce the computation overhead, they still suffer from the large memory overhead. Therefore, many neural variants of CFR have been proposed. They approximate the behaviour of CFR via neural networks to scale to large-scale games (Brown et al. 2019; Li et al. 2019; Steinberger 2019; Gruslys et al. 2020; Hennes et al. 2020; Fu et al. 2021; McAleer et al. 2022). At each iteration, these methods estimate the counterfactual regrets and update the strategy using the estimated counterfactual regrets. Then, they approximate the new strategy via neural networks. However, such approximation incurs an approximation error at each iteration. Moreover, they estimate the counterfactual regrets at iteration t according to the previous approximated strategies, which accumulate these errors. To address this accumulated error, we propose a new tabular regret minimization method and approximate its behavior via neural networks.

Preliminaries

In this section, we describe the necessary terminology. First, we introduce some concepts about extensive-forms games. Then we illustrate the procedure of learning Nash equilibrium (NE) via regret minimization methods.

Extensive-Form Games

Imperfect Information Extensive-form Games (IIEFGs) IIEFGs are a model of sequential interaction involving multiple agents and represented by a tree rooted at the root node r. Each node h in the tree belongs to a player from the set $\{0, 1, c\}$, where 0 is the *min* player, 1 is the *max* player, and c is the *chance* player. We use $\mathcal{H}_i$ to represent the set of nodes belonging to player i. The notation $A(h)$ denotes the actions available at node h. Each node z such that $A(z) = \emptyset$ is called leaf node and represents a terminal state of the game. The set of leaf nodes is denoted by $\mathcal{Z}$. For each leaf node z, there is a pair $(v_0(z), v_1(z)) \in \mathbb{R}^2$ which denotes the payoffs for the min player (player 0) and the max player (player 1) respectively. In our setting, $v_0(z) = -v_1(z)$ for all $z \in \mathcal{Z}$. To represent the private information, the nodes for each player i are partitioned into a collection $\mathcal{I}_i$, called information sets (infosets). Perfect recall means that no player will forget any information that has been revealed.

Behavioral Strategy A behavioral strategy σ_i for player i is defined on each infoset. For any infoset $I \in \mathcal{I}_i$, the probability for the action $a \in A(I)$ is denoted by $\sigma_i(I, a)$. The strategy other than player i is denoted by σ_{-i}. If every player plays according to σ and reach infoset I, the reaching probability is denoted by $\pi^\sigma(I)$. The contribution of i to this probability is $\pi_i^\sigma(I)$ and $\pi_{-i}^\sigma(I)$ for other than i. We use $\sigma_i(\cdot|I) \in \Delta^{A(I)}$ to denote the probability distribution over the actions at infoset I, where $\Delta^{A(I)}$ is an n-dimensional simplex ($n = A(I) - 1$).

Sequence-Form Strategy A sequence is an infoset-action pair (I, a), where I is an infoset and a is an action belonging to $A(I)$. Each sequence identifies a path from the root node to the infoset I and selects the action a. The set of sequences for player i is denoted Σ_i. The last sequence encountered on the path from r to I is denoted by ρ_I. Sequence-form strategy for player i is a non-negative vector $\boldsymbol{x}$ indexed over the set of sequences Σ_i. For each sequence $q = (I, a) \in \Sigma_i$, $\boldsymbol{x}_q$ is the probability that player i reaches the sequence q if she follows the strategy $\boldsymbol{x}$. In this paper, we formulate the sequence-form strategy space as a treeplex (Hoda et al. 2010). Let $\mathcal{X}$ and $\mathcal{Y}$ denote the set of sequence-form strategies for the min player and the max player. They are nonempty convex compact sets in Euclidean spaces $\mathbb{E}_x$, $\mathbb{E}_y$. We use $\boldsymbol{x}_I$ to denote the slice of a given strategy $\boldsymbol{x}$ corresponding to sequences belonging to infoset I. It has been proved that a behavioral strategy is realization–equivalent to a sequence-form strategy in IIEFGs with perfect recall (Von Stengel 1996). More details are in Appendix.

We use different strategy representations in different cases since the two representations have their benefits and limitations. For convenience, we provide a notation conversion table between the two strategy representations in Appendix.

Regret Minimization Methods

Regret This concept is from the online convex optimization framework (Zinkevich 2003) where a decision-maker selects a point $\boldsymbol{u}^t$ from the convex compact set $\mathcal{U} \in \mathbb{R}^d$ and faces a loss gradient $\boldsymbol{\ell}^t \in \mathbb{R}^d$ at each iteration t. The regret $R^T(\boldsymbol{u}) = \sum_{t=1}^{T} \langle \boldsymbol{\ell}^t, \boldsymbol{u}^t - \boldsymbol{u} \rangle$ denotes the difference between the accumulative loss gradient of the sequence output $\boldsymbol{u}^1, \cdots, \boldsymbol{u}^T$ and the loss gradient of a fixed point $\boldsymbol{u}$.

In two-player zero-sum IIEFGs with perfect recall, Nash equilibrium (NE) can be formulated as the solution to a *Bilinear Saddle Point Problem (BSPP)* (also called the sequence-form representation):

$$\underset{\boldsymbol{x}\in\mathcal{X}}{min}\, \underset{\boldsymbol{y}\in\mathcal{Y}}{max}\, \langle \boldsymbol{x}, \boldsymbol{A}\boldsymbol{y} \rangle = \underset{\boldsymbol{y}\in\mathcal{Y}}{max}\, \underset{\boldsymbol{x}\in\mathcal{X}}{min}\, \langle \boldsymbol{x}, \boldsymbol{A}\boldsymbol{y} \rangle, \tag{1}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the sequence-form strategies w.r.t the min player and the max player respectively, and $\boldsymbol{A}$ is the payoff matrix. The accuracy of a solution $[\boldsymbol{x}; \boldsymbol{y}] \in \mathcal{X} \times \mathcal{Y}$ is quantified by *saddle-point gap*

$$\varepsilon([\boldsymbol{x}; \boldsymbol{y}]) = \underset{\hat{\boldsymbol{y}}\in\mathcal{Y}}{max}\, \langle \boldsymbol{x}, \boldsymbol{A}\hat{\boldsymbol{y}} \rangle - \underset{\hat{\boldsymbol{x}}\in\mathcal{X}}{min}\, \langle \hat{\boldsymbol{x}}, \boldsymbol{A}\boldsymbol{y} \rangle. \tag{2}$$

Learning NE via Regret Minimization To learn NE, we wish to find a pair $[\boldsymbol{x}; \boldsymbol{y}]$ whose saddle-point gap converges to 0. Regret Minimization methods can be used to achieve this goal. The first step is to instantiate two regret minimizers R0 and R1 for strategy sets $\mathcal{X}$ and $\mathcal{Y}$. Then at each iteration t, regret minimizers R0 and R1 face the loss gradients $\boldsymbol{\ell}_0^t$ and $\boldsymbol{\ell}_1^t$, and output the strategies $\boldsymbol{x}^{t+1}$ and $\boldsymbol{y}^{t+1}$. Then they are used to produce the loss gradients $\boldsymbol{\ell}_0^{t+1} = \boldsymbol{A}\boldsymbol{y}^{t+1}$, and $\boldsymbol{\ell}_1^{t+1} = -\boldsymbol{A}^T\boldsymbol{x}^{t+1}$ at the next iteration $t+1$. Let $\bar{\boldsymbol{x}}$, $\bar{\boldsymbol{y}}$ denotes the average strategies output by R0 and R1. A folk theorem shows

$$\varepsilon([\bar{\boldsymbol{x}}; \bar{\boldsymbol{y}}]) \le (R_0^T + R_1^T)/T. \tag{3}$$

FTRL-ORW

In this section, we first describe FTRL. Second, we illustrate why the neural variants of CFR suffer from the accumulated approximation error. Then, to address this accumulated error, we propose a novel DGF called *Opponent Related Dilated Distance Generating Function (ORD-DGF)*. ORD-DGF is unrelated to the agent's previous strategies. In addition, ORD-DGF can be estimated from sampled trajectories. It allows the *FTRL with ORD-DGF (FTRL-ORW)* algorithm to update its strategy from sampled trajectories, which enables FTRL-ORW to scale to large-scale IIEFGs. Finally, we prove the convergence of the FTRL-ORW algorithm.

Follow-The-Regularized-Leader

Follow-The-Regularized-Leader (FTRL) is one of the most prominent regret minimization methods. Assuming we are the min player, at each iteration t, FTRL minimizes the sum of the accumulative loss gradient with a regularization term (also called *mirror-mapping operator*)

$$\boldsymbol{x}^{t+1} \in \underset{\boldsymbol{x}\in\mathcal{X}}{\arg min} \{ \langle \boldsymbol{\ell}^{1:t}, \boldsymbol{x} \rangle + \frac{1}{\eta} d^t(\boldsymbol{x}) \}, \tag{4}$$

where $\boldsymbol{\ell}^{1:t} = \sum_{\tau=1}^{t} \boldsymbol{\ell}^\tau$, $\boldsymbol{\ell}^t = \boldsymbol{A}\boldsymbol{y}^t$, η is the step-size parameter, and $d^t(\boldsymbol{x})$ is the *distance-generating function (DGF)* which is 1-strongly convex w.r.t. a specific norm on the sequence-form strategy polytope. In this paper, we consider a particular type of DGF which has demonstrated SOTA results in solving IIEFGs: *the dilated DGFs* (Hoda et al. 2010; Kroer et al. 2015, 2020; Farina, Kroer, and Sandholm 2019, 2021; Liu et al. 2022). A dilated DGF is constructed by taking a sum over suitable local DGFs for each infoset, where each local DGF is dilated by the parent variable leading to the infoset:

$$d^t(\boldsymbol{x}) = \sum_{I\in\mathcal{I}_0} \boldsymbol{x}_{\rho_I} \beta_I^t d_I \left(\frac{\boldsymbol{x}_I}{\boldsymbol{x}_{\rho_I}} \right), \tag{5}$$

where $d_I(\boldsymbol{x})$ is the local DGF at infoset I and β_I^t is the weight on the local DGF $d_I(\boldsymbol{x})$ at iteration t. From the definition of the sequence-form strategy, we get $\boldsymbol{x}_I/\boldsymbol{x}_{\rho_I} \in \Delta^{A(I)}$. For convenience, $\boldsymbol{x}_I/\boldsymbol{x}_{\rho_I}$ is denoted by $\tilde{\boldsymbol{x}}_I$ in the following. Moreover, we have $\tilde{\boldsymbol{x}}_I = \sigma_0(\cdot|I)$ and $\tilde{\boldsymbol{y}}_I = \sigma_1(\cdot|I)$.

Why Do Neural Variants of CFR Suffer From Accumulated Approximation Error

In this subsection, we show why the neural variants of CFR suffer from the accumulated approximation error. We first describe *Counterfactual Regret Minimization (CFR)*. CFR is a special case of FTR which utilizes the *FD dilated DGF* (Liu et al. 2022). Formally, assuming we are the min player, the FD dilated DGF sets the weight β_I^t as

$$\beta_I^t = \sum_{a\in A(I)} \left[\sum_{\tau=1}^{t} \boldsymbol{y}^\tau[I]\boldsymbol{c}[I] \left(\sum_{z\in C_{Ia}} \frac{\boldsymbol{x}_{\rho_z}^\tau \boldsymbol{y}^\tau[z]\boldsymbol{c}[z]}{\boldsymbol{x}_{Ia}^\tau \boldsymbol{y}^\tau[I]\boldsymbol{c}[I]} v_0(z) - \sum_{a\in A(I)} \sum_{z\in C_{Ia}} \frac{\boldsymbol{x}_{\rho_z}^\tau \boldsymbol{y}^\tau[z]\boldsymbol{c}[z]}{\boldsymbol{x}_{Ia}^\tau \boldsymbol{y}^\tau[I]\boldsymbol{c}[I]} v_0(z) \right) \right]^+, \tag{6}$$

where $\boldsymbol{y}^\tau[I]$ ($\boldsymbol{y}^\tau[z]$) is the probability that the max player reaches the infoset I (the leaf node z) if she follows the strategy $\boldsymbol{y}^\tau$, $\boldsymbol{c}[I]$ ($\boldsymbol{c}[z]$) is the probability that the chance player reaches the infoset I (the leaf node z), $[x]^+ = \max(x, 0)$, and C_{Ia} is the set of leaf nodes that the player may encounter if she selects action a at infoset I.

At each iteration t, to pick the next iteration strategy, CFR computes the local DGF weight β_I^t at each infoset I via the agent's past strategies $\boldsymbol{x}^1, \cdots, \boldsymbol{x}^t$. The neural variants of CFR approximate these strategies via neural networks, which incurs approximation errors. These approximation errors accumulate since the weight β_I^t is estimated via past approximated strategies and the estimated weight is used to update the agent's strategy.

Opponent Related Dilated DGF

To address the accumulated approximation error, we propose a dilated DGF called *Opponent Related Dilated DGF (ORD-DGF)*, which is independent of the agent's past strategies $\boldsymbol{x}^t, \cdots, \boldsymbol{x}^t$. Moreover, the value of ORD-DGF $d(\boldsymbol{x})$ for a strategy $\boldsymbol{x}$ can be estimated from sampled trajectories. The mirror-mapping operator is a single-agent optimization problem. Since the first term accumulated loss gradients and the second term the DGF value of the mirror-mapping operator can be estimated from sampled trajectories, such a single-agent optimization problem can also be addressed from sampled trajectories. It is suitable to solve large-scale games since sampling multiple actions for each infoset is too expensive in such games.

In this subsection, we first describe ORD-DGF, then show why the value of ORD-DGF for a given strategy can be estimated from sampled trajectories while other existing DGFs (except the FD dilated DGF) cannot. Assuming we are the min player, ORD-DGF at iteration t is defined as

$$d^t(\boldsymbol{x}) = \sum_{I \in \mathcal{I}_0} \boldsymbol{x}_{\rho_I} \dot{\boldsymbol{y}}^t[I] \boldsymbol{c}[I] d_I(\tilde{\boldsymbol{x}}_I), \tag{7}$$

and

$$\dot{\boldsymbol{y}}^t[I] = \sum_{\tau=1}^{t} \left[(1-\lambda^\tau)\boldsymbol{y}^\tau[I] + \lambda^\tau \hat{\boldsymbol{y}}[I]\right]/t, \tag{8}$$

where λ^τ is the mutant weight at iteration t, $\hat{\boldsymbol{y}}$ is a fixed strategy, and $\boldsymbol{y}^t[I]$ ($\hat{\boldsymbol{y}}[I]$) is the probability that the max player reaches the infoset I if she follows the strategy $\boldsymbol{y}^t$ ($\hat{\boldsymbol{y}}$). Note that $\boldsymbol{c}[I]$ is known and time-invariant. For the min player, the notion $\dot{\boldsymbol{y}}^t[I]$ and $\mathcal{I}_0$ in Eq. (7) are replaced by $\dot{\boldsymbol{x}}^t[I]$ and $\mathcal{I}_1$, respectively. In this paper, we set d_I as the negative entropy DGF $d(\boldsymbol{u}) = \sum_{i=1}^n \boldsymbol{u}_i \log \boldsymbol{u}_i$ since it provides better convergence for FTRL-ORW compared to other commonly used local DGFs, where n is the dimension of the vector $\boldsymbol{u}$. From the realization equivalence between the behavioral strategy and the sequence-form strategy, we have $\dot{\boldsymbol{y}}^t[I]\boldsymbol{c}[I] = \sum_{h \in I} \dot{\boldsymbol{y}}^t[h]\boldsymbol{c}[h] = \pi_{-0}^{\dot{\sigma}_{-0}^t}(I)$ and $\dot{\boldsymbol{x}}^t[I]\boldsymbol{c}[I] = \sum_{h \in I} \dot{\boldsymbol{x}}^t[h]\boldsymbol{c}[h] = \pi_{-1}^{\dot{\sigma}_{-1}^t}(I)$. Since ORD-DGF is only related to the observed opponent strategies, it does not suffer from the accumulated approximation error.

Now, we illustrate why the value of ORD-DGF for a strategy can be estimated from sampled trajectories while other existing DGFs cannot. Assuming we follow the strategy $\boldsymbol{x}$ and the opponent follows the strategy $\dot{\boldsymbol{y}}^t$, we sample N trajectories from the game. For ORD-DGF, we have $d^t(\boldsymbol{x}) = \mathbb{E}_{\mathbb{I}_n \sim (\boldsymbol{x}, \dot{\boldsymbol{y}}^t, \boldsymbol{c})}[\sum_{I \in \mathbb{I}_n} d_I(\hat{\boldsymbol{x}})/N]$, where $\mathbb{I}_n$ is the n-th sampled trajectory. On the contrary, if we use any other existing DGF except the FD dilated DGF, we have $d^t(\boldsymbol{x}) = \mathbb{E}_{\mathbb{I}_n \sim (\boldsymbol{x}, \dot{\boldsymbol{y}}^t, \boldsymbol{c})}[\sum_{I \in \mathbb{I}_n} j_I d_I(\hat{\boldsymbol{x}})/(N\dot{\boldsymbol{y}}^t[I]\boldsymbol{c}[I])]$, where j_I is a constant related to the agent's decision space. We cannot obtain the value of $\dot{\boldsymbol{y}}^t[I]\boldsymbol{c}[I]$ without traversing the entire game tree or expensive domain knowledge.

Substituting ORD-DGF into Eq. (4), we obtain the *FTRL with ORD-DGF (FTRL-ORW)* algorithm. Assuming we are the min player, at each iteration t, FTRL-ORW picks the next iteration strategy $\boldsymbol{x}^{t+1}$ according to

$$\boldsymbol{x}^{t+1} \in \underset{\boldsymbol{x} \in \mathcal{X}}{\arg min}\{\langle \dot{\boldsymbol{\ell}}^{1:t}, \boldsymbol{x}\rangle + \frac{1}{\eta} \sum_{I \in \mathcal{I}_i} \boldsymbol{x}_{\rho_I} \dot{\boldsymbol{y}}^t[I]\boldsymbol{c}[I] d_I(\tilde{\boldsymbol{x}}_I)\}, \tag{9}$$

where $\dot{\boldsymbol{\ell}}^t = (1-\lambda^t)\boldsymbol{A}\boldsymbol{y}^t + \lambda^t \boldsymbol{A}\hat{\boldsymbol{y}}$ since ORD-DGF is built with the observed opponent's average strategy. In a two-player zero-sum IIEFG with perfect recall, if each player runs FTRL-ORW T iterations, their average strategy converges to an $O(\sqrt{\log T / T^{2/3}})$-NE. Proof is in Appendix.

Proposition 1 *From the analysis of Hoda et al. (2010), $d(\boldsymbol{x}) = \sum_{I \in \mathcal{I}_0} \boldsymbol{x}_{\rho_I} \hat{\boldsymbol{y}}[I]\boldsymbol{c}[I] d_I(\tilde{\boldsymbol{x}}_I)$ is Υ-strongly convex w.r.t $\|\cdot\|_1$. Thus, $d^t(\boldsymbol{x}) = \sum_{I \in \mathcal{I}_0} \boldsymbol{x}_{\rho_I} \dot{\boldsymbol{y}}[I]\boldsymbol{c}[I] d_I(\tilde{\boldsymbol{x}}_I)$ is $\lambda^t\Upsilon$-strongly convex w.r.t $\|\cdot\|_1$. Let $\lambda^t = t^{-1/3}$, B be a positive constant such that $-d_I(\hat{\boldsymbol{x}}_I) \le B$ for each infoset I, C be the maximum number of nodes contained in any information set. If each player runs FTRL-ORW T iterations ($T \ge 3$), the saddle-point gap of the average strategy $[\bar{\boldsymbol{x}}; \bar{\boldsymbol{y}}]$ is bounded by*

$$\begin{aligned}\varepsilon([\bar{\boldsymbol{x}}; \bar{\boldsymbol{y}}]) \le & 3(\|\boldsymbol{A}\|_\infty \sqrt{|\mathcal{I}_0|} \\ & + \|\boldsymbol{A}^T\|_\infty \sqrt{|\mathcal{I}_1|}) \sqrt{\frac{BC \log T}{\Upsilon T^{2/3}}} + \frac{4\Delta}{T^{1/3}}\end{aligned} \tag{10}$$

Deep FTRL-ORW

The mirror-mapping operator of FTRL-ORW (Eq. (9)) is a single-agent optimization problem that can be addressed from sampled trajectories. In this section, we describe *Deep FTRL-ORW*, which employs the off-policy model-free *Maximum Entropy Deep Reinforcement Learning (MEDRL)* algorithms to address the optimization problem when only sampled trajectories are revealed. We first explain the reason for using MEDRL to compute Eq. (9). Then, we introduce the overall architecture of Deep FTRL-ORW, whose pseudocode is given in Algorithm 1. Finally, we provide the convergence analysis of Deep FTRL-ORW.

Compute Eq. (9) via Maximum Entropy Deep Reinforcement Learning

Assume we are the player i and player $1-i$ follows strategy $\dot{\sigma}_{-i}^t = \sum_{\tau=1}^t [(1-\lambda^\tau)\sigma_{-i}^\tau + \lambda^\tau \hat{\sigma}_{-i}]/t$, where σ_{-i}^t is the strategy of player $1-i$ at iteration t and $\hat{\sigma}$ is a fixed strategy. Computing Eq. (9) at iteration t is to find the optimal solution to a single-agent optimization problem: the

Algorithm 1: Deep FTRL-ORW for an agent

```
Input: $M_{RL}$, $M_{SL}$, $Q_\theta(I,a)$, $A_\phi(a|I)$, $\Pi_\psi(a|I)$, $\hat{\sigma}$
Parameter: $\theta$, $\phi$, $\psi$, $N$, $p$, $T$, $\alpha$, $\lambda_Q$, $\lambda_A$, $\lambda_\Pi$
Output: $\Pi_\psi(a|I)$
 1: $M_{RL} \leftarrow \emptyset$, $M_{SL} \leftarrow \emptyset$
 2: for each iteration $t = 1, \cdots, T$ do
 3:     $\omega \leftarrow \frac{\alpha}{t}$, $\phi_t \leftarrow \phi$
 4:     for each episode $e = 1, \cdots, N$ do
 5:         Sample $j \sim$ UNIF$(1, \cdots, t)$
 6:         $\sigma = \begin{cases} A_\phi(a|I), \text{with probability } p \\ \quad \hat{\sigma} \quad, \text{with probability } \lambda^j(1-p) \\ A_{\phi_j}(a|I), \text{else} \end{cases}$
 7:         for each environment step $h$ do
 8:             Observe information set $I_l$ and reward $r_l$
 9:             Sample action $a_l$ from $\sigma$
10:             Execute action $a_l$
11:             Observe next infoset $I_{l+1}$ and reward $r_{l+1}$
12:             Store transition $(I_l, a_l, r_{l+1}, I_{l+1})$ in $M_{RL}$
13:             if agent selects action by $A_\phi(a|I)$ then
14:                 Store transition $(I_l, a_l)$ in $M_{SL}$
15:             end if
16:         end for
17:         $\theta \leftarrow \theta - \lambda_Q \hat{\nabla} J_Q(\theta)$
18:         $\phi \leftarrow \phi - \lambda_A \hat{\nabla} J_A(\phi)$
19:         $\psi \leftarrow \psi - \lambda_\Pi \hat{\nabla} J_\Pi(\psi)$
20:     end for
21: end for
22: return $\Pi_\psi(a|I)$
```

agent i selects an action a_l at an infoset I_l, then transitions to an infoset I_{l+1} or a leaf node z, and receives a reward $v_i(z) - \log \sigma_i(I_l, a_l)/(t\eta)$ or $-\log \sigma_i(I_l, a_l)/(t\eta)$, respectively. If the agent transitions to a new infoset, it continues to choose an action, otherwise terminates. The optimal solution to this single-agent optimization problem is defined as

$$\sigma_i^{t+1} \in \arg\underset{\sigma_i}{max} \sum_l \mathbb{E}_{(z,I_l,a_l)\sim(\sigma_i,\dot{\sigma}_{-i}^t)}[v_i(z) - \frac{1}{t\eta}\log \sigma_i(I_l, a_l)], \tag{11}$$

Such a single-agent optimization problem is a *Partially Observable Markov Decision Process (POMDP)*. The most popular method for solving POMDPs is the *Deep Reinforcement Learning (DRL)* algorithms. Since the rewards in such a single-agent optimization problem are related to the term $\log \sigma_i(I_l, a_l)$, we employ the *Maximum Entropy Deep Reinforcement Learning (MEDRL)* algorithms to solve this optimization problem. The objective of MEDRL algorithms is to learn a strategy σ_i that maximizes the expected sum of the general entropy rewards:

$$\sigma_i^* \in \arg\underset{\sigma_i}{min} \sum_l \mathbb{E}_{(z,h_l,a_l)\sim(\sigma_i,\dot{\sigma}_{-i}^t))}[v_i(z) - \omega \log \sigma_i(h_l, a_l)]. \tag{12}$$

where h_l is the encountered node. Apparently, if we set $\omega = \frac{1}{t\eta}$, the optimal solution to Eq. (12) is equal to the optimal solution to Eq. (11) in *perfect information extensive-form games (PIEFGs)*. In other words, the solution to FTRL-ORW's mirror-mapping operator is equal to the solution to MEDRL algorithms in PIEFGs. Actually, it has been proven that MEDRL converges in PIEFGs (Geist, Scherrer, and Pietquin 2019). Although we consider IIEFGs rather than PIEFGs, we directly utilize the MEDRL algorithms to compute Eq. (11).

To compute the optimal solution to the mirror-mapping operator of FTRL-ORW (Eq. (9)) using MEDRL, each Deep FTRL-ORW agent employs two neural networks, a soft Q-function $Q_\theta(I, a)$, and a current MEDRL strategy $A_\phi(a|I)$. The parameters of these networks are θ and ϕ, respectively. In addition, each agent uses a replay buffer $\mathcal{M}_{RL}$ to memorize its experience of game transitions. In training, each agent alternates between optimizing both networks with stochastic gradient descent. Precisely, $Q_\theta(I, a)$ is trained to minimize the soft Bellman residual

$$J_Q(\theta) = \mathbb{E}_{(I,a,r,I')\sim\mathcal{M}_{RL}}[\frac{1}{2}(Q_\theta(I,a) - \hat{Q}(I,a))]^2, \tag{13}$$

with

$$\hat{Q}(I,a) = r + \sum_{a' \in I'} (Q_\theta(I', a') - \omega \log N_a(I', a')), \tag{14}$$

and $A_\phi(a|I)$ is trained to minimize the following objective

$$J_A(\phi) = \mathbb{E}_{(I)\sim\mathcal{M}_{RL}}[-\sum_{a\in I} A_\phi(a|I)(Q_\theta(I,a) - \omega A_\phi(a|I))]. \tag{15}$$

To reduce the sample complexity, we want all players to learn simultaneously while playing against each other. In this case, all agents follow their average strategies. However, the agent still needs to sample additional trajectories to track the current MEDRL strategies for the training of the average strategy network. To address this problem, Deep FTRL-ORW employs *anticipatory dynamics* (Shamma and Arslan 2005) to sample its current MEDRL strategy and track changes in the opponent's behavior simultaneously. Precisely, at each iteration t, each agent i follows the mixture strategy $\sigma_i = (1-\delta)\dot{\sigma}_i^t + \delta\sigma_i^t$ rather than the strategy $\dot{\sigma}_i^t$, where $\delta \in [0, 1]$ is the *anticipatory parameter* and σ_i^t is the current MEDRL strategy. If we use anticipatory dynamics, notice that only the **off-policy** MEDRL algorithms are feasible since the agent samples the trajectories according to the mixture strategy $\sigma_i = (1-p)\dot{\sigma}_i^t + p\sigma_i^{t,\prime}$ rather than the current strategy $\sigma_i^{t,\prime}$.

Outline of Deep FTRL-ORW

In Deep FTRL-ORW, a player is controlled by a separate Deep FTRL-ORW agent. All Deep FTRL-ORW agents learn from simultaneous play against each other. Each Deep FTRL-ORW agent memorizes its experience of game transitions and its current MEDRL strategy in two distinct replay buffers M_{RL} and M_{SL}. Each Deep FTRL-ORW agent has three networks, $Q_\theta(I, a)$, $A_\phi(a|I)$, and an average strategy $\Pi_\psi(a|I)$. The parameters of these networks are θ, ϕ, and ψ.

The first two networks and the replay buffer M_{RL} are used for the MEDRL algorithm training to compute Eq. (9). The network $\Pi_\psi(a|I)$ represents the average strategy during the evaluation. It is regularly trained to approximate the average strategy by optimizing the cross-entropy loss of the data stored in M_{SL}

$$J_\Pi(\psi) = \mathbb{E}_{(I,a)\sim\mathcal{M}_{SL}}[-\log N_g(I,a)]. \tag{16}$$

To avoid windowing artifacts caused by sampling from a finite memory, M_{SL} uses reservoir sampling.

As mentioned above, at each iteration t, each Deep FTRl-ORW agent i follows the mixture strategy $\sigma_i = (1-\delta)\dot{\sigma}_i^t + \delta\sigma_i^t$. Whatever the strategy used, the agent stores the transition tuple $(I_l, a_l, r_{l+1}, I_{l+1})$ in M_{RL}. Only when the agent selects an action according to the MEDRL strategy does she store transition tuple (I_l, a_l) in M_{SL}. Note that during training, the average strategy $\bar{\sigma}_i^t$ is implemented by uniformly sampling a strategy from the stored past strategies to address the approximation error caused by approximating the average strategy. During the evaluation, the average strategy is represented by the neural network $\Pi_\psi(a|I)$ to reduce the evaluation overhead.

Convergence of Deep FTRL-ORW

At each iteration t, MEDRL computes Eq. (9) and outputs strategies $\boldsymbol{x}^t$ and $\boldsymbol{y}^t$ for the min player and the max player, respectively. We assume that the optimal solutions of Eq. (9) for the min player and the max player are $\boldsymbol{x}^{t,*}$ and $\boldsymbol{y}^{t,*}$, respectively. Then the distance between $\boldsymbol{x}^t$ and $\boldsymbol{x}^{t,*}$ is denoted as $\boldsymbol{\epsilon}_0^t$. Similarly, the distance between $\boldsymbol{y}^t$ and $\boldsymbol{y}^{t,*}$ is denoted as $\boldsymbol{\epsilon}_1^t$. The solution accuracy of MEDRL can be measured by the magnitude of $\boldsymbol{\epsilon}_0^t$ and $\boldsymbol{\epsilon}_1^t$ at each iteration t. Proposition 2 states that the convergence bound of Deep FTRL-ORW is positively related to two terms: (i) the convergence rate of FTRL-ORW (the first term in Eq. (17), and (ii) the solution accuracy of MEDRL (the second term in Eq. (17)).

Proposition 2 *In a two-player zero-sum IIEFG with perfect recall, assume that each player runs Deep FTRL-ORW. Let T denote the number of Deep FTRL-ORW iterations, $[\boldsymbol{x}^t;\boldsymbol{y}^t]$ be the strategy profile output by Maximum Entropy Deep Reinforcement Learning at iteration t, $[\boldsymbol{x}^{t,*};\boldsymbol{y}^{t,*}]$ be the solution to Eq. (9) at iteration t, $[\boldsymbol{x}^t;\boldsymbol{y}^t] - [\boldsymbol{x}^{t,*};\boldsymbol{y}^{t,*}] = [\boldsymbol{\epsilon}_0^t;\boldsymbol{\epsilon}_1^t]$, E be a positive constant such that $\|\boldsymbol{\epsilon}_i^t\|_1 \le E$ for $i \in 0,1$ and all $t \in [T]$. After T iterations ($T \ge 3$), the saddle-point gap is bounded by*

$$\varepsilon([\bar{\boldsymbol{x}};\bar{\boldsymbol{y}}]) \le 3(\|\boldsymbol{A}\|_\infty\sqrt{|\mathcal{I}_0|} + \|\boldsymbol{A}^T\|_\infty\sqrt{|\mathcal{I}_1|})\sqrt{\frac{BC\log T}{\Upsilon T^{2/3}}} + \frac{4\Delta}{T^{1/3}} + (\|\boldsymbol{A}\|_\infty + \|\boldsymbol{A}^T\|_\infty)E, \tag{17}$$

where B, C, Υ are defined in Proposition 1, $\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}$ are the average strategies of the min player and the max player respectively.

Experiments

In this section, we evaluate the performance of Deep FTRL-ORW through extensive experiments. The experimental results of FTRL-ORW are provided in Appendix. In this section, we first demonstrate the empirical convergence to approximate NE for Deep FTRL-ORW on three standard IIEFG benchmarks, *i.e.*, Kuhn Poker, Leduc Poker, and Goofspiel(5), where Goofspiel(5) represents the number of cards is five. In this case, *exploitability* is employed as the performance metric that measures the gap to NE. It is half of the saddle-point gap. Then, we conduct experiments on large games of phantom tic tac toe and dark hex. Due to the large size of these games, we compare performance by playing against the random agent. All tested games are provided by OpenSpiel (Lanctot et al. 2019). All tested neural network algorithms have similar network structures in experiments, with 128 neurons in the hidden layer. All experiments run on a computer with four 20-core 3.10GHz CPUs, 394 GB memory, and two RTX3060 GPUs. Our code is available at https://github.com/menglinjian/Deep-FTRL-ORW.

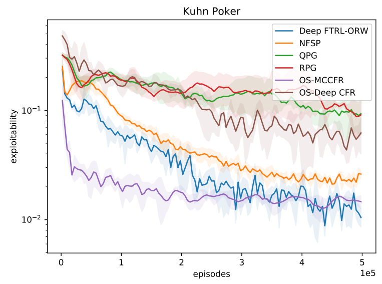
Convergence to Equilibrium

Configurations In this subsection, we investigate the empirical convergence of Deep FTRL-ORW to NE on three standard IIEFG benchmarks, *i.e.*, Kuhn Poker, Leduc Poker, and Goofspiel(5). We compare Deep FTRL-ORW with other model-free methods, such as NFSP, QPG/ RPG (Srinivasan et al. 2018), OS-Deep CFR, and OS-MCCFR (Lanctot et al. 2009). OS-Deep CFR is a special case of Deep CFR (Brown et al. 2019) which uses outcome sampling. Actually, OS-Deep CFR is the tabular version of OS-MCCFR. The implementations of NFSP and two PG algorithms are provided by OpenSpiel. In Kuhn Poker and Leduc Poker, the hyperparameters of NFSP are tuned from the recommendation of the OpenSpiel, and the hyperparameters of the two PG algorithms are provided by Farina and Sandholm (2021). In Goofspiel(5), the hyperparameters of tested neural network algorithms are fine-tuned from the hyperparameters used in Leduc Poker. The hyperparameters are shown in Appendix. The exploration term of OS-MCCFR is 0.1 which is provided by Farina and Sandholm (2021).

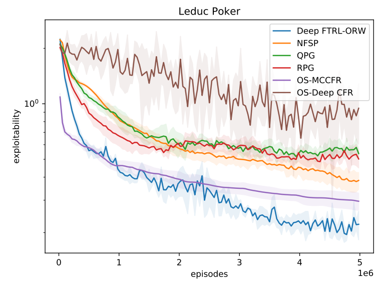
Results We run each algorithm four times with different random seeds. The results are shown in Figure 1. Deep FTRL-ORW provides the lowest exploitability in all tested games. Two PG algorithms show poor performance, which is consistent with the results in Farina and Sandholm (2021). We guess this is because they are too sensitive to hyperparameters. Compared with NFSP, Deep FTRL-ORW usually requires around 40-50% of the episodes to achieve results similar to NFSP in all tested games. It is consistent with the theory that the theoretical convergence of Deep FTRL-ORW is much better than NFSP. Compared with OS-MCCFR, Deep FTRL-ORW performs worse initially and has lower exploitability at the end. Our intuition is that the high estimation variance incurred by importance sampling causes the poor performance in the end. OS-Deep CFR shows much poorer performance than OS-MCCFR which might be caused by the accumulated approximation error.

Performance Against Random Agent

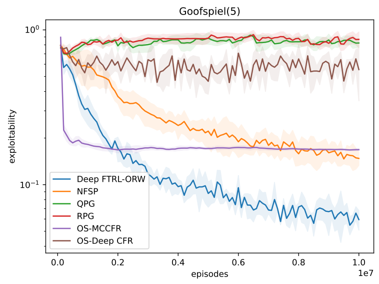
In this subsection, we show the performance of Deep FTRL-ORW in larger games, such as Phantom Tic Tac Toe



(a) Kuhn Poker

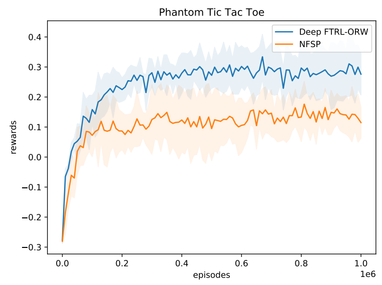


(b) Leduc Poker

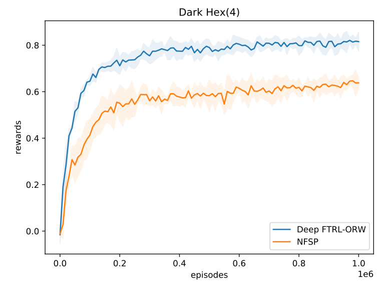


(c) Goofspiel(5)

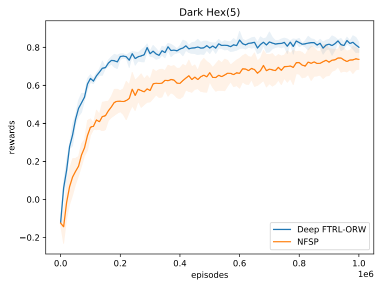
Figure 1: Convergence results of Deep FTRL-ORW, NFSP, QPG, RPG, OS-MCCFR, and OS-Deep CFR in Kuhn Poker, Leduc Poker, and Goofspiel(5). In all plots, the x-axis is the number of episodes of each algorithm, and the y-axis is shown on a log scale. The shaded area represents one standard deviation of the data over four random seeds.



(a) Phantom Tic Tac Toe



(b) Dark Hex(4)



(c) Dark Hex(5)

Figure 2: The results of Deep FTRL-ORW and NFSP against the random agent in Phantom Tic Tac Toe, Dark Hex(4) and Dark Hex(5). Dark Hex(4) represents that the board size of Dark Hex is 4. In all plots, the x-axis is the number of episodes of each algorithm, and the y-axis is the rewards. The shaded area represents one standard deviation of the data over four random seeds.

(Phantom TTT) and Dark Hex. These games are imperfect information versions of perfect-information games played on square boards. Due to the large size of these games, exploitability cannot be obtained. Therefore, we compare performance by playing against the random agent. We run 1000 episodes to obtain the average reward as the metric. In our experiments, we set the board size of Phantom TTT as 3×3, and set the board size of dark hex as 4 x 4 (dark hex(4)) and 5 x 5 (dark hex(5)). We only compare Deep FTRL-ORW with NFSP since OS-MCCFR cannot be scaled to these games due to the large game size, and OS-Deep CFR/QPG/RPG has poor performance even in small games. The hyperparameters of each algorithm are invariant across all tested games. We run each algorithm four times with different random seeds. The results are shown in Figure 2. In all tested games, we see that our method has the best performance. Moreover, the rewards curves of Deep FTRL-ORW are always higher than NFSP.

Conclusions

In this paper, we consider learning NE in two-player zero-sum IIEFGs with perfect recall. We propose a new FTRL algorithm called FTRL-ORW, which utilizes the Opponent Related Dilated DGF (ORD-DGF) to address the accumulated approximation error caused by the FD dilated DGF. To scale the FTRL-ORW algorithm to large-scale games, we introduce a new neural method called Deep FTRL-ORW. At each iteration, it employs Maximum Entropy Deep Reinforcement Learning (MEDRL) to compute the next iteration strategy of FTRL-ORW. Deep FTRL-ORW is a model-free method, which learns entirely from the sampled trajectories. We prove that FTRL-ORW and Deep FTRL-ORW converge to an $O(\sqrt{\log T/T^{2/3}})$-NE and $O(\sqrt{\log T/T^{2/3}} + E)$-NE, respectively. The experimental results show that FTRL-ORW achieves competitive performance compared with CFR, and Deep FTRL-ORW significantly outperforms existing model-free neural methods and OS-MCCFR.

Acknowledgments

This work is supported by Science and Technology Innovation 2030 New Generation Artificial Intelligence Major Project (No. 2018AAA0100905), the National Natural Science Foundation of China (No.: 62192783, 62206166), Primary Research and Development Plan of Jiangsu Province (No. BE2021028), Shenzhen Fundamental Research Program (No. 2021Szvup056), Shanghai Pujiang Program (No. 22PJ1403800), and in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

Bai, Y.; Jin, C.; Mei, S.; and Yu, T. 2022. Near-Optimal Learning of Extensive-Form Games with Imperfect Information. arXiv:2202.01752.

Brown, N.; Lerer, A.; Gross, S.; and Sandholm, T. 2019. Deep Counterfactual Regret Minimization. In *Proceedings of the 36th International Conference on Machine Learning*, 793–802.

Brown, N.; and Sandholm, T. 2018. Superhuman AI for Heads-Up No-Limit Poker: Libratus Beats Top Professionals. *Science*, 359(6374): 418–424.

Brown, N.; and Sandholm, T. 2019. Superhuman AI for Multiplayer Poker. *Science*, 365(6456): 885–890.

Chen, X.; Han, Z.; Zhang, H.; Xue, G.; Xiao, Y.; and Bennis, M. 2017. Wireless Resource Scheduling in Virtualized Radio Access Networks Using Stochastic Learning. *IEEE Transactions on Mobile Computing*, 17(4): 961–974.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12(7).

Farina, G.; Kroer, C.; and Sandholm, T. 2019. Optimistic Regret Minimization for Extensive-Form Games via Dilated Distance-Generating Functions. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 5221–5231.

Farina, G.; Kroer, C.; and Sandholm, T. 2020. Stochastic regret Minimization in Extensive-Form Games. In *Proceedings of the 37th International Conference on Machine Learning*, 3018–3028.

Farina, G.; Kroer, C.; and Sandholm, T. 2021. Better Regularization for Sequential Decision Spaces: Fast Convergence Rates for Nash, Correlated, and Team Equilibria. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, 432–432.

Farina, G.; and Sandholm, T. 2021. Model-Free Online Learning in Unknown Sequential Decision Making Problems and Games. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, volume 35, 5381–5390.

Farina, G.; Schmucker, R.; and Sandholm, T. 2021. Bandit Linear Optimization for Sequential Decision Making and Extensive-Form Games. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, volume 35, 5372–5380.

Fu, H.; Liu, W.; Wu, S.; Wang, Y.; Yang, T.; Li, K.; Xing, J.; Li, B.; Ma, B.; Fu, Q.; et al. 2021. Actor-Critic Policy Optimization in a Large-Scale Imperfect-Information Game. In *Proceedings of the 10th International Conference on Learning Representations*.

Geist, M.; Scherrer, B.; and Pietquin, O. 2019. A Theory of Regularized Markov Decision Processes. In *Proceedings of the 36th International Conference on Machine Learning*, 2160–2169.

Gruslys, A.; Lanctot, M.; Munos, R.; Timbers, F.; Schmid, M.; Perolat, J.; Morrill, D.; Zambaldi, V.; Lespiau, J.-B.; Schultz, J.; et al. 2020. The advantage regret-matching actor-critic. arXiv:2008.12234.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, 1861–1870.

Heinrich, J.; and Silver, D. 2016. Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. arXiv:1603.01121.

Hennes, D.; Morrill, D.; Omidshafiei, S.; Munos, R.; Perolat, J.; Lanctot, M.; Gruslys, A.; Lespiau, J.-B.; Parmas, P.; Duéñez-Guzmán, E.; et al. 2020. Neural Replicator Dynamics: Multiagent Learning via Hedging Policy Gradients. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 492–501.

Hoda, S.; Gilpin, A.; Pena, J.; and Sandholm, T. 2010. Smoothing Techniques for Computing Nash Equilibria of Sequential Games. *Mathematics of Operations Research*, 35(2): 494–512.

Kozuno, T.; Ménard, P.; Munos, R.; and Valko, M. 2021. Learning in Two-Player Zero-sum Partially Observable Markov Games with Perfect Recall. *In Proceedings of the 35th International Conference on Neural Information Processing*, 34: 11987–11998.

Kroer, C.; Waugh, K.; Kilinç-Karzan, F.; and Sandholm, T. 2015. Faster First-Order Methods for Extensive-Form Game Solving. In *Proceedings of the 16th ACM Conference on Economics and Computation*, 817–834.

Kroer, C.; Waugh, K.; Kılınç-Karzan, F.; and Sandholm, T. 2020. Faster Algorithms for Extensive-Form Game Solving via Improved Smoothing Functions. *Mathematical Programming*, 179(1): 385–417.

Lanctot, M.; Lockhart, E.; Lespiau, J.-B.; Zambaldi, V.; Upadhyay, S.; Pérolat, J.; Srinivasan, S.; Timbers, F.; Tuyls, K.; Omidshafiei, S.; et al. 2019. OpenSpiel: A Framework for Reinforcement Learning in Games. arXiv:1908.09453.

Lanctot, M.; Waugh, K.; Zinkevich, M.; and Bowling, M. 2009. Monte Carlo Sampling for Regret Minimization in Extensive Games. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, 1078–1086.

Lanctot, M.; Zambaldi, V.; Gruslys, A.; Lazaridou, A.; Tuyls, K.; Pérolat, J.; Silver, D.; and Graepel, T. 2017. A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 4193–4206.

Li, H.; Hu, K.; Zhang, S.; Qi, Y.; and Song, L. 2019. Double Neural Counterfactual Regret Minimization. In *In Proceedings of the 7th International Conference on Learning Representations*.

Lisỳ, V.; Davis, T.; and Bowling, M. 2016. Counterfactual Regret Minimization in Sequential Security Games. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 544–550.

Liu, W.; Jiang, H.; Li, B.; and Li, H. 2022. Equivalence Analysis between Counterfactual Regret Minimization and

Online Mirror Descent. In *In Proceedings of the 39th International Conference on Machine Learning*, 13717–13745.

McAleer, S.; Farina, G.; Lanctot, M.; and Sandholm, T. 2022. ESCHER: Eschewing Importance Sampling in Games by Computing a History Value Function to Estimate Regret. arXiv:2206.04122.

Sandholm, T. 2015. Steering Evolution Strategically: Computational Game Theory and Opponent Exploitation for Treatment Planning, Drug Design, and Synthetic Biology. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 4057–4061.

Shalev-Shwartz, S.; and Singer, Y. 2007. A Primal-Dual Perspective of Online Learning Algorithms. *Machine Learning*, 69(2): 115–142.

Shamma, J. S.; and Arslan, G. 2005. Dynamic Fictitious Play, Dynamic Gradient Play, and Distributed Convergence to Nash Equilibria. *IEEE Transactions on Automatic Control*, 50(3): 312–327.

Srinivasan, S.; Lanctot, M.; Zambaldi, V.; Pérolat, J.; Tuyls, K.; Munos, R.; and Bowling, M. 2018. Actor-Critic Policy Optimization in Partially Observable Multiagent Environments. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 3426–3439.

Steinberger, E. 2019. Single Deep Counterfactual Regret Minimization. arXiv:1901.07621.

Steinberger, E.; Lerer, A.; and Brown, N. 2020. Dream: Deep Regret Minimization with Advantage Baselines and Model-Free Learning. arXiv:2006.10410.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster Level in StarCraft II Using Multi-Agent Reinforcement Learning. *Nature*, 575(7782): 350–354.

Von Stengel, B. 1996. Efficient Computation of Behavior Strategies. *Games and Economic Behavior*, 14(2): 220–246.

Zinkevich, M. 2003. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *Proceedings of the 20th International Conference on Machine Learning*, 928–936.

Zinkevich, M.; Johanson, M.; Bowling, M.; and Piccione, C. 2007. Regret Minimization in Games with Incomplete Information. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, 1729–1736.