# Model-Free Neural Counterfactual Regret Minimization With Bootstrap Learning

Weiming Liu, Bin Li, *Member, IEEE*, and Julian Togelius, *Senior Member, IEEE*

***Abstract*—Counterfactual regret minimization (CFR) has achieved many fascinating results in solving large-scale imperfect information games (IIGs). Neural network approximation CFR (neural CFR) is one of the promising techniques that can reduce computation and memory consumption by generalizing decision information between similar states. Current neural CFR algorithms have to approximate cumulative regrets. However, efficient and accurate approximation in a large-scale IIG is still a tough challenge. In this article, a new CFR variant, recursive CFR (ReCFR), is proposed. In ReCFR, recursive substitute values (RSVs) are learned and used to replace cumulative regrets. It is proven that ReCFR can converge to a Nash equilibrium at a rate of $O(1/\sqrt{T})$. Based on ReCFR, a new model-free neural CFR with bootstrap learning, neural ReCFR-B, is proposed. Due to the recursive and noncumulative nature of RSVs, neural ReCFR-B has lower variance training targets than other neural CFRs. Experimental results show that neural ReCFR-B is competitive with the state-of-the-art neural CFR algorithms at a much lower training cost.**

***Index Terms*—Counterfactual regret minimization (CFR), game theory, imperfect information games (IIG), neural networks.**

## I. Introduction

IMPERFECT information game (IIG) is a kind of challenging game in which players can only obtain partial information. Conventional methods for perfect information games (PIGs) [1]–[4] generally do not play IIGs well [5]–[7]. Usually, IIGs are solved using equilibrium-finding algorithms [8]–[11]. In this article, we focus on two-player, zero-sum IIGs in extensive form, for example, heads-up no-limit Texas hold'em poker (HUNL). In recent years, counterfactual regret minimization (CFR) algorithm [8] has been applied in human-level poker agents [12]–[14]. CFR is an iterative algorithm. It has been proven that the average strategy will converge to a Nash equilibrium if the zero-thresholded regret $\max\{\max_a R_p^T(I,a),0\}$ at every state increases sublinearly [8], where $R_p^T(I,a)=\sum_{t=1}^{T} r^t(I,a)$ is the *cumulative regret*, and $r^t(I,a)$ is the *instantaneous regret*, and will be defined later. To minimize the cumulative regret at every state, a CFR algorithm needs to choose a strategy at every iteration according to the regrets. Conventional tabular CFR visits all states at every iteration and stores the cumulative regrets in a table. So it is computation expensive and memory intensive in the large-scale IIGs.

To deal with the large-scale IIGs, the technique "abstraction" [15] was proposed to cluster states and actions. However, abstraction techniques are domain specific and depend on expert knowledge. Function approximation CFR (FCFR) [16], [17] is a kind of algorithm that approximates cumulative regrets with function approximators. Recently, some FCFRs with neural network approximators (neural CFR) have been proposed [18]–[20]. Thanks to the potential generalization ability of function approximators, in an FCFR, it can be considered that the states are clustered in an implicit space. In other words, FCFR may abstract a game automatically, and, hopefully, reducing the computation and memory consumption.

Since it is intractable to traverse the full game tree and approximate the cumulative regrets directly in a large-scale IIG, FCFR usually trains the approximator on surrogate training targets. As proven in [17] and [18], the quality of the approximation directly impacts the quality of the output strategy. However, the training cost for approximating cumulative regrets could be high. Take a linear approximator with $p$ parameters as an example. Assume that the variance in training targets is $\sigma^2$ and the dataset size is $m$. It is known that the variance in approximated values on the fixed dataset is $\frac{p}{m}\sigma^2$ [21]. Unfortunately, the variance in training targets in a zero-sum IIG is likely to be high due to the adversarial nature. So, as a compromise, the dataset should be large. Also, the approximator should be trained for many epochs to prevent underfitting. Therefore, the cost for approximating cumulative regrets could be high. We will discuss more in Section III-D.

Furthermore, existing neural CFR algorithms usually require an exact simulator for model-based sampling (sampling multiple actions in a single state) in order to reduce the variance in training targets. To extend CFR to more IIGs with unknown models, it is necessary to develop model-free algorithms.

To reduce the training cost and use model-free sampling, it is desired to avoid approximating cumulative regrets. In this article, a new CFR algorithm, recursive CFR (ReCFR), is proposed, in which cumulative regrets are replaced by noncumulative

Manuscript received 13 July 2021; revised 18 December 2021; accepted 7 March 2022. Date of publication 11 March 2022; date of current version 15 September 2023. This work was supported by the National Natural Science Foundation of China under Grant U19B2044 and Grant 61836011. *(Corresponding author: Bin Li.)*

Weiming Liu is with the School of Data Science, University of Science and Technology of China, Hefei 230027, China (e-mail: weiming@mail.ustc.edu.cn).

Bin Li is with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: binli@ustc.edu.cn).

Julian Togelius is with the Department of Computer Science and Engineering, New York University, New York, NY 11201 USA (e-mail: julian@togelius.com).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TG.2022.3158649.

Digital Object Identifier 10.1109/TG.2022.3158649

2475-1502 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

recursive substitute values (RSVs) proposed in warm CFR [22]. We prove that ReCFR converges to a Nash equilibrium at a rate of $O(1/\sqrt{T})$. We also prove that vanilla CFR and full-width extensive-form fictitious play (XFP) [9] are special cases of ReCFR.

Based on ReCFR, a new neural CFR algorithm, neural ReCFR with bootstrapping (neural ReCFR-B), is proposed. In the algorithm, two RSV networks (one for each player) are trained to approximate RSVs. The training targets for RSVs can be estimated by bootstrap learning, which is an important benefit brought by the recursive nature of RSVs (cumulative regrets do not have this property). Since RSVs are noncumulative, the training targets should have lower variance than that for cumulative regrets. Also, bootstrap learning can reduce the variance, at the cost of being a bit biased [23]. So, due to the lower variance training targets, neural ReCFR-B may use model-free sampling and train the neural networks with fewer samples at every iteration than other neural CFR algorithms. Experimental results show that neural ReCFR-B is competitive with the state-of-the-art neural CFR algorithms [18]–[20] at a much lower training cost.

The contributions of this article are mainly in three aspects.

1) RSVs and warm CFR [22] are revisited, and a new exploitability bound is proven. Two new and intuitive lemmas (Lemmas 2 and 3) are also proposed for analyzing CFR algorithms.
2) A new CFR algorithm based on RSVs is presented, and it is proven to converge to a Nash equilibrium at a rate of $O(1/\sqrt{T})$. It is also shown that ReCFR can generalize both fictitious play [9] and vanilla CFR.
3) A new model-free neural CFR algorithm based on ReCFR is presented, and it shows the higher training efficiency than other algorithms on medium-size and large-size IIGs.

In the rest of this article, related work is first discussed in Section II. Then, notations and background are given in Section III. In Section IV, RSVs and warm CFR are revisited, and two new lemmas are given. In Sections V and VI, ReCFR and neural ReCFR-B are described in detail. Then, the experimental setup and the results of ReCFR and neural ReCFR-B are shown in Section VII. Finally, Section VIII concludes this article.

## II. Related Work

In recent years, many techniques have been proposed to improve the performance of CFR in large-scale IIGs [6], [12], [24], and [25]. Based on these techniques, the state-of-the-art algorithms for HUNL are Libratus [13] and Pluribus [14].

Regression CFR [16] is the first FCFR. Deep CFR [18] and double neural CFR (DNCFR) [19] are two more recent neural CFRs. Deep CFR uses two reservoir memory buffers to store all past instantaneous regrets and uses two regret networks (one for each player) to approximate the mean values of the past regrets. However, due to the high variance training targets, the variance in approximated cumulative regrets could be high. Another neural CFR, DNCFR, sums the current-iterate instantaneous regrets and the previous approximation as the training targets. However, the approximation error in each iteration will accumulate. So, as the number of iterations goes to infinity, the bias is unbounded. The problem may be alleviated using regret matching+ algorithm [26] and model-based robust sampling [19], but there is no theoretical guarantee. Single deep CFR [27] is a variant of deep CFR, which does not need to approximate the average strategy. Instead, it keeps all regret networks of past iterations and computes an average strategy at the end. However, cumulative regrets are still approximated.

Neural fictitious self-play (NFSP) [28] is a model-free algorithm based on full-width extensive-form fictitious play (XFP) [9]. NFSP uses DQN [3] to learn the best-response strategy for each player at each iteration and uses a strategy network to approximate the average best-response strategy. Since CFR has a better convergence guarantee than XFP, people may also be interested in model-free neural CFR algorithms. ARMAC [29] and DREAM [20] are both model-free neural CFR algorithms, such as neural ReCFR-B. However, both of them approximate cumulative regrets. Specifically, ARMAC approximates cumulative regrets by replaying past strategies and regenerating past regrets, whereas DREAM uses a reservoir buffer to store all past regrets. Besides, they also train an additional state-action network to reduce the variance in training targets [30]. A critical difference of neural ReCFR-B from these two methods is that it does not store or regenerate past regrets but learns RSVs only based on the average strategy.

Some other works have combined equilibrium-finding algorithms with gradient descending [7], [31]. They have achieved performance comparable with NFSP, and they seem to have the potential for solving large-scale IIGs. However, they have only been tested in small-scale games.

## III. Notations and Background

For a two-player zero-sum IIG, the set of players is denoted by $P = \{1, 2\}$. The chance player, denoted by $c$, is introduced to take actions for random events. A history $h$ is represented as a sequence of actions taken by all the players and the chance player. The root history is represented as an empty sequence, written as $\emptyset$. Denote the set of histories by $H$. For any nonterminal history $h \in H$, the set of legal actions at $h$ is denoted by $A(h)$. The set of all actions is denoted by $\mathcal{A}$. The acting player at history $h$ is denoted by $P(h)$, where $P(h) \in P \cup \{c\}$. In this article, we assume that $P(\emptyset) \in P$ and will present the main results under the viewpoint of $p = P(\emptyset)$.[1] After player $P(h)$ takes an action $a \in A(h)$, the resulted history is denoted by $h \cdot a$. If there exists a sequence of actions from $h$ to $h'$, then $h'$ is a descendent of $h$, denoted by $h \sqsubset h'$. Let $h \sqsubseteq h'$ represent that $h \sqsubset h'$ or $h = h'$. We only consider depth-limited games in this article. The set of terminal histories is denoted by $Z$. For any $z \in Z$, the *payoff* for player $p$ is $u_p(z) \in \mathbb{R}$. As the game is two-player and zero-sum, we have $u_1(z) = -u_2(z)$. Define $\text{Succ}_p(h)$ as the set of the earliest reachable histories such that for any $h' \in \text{Succ}_p(h)$, $P(h') = p$ or $h' \in Z$. "Earliest reachable" means that $h \sqsubseteq h'$ and there is no $h''$ such that $h'' \sqsubset h'$.

[1]For the other player $p' \neq P(\emptyset)$, we can construct an equivalent surrogate game with $P(\emptyset) = p'$ by adding a player $p'$ decision point at the beginning of the game with a single action [22].

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

In an IIG, the histories of each player $p \in P$ are divided into *information sets* (infosets). The set of infosets of player $p$ is denoted by $\mathcal{I}_p$. Let $\mathcal{I} = \mathcal{I}_1 \cup \mathcal{I}_2$. For any infoset $I \in \mathcal{I}_p$, all histories $h, h' \in I$ are indistinguishable to $p$, so $A(h) = A(h')$. Define $P(I) = P(h)$ and $A(I) = A(h)$ for any $h \in I$. Let $I(h)$ denote the infoset of history $h$, i.e., $I(h) = I$ for any $h \in I$. Besides, define $\text{Succ}_p(I, a)$ as the set of the earliest reachable infosets after action $a$ has been taken. Formally, $\text{Succ}_p(I, a) = \{I(h') | h' \notin Z, h' \in \text{Succ}_p(h \cdot a), h \in I\}$. Let $\text{Succ}_p(I) = \bigcup_{a \in A(I)} \text{Succ}_p(I, a)$. The range of payoffs reachable from $I$ is denoted by $\Delta(I)$. Formally, $\Delta(I) = \max_{z \in Z, h \in I: h \sqsubset z} u_{P(I)}(z) - \min_{z \in Z, h \in I: h \sqsubset z} u_{P(I)}(z)$. Let $\Delta = \max_{I \in \mathcal{I}} \Delta(I)$.

A strategy $\sigma_p$ of a player $p \in P$ is a function that maps any infoset $I \in \mathcal{I}_p$ to a probability vector over $A(I)$. The set of strategies of player $p$ is denoted by $\Sigma_p$. Given a strategy $\sigma_p \in \Sigma_p$, $\sigma_p(I)$ is the probability vector, and $\sigma_p(I, a)$ is the probability of choosing action $a$ at infoset $I$. Since the histories in an infoset are indistinguishable, the strategy in each of them must be identical. So $\sigma_p(h) = \sigma_p(I)$ and $\sigma_p(h, a) = \sigma_p(I, a)$ for any $h \in I$ and $a \in A(I)$. The strategy of the other player (the opponent) is denoted by $\sigma_{-p}$. Denote a strategy profile by $\sigma = \langle \sigma_p, \sigma_{-p} \rangle$. The chance player's strategy, which is fixed and known to all the players, is denoted by $\sigma_c(h, a)$.

$\pi^\sigma(h) = \prod_{h' a \sqsubseteq h} \sigma_{P(h')}(h', a)$ is called the *reach* of $h$, which is the probability of reaching $h$ when all the players act according to $\sigma$. $\pi^\sigma_p(h)$ is the contribution of $p$ to this probability. $\pi^\sigma_{-p}(h)$ is the contribution of the opponent and the chance player. The probability of reaching $h'$ from $h$ is denoted by $\pi^\sigma(h, h')$. In this article, we only consider *perfect recall* games. Therefore, we define $\pi^\sigma_p(I) = \pi^\sigma_p(h)$ for any $h \in I$. Accordingly, define the reach of the opponent and the chance player as $\pi^\sigma_{-p}(I) = \sum_{h \in I} \pi^\sigma_p(h)$. The *expected payoff* of the game for player $p$ is denoted by $u_p(\sigma_p, \sigma_{-p})$. Formally, $u_p(\sigma_p, \sigma_{-p}) = \sum_{z \in Z} \pi^\sigma(z) u_p(z)$. A *best response* $\text{BR}(\sigma_{-p})$ is a strategy of player $p$ such that $\text{BR}(\sigma_{-p}) = \text{argmax}_{\sigma'_p \in \Sigma_p} u_p(\sigma'_p, \sigma_{-p})$. A *Nash equilibrium* $\sigma^* = \langle \sigma^*_p, \sigma^*_{-p} \rangle$ is a strategy profile where every player plays a best response. The *exploitability* of a strategy $\sigma_p$ is the distance to a Nash equilibrium, defined as $e(\sigma_p) = u_p(\sigma^*_p, \text{BR}(\sigma^*_p)) - u_p(\sigma_p, \text{BR}(\sigma_p))$. Define the total exploitability as $e(\sigma) = \sum_{p \in P} e(\sigma_p)$.

### A. Counterfactual Regret Minimization (CFR)

CFR is an iterative algorithm for two-player zero-sum IIGs. It computes a strategy profile at every iteration using regret matching (RM) algorithm [8] according to the cumulative regrets of infosets. Then, the full game tree is traversed, and the cumulative regrets and average strategy are updated according to the strategy. It has been proven that the total exploitability of the average strategy is bounded by $O(1/\sqrt{T})$ after $T$ iterations of CFR [8] are played.

Let $\sigma^t$ be the strategy at iteration $t$. The *counterfactual value* of action $a$ at infoset $I$ is defined as

$$v_p^{\sigma^t}(I, a) = \sum_{h \in I} \sum_{z \in Z: h \sqsubset z} \pi^{\sigma^t}_{-p}(h) \pi^{\sigma^t}(h \cdot a, z) u_p(z) \quad (1)$$

where $v_p^{\sigma^t}(I) = \sum_{a \in A(I)} \sigma^t_p(I, a) v_p^{\sigma^t}(I, a)$ is the counterfactual value of infoset $I$. Counterfactual values can also be defined recursively: at each infoset $I$

$$v_p^{\sigma^t}(I, a) = \sum_{h \in I} \sum_{z \in Z: z \in \text{Succ}_p(h \cdot a)} \pi^{\sigma^t}_{-p}(z) u_p(z) + \sum_{I' \in \text{Succ}_p(I, a)} v_p^{\sigma^t}(I'). \quad (2)$$

This equation has been used in some existing literature [8], [22]. A proof is also provided in Appendix A.[2] Let $v_p^{\sigma^t} = v_p^{\sigma^t}(I(\emptyset))$, then, $u_p(\sigma^t_p, \sigma^t_{-p}) = v_p^{\sigma^t}$. In the rest of this article, we mainly use $v_p^{\sigma^t}$ and $v_p^{\langle \sigma^t_p, \sigma^t_{-p} \rangle}$ to denote the expected payoff. The *instantaneous regret* is defined as $r^t_p(I, a) = v_p^{\sigma^t}(I, a) - v_p^{\sigma^t}(I)$. The *cumulative counterfactual regret* (*cumulative regret*) of action $a$ at $I$ is

$$R^T_p(I, a) = \sum_{t=1}^{T} v_p^{\sigma^t}(I, a) - \sum_{t=1}^{T} v_p^{\sigma^t}(I). \quad (3)$$

The *average strategy* $\overline{\sigma}^T_p(I, a)$ is computed[3] according to the following:

$$\overline{\sigma}^T_p(I, a) = \frac{\sum_{t=1}^{T} \pi^{\sigma^t}_p(I) \sigma^t_p(I, a)}{\sum_{t=1}^{T} \pi^{\sigma^t}_p(I)}. \quad (4)$$

The *total regret* of player $p$ after $T$ iterations is

$$R^T_p = \max_{\sigma'_p \in \Sigma_p} \sum_{t=1}^{T} v_p^{\langle \sigma_p, \sigma'^t_{-p} \rangle} - \sum_{t=1}^{T} v_p^{\sigma^t}. \quad (5)$$

It has been proven in [8] that $R^T_p \le \sum_{I \in \mathcal{I}_p} \max_a (R^T_p(I, a))_+$, where $(\cdot)_+ = \max\{\cdot, 0\}$. Therefore, the total regret $R^T_p$ can be minimized by minimizing $\max_a (R^T_p(I, a))_+$ at every infoset, using, e.g., RM. At each infoset, RM computes the next iteration strategy according to

$$\sigma^{t+1}_p(I, a) = \frac{(R^t_p(I, a))_+}{\sum_{a' \in A(I)} (R^t_p(I, a'))_+}. \quad (6)$$

If $\sum_{a'} (R^t_p(I, a'))_+ = 0$, the action with the highest cumulative regret is assigned with probability 1 (w.p.1.) [18]. According to [22], we have the following lemma.

*Lemma 1 ([22]):* After T iterations of CFR are played, for any infoset $I$, $\sum_{a \in A(I)} (R^T_p(I, a))^2_+ \le \pi^{\overline{\sigma}^T}_{-p}(I) \Delta^2(I) |A(I)| T$.

Finally, it is well known that, in a two-player zero-sum game, the total exploitability equals the average total regret, i.e, $\epsilon(\overline{\sigma}^T) = \frac{1}{T} \sum_{p \in P} R^T_p$. This can be seen by the definitions. Therefore, $\epsilon(\overline{\sigma}^T) = O(1/\sqrt{T})$ in CFR [8].

### B. Recursive Substitute Values (RSVs) and Warm CFR

RSVs were first proposed in warm CFR [22]. Given an arbitrary strategy $\sigma$, warm CFR initializes the cumulative regrets

[2] Appendix: [Online]. Available: https://arxiv.org/abs/2012.01870

[3] We assume $\overline{\sigma}^t(I, a) > 0$ (and, thus, $\pi^{\overline{\sigma}^t}(I) > 0$) for any $I \in \mathcal{I}$ and $t \ge 1$. This is true if, e.g., $\sigma^1(I, a) = \frac{1}{|A(I)|} > 0$ for every $I \in \mathcal{I}$.

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

with substitute regrets, hoping the CFR algorithm can converge faster. The initialization is done by treating the strategy $\sigma$ as the average strategy after $T$ iterations of CFR are run. To some extent, the initialization is equivalent to recovering the cumulative regrets. Let $v_p'^{\sigma}(I,a)$ and $v_p'^{\sigma}(I)$ be the RSVs that recover the true average counterfactual values: $\frac{1}{T}\sum_{t=1}^{T} v_p^{\sigma^t}(I,a)$ and $\frac{1}{T}\sum_{t=1}^{T} v_p^{\sigma^t}(I)$. Then, the *substitute cumulative counterfactual regret* (*substitute regret*)

$$R_p'^{T}(I,a) = T\left(v_p'^{\sigma}(I,a) - v_p'^{\sigma}(I)\right) \tag{7}$$

should recover $R_p^T(I,a)$ at every infoset. According to the definition of counterfactual values and Lemma 1, the RSVs should satisfy the two constraints in the following if they are the true average counterfactual values:

$$v_p'^{\sigma}(I,a) = \sum_{h\in I}\sum_{z\in Z: z\in \text{Succ}_p(h\cdot a)} \pi_{-p}^{\sigma}(z)u_p(z) + \sum_{I'\in \text{Succ}_p(I,a)} v_p'^{\sigma}(I') \tag{8}$$

$$\sum_{a\in A(I)} \left(v_p'^{\sigma}(I,a) - v_p'^{\sigma}(I)\right)_+^2 \le \frac{\pi_{-p}^{\sigma}(I)\Delta^2(I)\,|A(I)|}{T}. \tag{9}$$

There could be many RSV profiles that fulfill the two constraints abovementioned. However, we can choose the RSVs for each infoset recursively: at infoset $I$, $v_p'^{\sigma}(I,a)$ is computed according to the immediate payoffs and the RSVs of its earliest reachable infosets, while $v_p'^{\sigma}(I)$ is chosen to fulfill (9).[4] Define the *substitute expected payoff* as follows:

$$v_p'^{\sigma} = v_p'^{\sigma}(I(\emptyset)). \tag{10}$$

In warm CFR, another $T'$ CFR iterations are run based on the substitute regrets. Specifically, define

$$R_p'^{T,T'}(I,a) = R_p'^{T}(I,a) + \sum_{t'=1}^{T'} r_p^{\sigma^{t'}}(I,a). \tag{11}$$

The strategy $\sigma_p^{t'}$ at iteration $t'$ is computed using RM according to $R_p'^{T,t'-1}(I,a)$. Define the average strategy as $\overline{\sigma}_p^{T,T'} = \frac{1}{T+T'}(T\sigma_p + T'\overline{\sigma}_p^{T'})$, where $\overline{\sigma}_p^{T'}$ is the average strategy of the $T'$ iterations. It is proven in [22] that

$$\sum_{a\in A(I)} \left(R_p'^{T,T'}(I,a)\right)_+^2 \le \pi_{-p}^{\overline{\sigma}^{T,T'}}(I)\Delta^2(I)|A(I)|(T+T'). \tag{12}$$

As a result, the average strategy will converge to a Nash equilibrium at a rate of $O(1/\sqrt{T+T'})$ under specific condition (Assumption 1, will be discussed in the next section).

### C. Monte Carlo CFR

Monte Carlo CFR (MCCFR) [32] is a variant of CFR that only traverses parts of the game tree at every iteration and estimates the instantaneous regrets by a Monte Carlo (MC) method. At iteration $t$, suppose a subset $Q$ of the full game tree is sampled, the sampled counterfactual value at $(I,a)$ is defined as

$$\tilde{v}_p^{\sigma^t}(I,a) = \sum_{z\in Q\cap Z} \frac{1}{q(z)}\pi_{-p}^{\sigma^t}(h)\pi^{\sigma^t}(h\cdot a, z)u_p(z) \tag{13}$$

where $q(z)$ is the probability of sampling $z$. Define $\tilde{v}_p^{\sigma^t}(I) = \sum_{a\in A(I)}\sigma_p^t(I,a)\tilde{v}_p^{\sigma^t}(I,a)$. The sampled instantaneous regret is $\tilde{r}_p^{\sigma^t}(I,a) = \tilde{v}_p^{\sigma^t}(I,a) - \tilde{v}_p^{\sigma^t}(I)$. The sampled cumulative regret is $\tilde{R}_p^T(I,a) = \sum_{t=1}^{T}\tilde{r}_p^{\sigma^t}(I,a)$, which is an unbiased estimator of the cumulative regret. There are many MCCFR algorithms [32]–[34], of which external sampling CFR and outcome sampling CFR (OSCFR) proposed in [32] are the most common.

### D. Function Approximation of Cumulative Regrets

For a two-player IIG, FCFR is usually combined with MCCFR and approximates the MC sampled cumulative regrets by training on surrogate training targets [18]–[20]. Denote the training target at $(I,a)$ by $\hat{R}^T(I,a)$ and assume $\hat{R}^T(I,a) = \tilde{R}^T(I,a) + \epsilon$, where $\tilde{R}^T(I,a)$ is the MC sampled cumulative regret and $\epsilon$ is a random variable at the infoset. The variance in training targets is defined as $\text{var}\,(\hat{R}^T) = \mathbb{E}_{I,a,\epsilon}(\hat{R}^T(I,a) - \mathbb{E}_\epsilon\hat{R}^T(I,a))^2$. The expected error is $\text{Err} = \mathbb{E}_{I,a,D}(f^T(I,a|D) - \tilde{R}^T(I,a))^2$. According to the famous bias-variance decomposition, $\text{Err} = (\text{bias}(f^T))^2 + \text{var}(f^T)$, where $(\text{bias}(f^T))^2$ and $\text{var}(f^T)$ are the bias and variance in approximated regrets, respectively. Specifically, $(\text{bias}(f^T))^2 = \mathbb{E}_{I,a}(\mathbb{E}_D f^T(I,a|D) - \tilde{R}^T(I,a))^2$ and $\text{var}(f^T) = \mathbb{E}_{I,a,D}(f^T(I,a|D) - \mathbb{E}_D f^T(I,a|D))^2$. The bias could be low if the training targets are unbiased estimators. However, as we mentioned before, the variance in approximated cumulative regrets is affected by the variance in training targets and dataset size.

As discussed in existing literature [17]–[19], there are at least two methods with different training targets for approximating sampled cumulative regrets. Note that a regret $\tilde{r}^{\sigma^t}(I,a)$, $t = 1,\ldots,T$ sampled according to MCCFR is an unbiased estimator of a kind of average regret at infoset $I$. So, the first method, which has been used in deep CFR and DREAM, is to train on a dataset $D = \{(I, \tilde{r}^{\sigma^t}(I,\cdot))|x^t(I) = 1, t = 1,\ldots,T\}$, where $x^t(I)$ is an indicator variable that is 1 if and only if $I$ is sampled at iteration $t$, and it is equal to 0, otherwise. Due to the adversarial nature of zero-sum IIGs, the sampled instantaneous regrets from different iterations are nonIID, and their variance tends to be high. To reduce the variance in training targets, model-based sampling is used in deep CFR, and a variance reduction technique [30] is used in DREAM. A large regret memory buffer is also used. Besides, considering that the one-to-many mapping $(I,a)\mapsto\tilde{r}^{\sigma^t}(I,a)$ is complex, they train the approximators for many epochs to prevent underfitting.

The second method, which was first proposed in [17] and has been used in DNCFR [19], is to estimate $\tilde{R}^T(I,a)$ at iteration $T$ with $f^{T-1}(I,a) + \tilde{r}^{\sigma^T}(I,a)$ and train the approximator on a dataset $D = \{(I, f^{T-1}(I,\cdot) + \tilde{r}^{\sigma^T}(I,\cdot))|I\in x^t(I)\}$. So, this is a method bootstrapping on the last approximation. The variance in training targets should be low since variance only comes from

[4]Because of the nonsmooth zero-thresholding operator, $v_p'^{\sigma}(I)$ cannot be presented in an explicit form. More details are provided in Appendix A.

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

the current-iterate instantaneous regrets. However, the targets are *biased* estimators of the sampled cumulative regrets, and the approximation error can *accumulate* over iterations. In other words, the bias in approximated cumulative regrets is unbounded as the number of iterations goes to infinity. As a compromise, DNCFR trains the approximator for many epochs at every iteration. In conclusion, the cost of approximating cumulative regrets is high.

## IV. Revising RSVs and Warm CFR

First, we would like to present a new and intuitive lemma for analyzing CFR algorithms.

*Lemma 2:* For any strategy profile $\sigma = \langle \sigma_p, \sigma_{-p} \rangle$, and another strategy $\sigma'_p$ of player $p$, we have $v_p^{\langle \sigma'_p, \sigma_{-p} \rangle} - v_p^{\sigma} = \sum_{I \in \mathcal{I}_p} \sum_{a \in A(I)} \pi_p^{\sigma'}(I)(v_p^{\sigma}(I,a) - v_p^{\sigma}(I))\sigma'_p(I,a)$.

The lemma utilizes the recursive definition of counterfactual values (2). Lemma 2 shows that the difference between expected payoffs can be completely represented by instantaneous regrets. As an application, the famous inequality $R_p^T \le \sum_{I \in \mathcal{I}_p} \max_a (R_p^T(I,a))_+$ [8] is immediately recovered because $\sum_{t=1}^T v_p^{\langle \sigma_p, {}'\sigma_{-p}^t \rangle} - \sum_{t=1}^T v_p^{\sigma^t} = \sum_{I \in \mathcal{I}_p} \sum_{a \in A(I)} \pi_p^{\sigma'}(I)(\sum_{t=1}^T r_p^t(I,a))\sigma'_p(I,a)$.

Since RSVs have (8) similar to (2), we have Lemma 3. Both lemmas are proven in Appendix A.

*Lemma 3:* For any strategy $\sigma = \langle \sigma_p, \sigma_{-p} \rangle$, another strategy $\sigma'_p$ of player $p$, and arbitrary $v_p'^{\sigma}(I)$ at all infosets, compute $v_p'^{\sigma}(I,a)$ according to (8), then $v^{\langle \sigma'_p, \sigma_{-p} \rangle} - v_p'^{\sigma} = \sum_{I \in \mathcal{I}_p} \sum_{a \in A(I)} \pi_p^{\sigma'}(I)(v_p'^{\sigma}(I,a) - v_p'^{\sigma}(I))\sigma'_p(I,a)$.

Note that Lemma 3 does not rely on how $v_p'^{\sigma}(I)$ is chosen at each infoset. Now, let us go back to the setting of warm CFR. Assume the initial strategy $\sigma$ is an average strategy $\overline{\sigma}^T$ generated by an arbitrary iterative algorithm, and another $T'$ iterations of CFR are run based on the substitute regrets. As a result of Lemmas 2 and 3, the total regret of the $T + T'$ iterations is

$$R_p^{T+T'} = \max_{\sigma'_p \in \Sigma_p} \sum_{t=1}^{T+T'} v_p^{\langle \sigma_p, {}'\sigma_{-p}^t \rangle} - \sum_{t=1}^{T+T'} v_p^{\sigma^t}$$

$$\le \underbrace{\left( T v_p'^{\sigma} - \sum_{t=1}^{T} v_p^{\sigma^t} \right)}_{\mathbf{Term}_1} + \underbrace{\sum_{I \in \mathcal{I}_p} \max_a \left( R_p'^{T,T'}(I,a) \right)_+}_{\mathbf{Term}_2}. \quad (14)$$

A proof is provided in Appendix A. In the equation, the total regret of the $T + T'$ iterations is *decomposed* to two terms related to the RSVs. Specifically, $\mathbf{Term}_1$ measures the difference between the substitute expected payoff and the true expected payoffs in the first $T$ iterations, whereas $\mathbf{Term}_2$ is the sum of the substitute regrets.

*Assumption 1 ([22]):* $v_1'^{\sigma} + v_2'^{\sigma} \le 0$.

As shown in [22], if Assumption 1 is true, the total regret will be bounded by the sum of the substitute regrets, i.e., $\sum_{p \in P} R_p^{T+T'} \le \sum_{p \in P} \sum_{I \in \mathcal{I}_p} \max_a (R_p'^{T,T'}(I,a))_+$. Note that $\sum_{t=1}^T v_p^{\sigma^t}$ in $\mathbf{Term}_1$ in (14) is canceled out as $v_1^{\sigma^t} + v_2^{\sigma^t} = 0$. Consequently, according to (12), we have $\epsilon(\overline{\sigma}^{T,T'}) = \frac{1}{T+T'} \sum_{p \in P} R_p^{T+T'} = O(1/\sqrt{T+T'})$. However, is Assumption 1 necessary for convergence? The answer may be no. Note that $\mathbf{Term}_1$ in (14) does not depend on $T'$, and it may be bounded by the total regret of the initial $T$ iterations. Based on these observations, we have the following theorem.

**Algorithm 1:** Recursive CFR.

```
Input: maximum iterations T.
for iteration t = 1 to T do
  for player p ∈ P do
    for infoset I ∈ I_p do
      σ_p^t(I,a) ← { RM(R'^{t-1}_p(I,a)),   t > 1,
                   { 1/|A(I)|,              t = 1.
      Update σ̄_p^t(I,a). ▷(4)
      Compute v'_p^{σ̄^t}(I,a) and v'_p^{σ̄^t}(I). ▷(8), (15)
      R'^t_p(I,a) ← t(v'_p^{σ̄^t}(I,a) − v'_p^{σ̄^t}(I)).
```

*Theorem 1:* Given an arbitrary strategy $\sigma$ and $T > 0$, choose the RSVs according to (8) and (9) and assume $v_p'^{\sigma}(I) \le \max_a v_p'^{\sigma}(I,a)$ at every infoset. If another $T'$ iterations of CFR are run based on the substitute regrets, then $\epsilon(\overline{\sigma}^{T,T'}) \le \frac{T\epsilon(\sigma)}{T+T'} + \frac{1}{\sqrt{T+T'}} \sum_{I \in \mathcal{I}_p} \sqrt{\pi_{-p}^{\overline{\sigma}^{T,T'}}(I)\Delta(I)\sqrt{|A(I)|}}$.

The proof is given in Appendix A. As we can see, when $T'$ is sufficiently large (e.g., $T' \ge (T\epsilon(\sigma))^2 - T$), the first term $T\epsilon(\sigma)/(T+T') = O(1/\sqrt{T+T'})$, and thus $\epsilon(\overline{\sigma}^{T,T'}) = O(1/\sqrt{T+T'})$. For example, if the initial strategy is an average strategy of a CFR algorithm, we have $\epsilon(\sigma) = O(1/\sqrt{T})$ and $e(\sigma^{T,T'}) = O(\sqrt{T}/(T+T') + 1/\sqrt{T+T'}) = O(1/\sqrt{T+T'})$ for any $T' > 0$. Note that Theorem 1 does not rely on Assumption 1.

In this section, we relax Assumption 1 in warm CFR to a trivial constraint: $v_p'^{\sigma}(I) \le \max_a v_p'^{\sigma}(I,a)$ for all infosets. Following the idea, we can prove the convergence of ReCFR given in the next section.

## V. New CFR Algorithm

In this section, a new CFR algorithm, named ReCFR, is presented. ReCFR is based on RSVs proposed in [22]. Instead of only computing the RSVs at the beginning for warm starting, ReCFR discards the cumulative regrets and replaces them with the substitute regrets at every iteration. Therefore, the cumulative regrets are never tracked. The pseudocode of ReCFR is given in Algorithm 1.

Specifically, at iteration $t$, the RSV $v_p'^{\overline{\sigma}^t}(I,a)$ is computed according to (8). However, we choose $v_p'^{\overline{\sigma}^t}(I)$ at each infoset according to the constraint

$$\sum_{a \in A(I)} \left( t v_p'^{\overline{\sigma}^t}(I,a) - t v_p'^{\overline{\sigma}^t}(I) \right)_+^2 = \lambda_p^t(I) \quad (15)$$

where $\lambda_p^t(I) \in [0, \infty)$ is a parameter, will be set later. Note that when $\lambda_p^t(I) > 0$, the $v_p'^{\overline{\sigma}^t}(I)$ fulfilling the constraint exists and is unique. Moreover, $v_p'^{\overline{\sigma}^t}(I) < \max_a v_p'^{\overline{\sigma}^t}(I,a)$. When $\lambda_p^t(I) = 0$, we force that $v_p'^{\overline{\sigma}^t}(I) = \max_a v_p'^{\overline{\sigma}^t}(I,a)$. The substitute regret

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

$R_p'^t(I,a)$ is

$$R_p'^t(I,a) = t(v_p'^{\overline{\sigma}^t}(I,a) - v_p'^{\overline{\sigma}^t}(I)). \tag{16}$$

The strategy at iteration $t+1$ is computed using RM

$$\sigma_p^{t+1}(I,a) = \frac{(R_p'^t(I,a))_+}{\sum_{a'\in A(I)}(R_p'^t(I,a'))_+}. \tag{17}$$

When $\sum_{a'\in A(I)}(R_p'^t(I,a'))_+ = 0$, the action with the maximal $R_p'^t(I,a)$ is assigned w.p.1. We set $\sigma_p^1(I,a) = 1/|A(I)|$. The average strategy is computed according to (4). Since solving (15) requires a linear search, the cost of ReCFR at every iteration is $O(|\mathcal{I}||\mathcal{A}|^2)$, which is worse than vanilla CFR ($O(|\mathcal{I}||\mathcal{A}|)$).

Note that when $\lambda_p^t(I) = 0$ at all infosets, we have $v_p^{\langle\sigma_p^{t+1},\overline{\sigma}_{-p}^t\rangle} = \max_{\sigma_p'} v_p^{\langle\sigma_p',\overline{\sigma}_{-p}^t\rangle}$. In other words, $\sigma_p^{t+1}$ is a best response to $\overline{\sigma}_{-p}^t$. According to the definition of XFP [9], we have proposition 1. Similarly, when $\lambda_p^t(I) = \sum_a (R^t(I,a))_+^2$ at each infoset, vanilla CFR is recovered. The proofs for the propositions are given in Appendix B.

*Proposition 1:* ReCFR is equivalent to XFP if $\lambda_p^t(I) = 0$ at every infoset.

*Proposition 2:* ReCFR is equivalent to vanilla CFR if $\lambda_p^t(I) = \sum_a (R^t(I,a))_+^2$ at each infoset.

### A. Properties of ReCFR

According to (14), the total regret of ReCFR at iteration $T$ can also be decomposed as

$$R_p^T \le \underbrace{\left(Tv_p'^{\overline{\sigma}^T} - \sum_{t=1}^T v_p^{\sigma^t}\right)}_{\mathbf{Term}_1} + \underbrace{\sum_{I\in\mathcal{I}_p}\max_a (R_p'^T(I,a))_+}_{\mathbf{Term}_2}. \tag{18}$$

Therefore, if Assumption 1 is true at iteration $T$, $\mathbf{Term}_1$ can be canceled out when summing over $R_p^T$ of the players, and thus $\epsilon(\overline{\sigma}^T) \le \frac{1}{T}\sum_{p\in P}\sum_{I\in\mathcal{I}_p}\sqrt{\lambda_p^T(I)}$ (remember that $\epsilon(\overline{\sigma}^T) = \frac{1}{T}\sum_{p\in P} R_p^T$). However, it is *nontrivial* to ensure both Assumption 1 and $\lambda_p^T(I) = O(T)$ *as* $T\to\infty$.[5] In other words, we may not be able to guarantee that $\epsilon(\overline{\sigma}^T) = O(1/\sqrt{T})$. Fortunately, similar to Theorem 1, Assumption 1 is unnecessary as long as both terms in (18) are bounded by $O(\sqrt{T})$. Specially, $\mathbf{Term}_1$ equals

$$\sum_{t=1}^T \left(tv_p'^{\overline{\sigma}^t} - (t-1)v_p'^{\overline{\sigma}^{t-1}} - v_p^{\sigma^t}\right). \tag{19}$$

Thanks to Lemmas 2 and 3, for $t \ge 1$, we have

$$\begin{aligned} &tv_p'^{\overline{\sigma}^t} - (t-1)v_p'^{\overline{\sigma}^{t-1}} - v_p^{\sigma^t} \\ &= \sum_{I\in\mathcal{I}_p}\sum_{a\in A(I)} \pi_p^{\sigma^{t+1}}(I) g_p'^t(I,a)\sigma_p^{t+1}(I,a) \end{aligned} \tag{20}$$

where $g_p'^t(I,a) = (R_p'^{t-1}(I,a) + r_p^{\sigma^t}(I,a)) - R_p'^t(I,a)$. We set $R_p'^0(I,a) = 0$ and $v_p'^{\overline{\sigma}^0} = 0$ as they are irrelevant to the algorithm. Here, both $R_p'^{t-1}(I,a) + r_p^{\sigma^t}(I,a)$ and $R_p'^t(I,a)$ are the substitute regrets of infoset $I$ at iteration $t$, according to (11). So it is potential that the difference is negligible. Based on the abovementioned analysis, Theorem 2 is obtained. The proof is given in Appendix B, where (20) is also proven.

*Theorem 2:* After $T$ iterations of ReCFR are played, for each player $p\in P$, if $\lambda_p^t(I) > 0$ at every infoset, then, $R_p^T \le \sum_{t=1}^T\sum_{I\in\mathcal{I}_p}\frac{1}{2\sqrt{\lambda_p^t(I)}}(\lambda_p^{t-1}(I) - \lambda_p^t(I) \quad + \sum_a (r_p^{\sigma^t}(I,a))^2)_+ + \sum_{I\in\mathcal{I}_p}\sqrt{\lambda_p^T(I)}$.

Theorem 2 shows that the two terms in (18) are bounded by the two terms in the theorem, respectively. When $\lambda_p^{t-1}(I) - \lambda_p^t(I) + \sum_a (r_p^{\sigma^t}(I,a))^2 \le 0$, $\mathbf{Term}_1$ in (18) will be bounded by zero, and thus $\epsilon(\overline{\sigma}^T) \le \frac{1}{T}\sum_{p\in P}\sum_{I\in\mathcal{I}_p}\sqrt{\lambda_p^T(I)}$, as shown in Corollary 1.

*Corollary 1:* If $\lambda_p^t(I) = \pi_{-p}^{\overline{\sigma}^t}(I)\Delta^2(I)|A(I)|t$ at each infoset, then $\epsilon(\overline{\sigma}^T) \le \frac{1}{\sqrt{T}}\sum_{p\in P}\sum_{I\in\mathcal{I}_p}\sqrt{\pi_{-p}^{\overline{\sigma}^T}(I)}\Delta(I)\sqrt{|A(I)|}$.

This corollary is based on an inequality [22]

$$\sum_a (r_p^{\sigma^t}(I,a))^2 \le \pi_{-p}^{\sigma^t}(I)\Delta^2(I)|A(I)|. \tag{21}$$

Note that $\pi_{-p}^{\overline{\sigma}^t}(I)t = \sum_{k=1}^t \pi_{-p}^{\sigma^k}(I)$. As we can see, $\epsilon(\overline{\sigma}^T) = O(1/\sqrt{T})$. However, setting $\lambda_p^t(I)$ according to the corollary would perform poorly because the inequality in (21) could be loose. In other words, $\lambda_p^t(I)$ could be further reduced. Actually, we can choose $\lambda_p^t(I)$ in a broad range. According to the Theorem 2, the total regret is bounded by $O(\sqrt{T})$ as long as $\lambda_p^{t-1}(I) - \lambda_p^t(I)$ is upper bounded and $\lambda_p^t(I) = \Theta(t)$.

*Corollary 2:* If $\lambda_p^t(I) = \lambda\pi_{-p}^{\overline{\sigma}^t}(I)\Delta^2(I)|A(I)|t$, then $\epsilon(\overline{\sigma}^T) \le (\frac{1}{\sqrt{\lambda}} + \sqrt{\lambda})\frac{1}{\sqrt{T}}\sum_{p\in P}\sum_{I\in\mathcal{I}_p}\sqrt{\pi_{-p}^{\overline{\sigma}^T}(I)}\Delta(I)\sqrt{|A(I)|}$.

The proofs for both corollaries are given in Appendix B. In Corollary 2, $\lambda\in(0,\infty)$ is a hyperparameter. As we can see, the optimal exploitability bound is achieved when $\lambda = 1$. Although Corollary 2 provides a bound worse than that in Corollary 1, it allows $\lambda$ to be chosen in $(0,\infty)$.

Note that Theorem 2 does not apply when $\lambda_p^t(I) = 0$ at some infosets. Therefore, the new theoretical results do not apply to CFR and XFP. Actually, when ReCFR is equivalent to CFR, $\mathbf{Term}_1$ in (18) equals zero, and the known inequality $R_p^T \le \sum_{I\in\mathcal{I}_p}\max_a(R_p^T(I,a))_+$ is recovered. When ReCFR is equivalent to XFP, $\mathbf{Term}_2$ equals zero, and $\mathbf{Term}_1$ is exactly the definition of $R_p^T$ given in (5). So, (18) does not provide new information for CFR or XFP. However, we can consider ReCFR a method that generalizes CFR and XFP.

### B. Adapting the Hyperparameter

Corollary 2 suggests that the optimal $\lambda$ should be 1. However, empirical results in Fig. 1 show that the hyperparameter could be much smaller. The reason may be that the bound for $\mathbf{Term}_1$ in (18) is too loose. So, a smaller $\lambda$ is needed to balance the two terms. Although Assumption 1 is not required in ReCFR,

[5]Corollary 1 in [22] does not apply. It assumes $T$ iterations of CFR were played (so Lemma 1 holds) when computing the RSVs, which is not true in ReCFR.

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

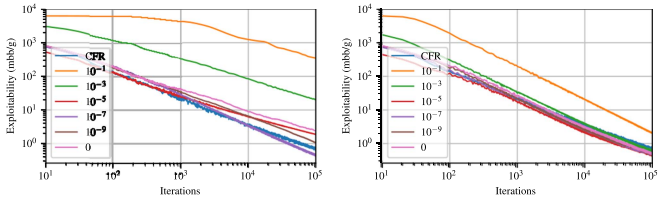


Fig. 1. Exploitability curves of vanilla CFR and ReCFR on Leduc poker. Left-hand side: ReCFR with constant λs shown in the legend. Right-hand side: ReCFR with adaptive λs initialized with the values in the legend.

satisfying it may make the algorithm behave like a CFR. In this article, we propose to use a simple adaptive algorithm to maintain that $v_1'^{\overline{\sigma}^t} + v_2'^{\overline{\sigma}^t} = 0$ loosely. Note that increasing $\lambda_p^t(I)$ at any infoset could reduce $v_p'^{\overline{\sigma}^t}$. So, we check $v_1'^{\overline{\sigma}^t} + v_2'^{\overline{\sigma}^t}$ at every iteration. If it is greater than 0, we increase λ: $\lambda = \beta_{\text{amp}}\lambda$ with $\beta_{\text{amp}} > 1$; otherwise, λ is reduced: $\lambda = \beta_{\text{damp}}\lambda$ with $\beta_{\text{damp}} < 1$.

## VI. New Neural Network Approximation CFR

This section describes neural ReCFR with bootstrapping (neural ReCFR-B) in detail. At each iteration, Neural ReCFR-B approximates the RSVs instead of the cumulative regrets. The RSVs are computed according to (8) and (15). According to (8), $v_p'^{\overline{\sigma}^t}(I,a)$ is scaled by the reach of the opponent and the chance player. Dividing it by $\pi_{-p}^{\overline{\sigma}^t}(I)$ will unify the ranges of the RSVs, which is helpful for neural approximation. Let $u_p'^{\overline{\sigma}^t}(I,a) = v_p'^{\overline{\sigma}^t}(I,a)/\pi_{-p}^{\overline{\sigma}^t}(I)$ and $u_p'^{\overline{\sigma}^t}(I) = v_p'^{\overline{\sigma}^t}(I)/\pi_{-p}^{\overline{\sigma}^t}(I)$. In this article, we choose to approximate $u_p'^{\overline{\sigma}^t}(I,a)$ instead of $v_p'^{\overline{\sigma}^t}(I,a)$. Put $u_p'^{\overline{\sigma}^t}(I,a)$ into (15), then $u_p'^{\overline{\sigma}^t}(I)$ is required to fulfill

$$\sum_{a\in A(I)} \left(u_p'^{\overline{\sigma}^t}(I,a) - u_p'^{\overline{\sigma}^t}(I)\right)_+^2 = \beta_p^t(I) \tag{22}$$

where $\beta_p^t(I) = \lambda_p^t(I)/(\pi_{-p}^{\overline{\sigma}^t}(I)t)^2$. According to Corollary 2, we set $\beta_p^t(I) = \lambda\Delta^2(I)|A(I)|/(\pi_{-p}^{\overline{\sigma}^t}(I)t)$. Put $u_p'^{\overline{\sigma}^t}(I,a)$ into (8), we get

$$u_p'^{\overline{\sigma}^t}(I,a) = \mathbb{E}_{h\sim I, h'\sim \text{Succ}_p^{\overline{\sigma}^t}(h\cdot a)} \left\{\mathbb{1}_{h'\in Z} u_p(h') + \mathbb{1}_{h'\notin Z} u_p'^{\overline{\sigma}^t}(I(h'))\right\}. \tag{23}$$

A proof for the equation is provided in Appendix C. Equation (23) implies that if we *sample* the payoffs and the RSVs of the earliest reachable histories starting from $(I,a)$, then the expectation of the sampled values is precisely the RSV of action $a$ at infoset $I$. According to this equation, a bootstrap method similar to Q-learning [35] is derived.

### A. Bootstrap Learning for RSVs

According to (23), we propose ReCFR with bootstrapping (ReCFR-B).

*Definition 1:* ReCFR-B is an ReCFR that learns the RSVs using a bootstrap method: at each iteration $t$ of ReCFR-B, initialize $u_p'^{\overline{\sigma}^t,1}(I,a)$ with an arbitrary value for each $p\in P$, $I\in\mathcal{I}_p$, and $a\in A(I)$. Let each $p\in P$ play $K$ games with the opponent who uses strategy $\overline{\sigma}_{-p}^t$. During the play, for each transition $(h,a,h')$ encountered in game $1\le k\le K$, update $u_p'^{\overline{\sigma}^t,k}(I(h),a)$ according to

$$u_p'^{\overline{\sigma}^t,k+1}(I(h),a) = u_p'^{\overline{\sigma}^t,k}(I(h),a) + \alpha_k \left(\mathbb{1}_{h'\in Z} u_p(h') + \mathbb{1}_{h'\notin Z}\gamma u_p'^{\overline{\sigma}^t,k}(I(h')) - u_p'^{\overline{\sigma}^t,k}(I(h),a)\right)$$

where $u_p'^{\overline{\sigma}^t,k}(I)$ fulfills

$$\sum_{a\in A(I)} \left(u_p'^{\overline{\sigma}^t,k}(I,a) - u_p'^{\overline{\sigma}^t,k}(I)\right)_+^2 = \beta_p^t(I).$$

*Theorem 3:* At each iteration $t$ of ReCFR-B, if every terminal history is visited with a nonzero probability in each game, $0<\gamma\le 1$, $\sum_k \alpha_k = \infty$, and $\sum_k \alpha_k^2 < \infty$, then $u_p'^{\overline{\sigma}^t,K}(I,a)$ converges to $u_p'^{\overline{\sigma}^t}(I,a)$ w.p.1. for every $(I,a)$ as $K\to\infty$.

The proof is provided in Appendix C. In practice, we set $\alpha_k$ to a constant learning rate and set $\gamma$ to 1 as NFSP did [28]. According to the definition, the update of $u_p'^{\overline{\sigma}^t,k}(I,a)$ does not depend on how infoset $I$ is reached and how the player $p$ acts in descendants. So the learning is off-policy. ReCFR-B is similar to (batch-)OSCFR [32], as both of them update the values of infosets by sampling trajectories. OSCFR is usually considered model-free [20], [29].[6] So, if $\beta_p^t(I)$ for each infoset is chosen without referring to any private information, e.g., $\sigma_c(h,a)$ and $\Delta(I)$, about the transition model, we can consider ReCFR-B model-free. In practice, we use $\beta_p^t(I) = \lambda\Delta^2(I)|A(I)|/(\pi_{-p}^{\overline{\sigma}^t}(I)t)$ as it gives a good exploitability bound.

### B. Neural ReCFR With Bootstrapping

For each player, an RSV network $\mathcal{R}(\theta_p^t)$ with parameters $\theta_p^t$ is used to approximate $u_p'^{\overline{\sigma}^t}(I,a)$ at all infosets. Also, a neural network $\Pi(\phi_p^t)$ with parameters $\phi_p^t$ is used to approximate the average strategy $\overline{\sigma}_p^t$. At iteration $t$, the next iteration strategy $\sigma_p^{t+1}$ is computed according to the output of the RSV network

$$\sigma_p^{t+1}(I,a) = \text{RM}\left(\mathcal{R}\left(I,a|\theta_p^t\right) - u_p'^{\overline{\sigma}^t}(I)\right) \tag{24}$$

where $u_p'^{\overline{\sigma}^t}(I)$ is the RSV of infoset $I$, and it is chosen to fulfill (22) with $u_p'^{\overline{\sigma}^t}(I,a) \leftarrow \mathcal{R}(I,a|\theta_p^t)$. Since RM is *scale-invariant*, this equation is equivalent to (17) if the approximation is accurate. We use anticipatory dynamics [28] to estimate the average strategy $\overline{\sigma}^t$. Specifically, the strategy for each player at iteration $t$ is $\hat{\sigma}_p = (1-\eta)\Pi(\phi_p^{t-1}) + \eta\sigma_p^t$, where $\eta$ is a hyperparameter. We set $\eta = 0.1$ as NFSP did. In practice, multiple trajectories are sampled. So, $\Pi(\phi_p^{t-1})$ is selected with probability $1-\eta$ for playing, while $\sigma_p^t$ is selected with probability $\eta$. The pseudocode of Neural ReCFR-B is given in Algorithm 2. At each iteration and for each player $p$, $K$ games are played using the anticipatory strategy $\hat{\sigma}$. During the play, the transitions are collected into a temporary dataset $\mathcal{D}_p^{\mathcal{R}}$, and the behavior tuples are stored in a reservoir buffer [28] $\mathcal{M}_p^{\Pi}$. The RSV network for player $p$ is

[6] The players are regarded as a whole. However, the chance player's strategy is a part of the transition model.

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

**Algorithm 2:** Neural ReCFR With Bootstrapping.

Input: maximum iterations $T$.
Initialize RSV network parameters $\theta_p^0$ and policy network parameters $\phi_p^0$ for each player.
Initialize strategy memory buffer $\mathcal{M}_p^{\Pi}$ for each player.
**for** iteration $t = 1$ to $T$ **do**
  **for** player $p \in P$ **do**
    Initialize a temporary dataset $\mathcal{D}_p^{\mathcal{R}}$.
    **for** game $k = 1$ to $K$ **do**
      Sample a strategy $\hat{\sigma}_i$ for each player $i \in P$, $\hat{\sigma}_i = \begin{cases} \sigma_i^t, & \text{with probability } \eta, \\ \Pi(\phi_i^{t-1}), & \text{with probability } 1-\eta. \end{cases}$
      $\mathcal{T}_p^k \leftarrow \text{Play}(\hat{\sigma}_p, \hat{\sigma}_{-p})$. ▷Sample a trajectory $\mathcal{T}_p^k$ using $\hat{\sigma}$.
      Collect all transitions $(h, a, h,' \pi_{-p}^{\overline{\sigma}^t}(I(h))) \in \mathcal{T}_p^k$ to dataset $\mathcal{D}_p^{\mathcal{R}}$.
      **if** player $p$ follows the current strategy $\sigma^t$ **then**
        Store all behavior tuples $(h, a) \in \mathcal{T}_p^k$ in strategy memory buffer $\mathcal{M}_p^{\Pi}$ .
    Train $\theta_p^t$ on loss (25) and train $\phi_p^t$ on loss (26).

trained by minimizing

$$\mathcal{L}(\theta_p^t) = \mathbb{E}_{\left(h,a,h,'\pi_{-p}^{\overline{\sigma}^t}(I(h))\right)\sim\mathcal{D}_p^{\mathcal{R}}} \Bigg\{ \left(\mathbb{1}_{h'\in Z} u_p(h') + \mathbb{1}_{h'\notin Z} u_p^{\prime\overline{\sigma}^t}(I(h')) - \mathcal{R}(I(h), a|\theta_p^t)\right)^2 \Bigg\} \tag{25}$$

where $\theta_p^t$ is the trainable parameters initialized with $\theta_p^{t-1}$, and $u_p^{\prime\overline{\sigma}^t}(I(h'))$ is chosen to fulfill (22) with $u_p^{\prime\overline{\sigma}^t}(I(h'), a) \leftarrow \mathcal{R}(I(h'), a|\theta_p^t)$. The average network for player $p$ is trained by minimizing the cross-entropy loss

$$\mathcal{L}(\phi_p^t) = \mathbb{E}_{(h,a)\sim\mathcal{M}_p^{\Pi}} \left\{ -\log \Pi(I(h), a|\phi_p^t) \right\} \tag{26}$$

where $\phi_p^t$ is the trainable parameters initialized with $\phi_p^{t-1}$.

The algorithm requires each player $p \in P$ plays with an opponent who uses the average strategy $\overline{\sigma}_{-p}^t$. However, the strategy for player $p$ is not specified. We propose asymmetric learning, in which player $p$ uses a mixed strategy of the uniform random strategy and strategy $\sigma_p^t$, i.e., $\hat{\sigma}_p = (1-\eta)\,\text{Uniform} + \eta\sigma_p^t$. Accordingly, the method that both players use the anticipatory strategy is called symmetric learning. We use asymmetric learning in our experiments by default.

### C. Discussion

*Approximating the average strategy:* In neural ReCFR-B, approximating cumulative regrets is avoided, but the average strategy is still approximated. Single deep CFR [27] proposed remembering all past regret networks to avoid doing that. This method should also be compatible with Neural ReCFR-B. Nevertheless, it should be easier to train the average networks since the average strategy changes more slowly than the cumulative regrets.

*Relationship with DNCFR:* Both neural ReCFR-B and DNCFR use bootstrap learning to reduce the variance in training targets. However, a critical difference is that Neural ReCFR-B bootstraps on the RSVs of the earliest reachable infosets, while DNCFR bootstraps on the last approximations. Since we only consider depth-limited games, the approximation error in neural ReCFR-B is *bounded* at every iteration, while the error in DNCFR is *accumulated*.

*Relationship with NFSP:* Since ReCFR is equivalent to XFP according to Proposition 1 when $\lambda_p^t(I) = 0$ at all infosets, neural ReCFR-B can also be regarded as NFSP in this case. Note that the learning algorithm, i.e., DQN [3], in NFSP has two essential components: 1) a value memory buffer, which may improve sample efficiency, and 2) a target network, which may stabilize the training. We will test Neural ReCFR-B with these two components in Section VII.

## VII. Experimental Setup and Results

We first test ReCFR on Leduc poker [36] to show the convergence properties of the algorithm. Leduc poker is a small size game with two rounds of betting. Then, neural ReCFR-B is tested on heads-up flop hold'em poker (FHP) [18] and heads-up limit Texas hold'em (HULH).[7] FHP is a medium-size game with over $10^{12}$ nodes and $10^9$ infosets, whereas HULH is a large-size game with over $10^{17}$ nodes and $10^{14}$ infosets. More details about the games are given in Appendix D. The neural network architecture is the same as in [18]. It is a seven-layer fully connected neural network.

We train the neural networks using Adam optimizer [37], with a batch size of 128, a learning rate of 0.001, and gradient norm clipping to 1. At every iteration, 1000 plays are performed to collect samples. The RSV networks are trained for two epochs (approximately 32 SGD steps with a batch size of 128 on FHP), while the average networks are trained for 16 SGD steps. The strategy memory size is set to 10 million. For HULH, we increase the batch size to 6400, the number of SGD steps to 64, and the number of plays to 100 000. The strategy memory size is also increased to 40 million. We also test Neural ReCFR-B with two RSV memory buffers (one for each player). In this setting, the RSV memory sizes for FHP and HULH are 1 million and 4 million, respectively, and the RSV networks are trained for 32 and 64 SGD steps, respectively.

[7]Source code: [Online]. Available: https://github.com/Liuweiming/Neural_ReCFR_B

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

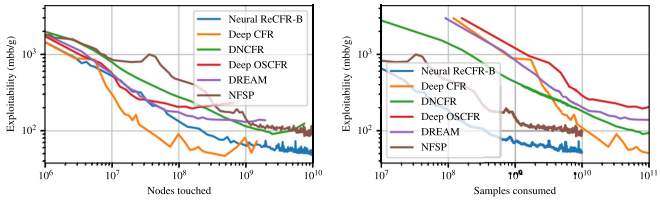


Fig. 2. Exploitability curves of different algorithms on FHP. The $x$-axes represent the number of nodes touched and the number of samples consumed (= #SGD steps × batch size), respectively.

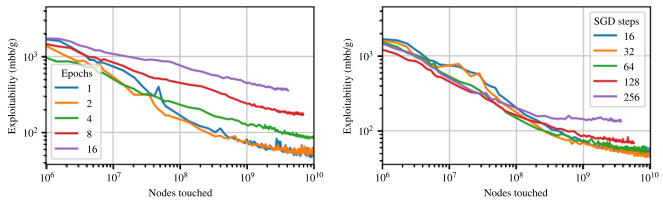


Fig. 3. Exploitability curves of neural ReCFR-B with different numbers of epochs/SGD steps on FHP. Left-hand side: the default neural ReCFR-B. Right-hand side: neural ReCFR-B with RSV memory buffers.

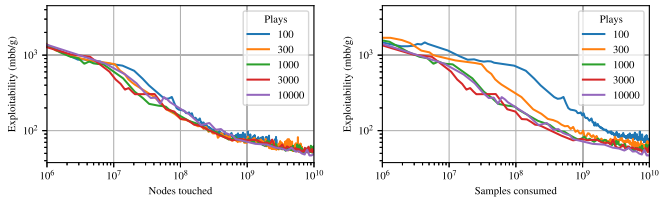


Fig. 4. Exploitability curves of neural ReCFR-B with different numbers of plays on FHP. The $x$-axes represent the number of nodes touched and the number of samples consumed, respectively.

Neural ReCFR-B is compared with two model-based neural CFRs: deep CFR and DNCFR, and two model-free algorithms: DREAM and NFSP, on FHP. A model-free variant of deep CFR with outcome sampling [32] (deep OSCFR) is also included. All the algorithms are implemented based on OpenSpiel [38]. We implement deep CFR and DREAM with the hyperparameters given in [18] and [20], respectively. The hyperparameters for DNCFR and NFSP are determined through a set of experiments. A head-to-head comparison is performed between neural ReCFR-B and deep CFR on HULH. All the settings are given in Appendix E.

Performance is measured using exploitability in terms of milli big blinds per game (mbb/g). For a specific exploitability value, the numbers of nodes touched and the numbers of samples consumed (= #SGD steps × batch size) of different algorithms are compared, respectively. Algorithms touching fewer nodes are more sample efficient, whereas algorithms consuming fewer samples are more training efficient.

### A. Experimental Results of ReCFR

In this section, we test ReCFR on Leduc poker [36]. ReCFR is compared with vanilla CFR, as the latter can be regarded as a special case of ReCFR. We first test ReCFR with a constant $\lambda$ in $[0, 1]$. As shown on the left-hand side in Fig. 1, ReCFR with any $\lambda$ converges. It seems that ReCFR with $\lambda = 10^{-7}$ is the fastest, even faster than vanilla CFR. However, if $\lambda$ is reduced to 0, ReCFR will degenerate to XFP, and it converges slower than CFR. We also test ReCFR with an adaptive $\lambda$, as shown on the right-hand side in Fig. 1. As we can see, ReCFR with different initial $\lambda$ (even 0) converges as fast as CFR, except the one with an initial $\lambda = 1$. So, with the adaptive method, it is easier to choose the hyperparameter for ReCFR, as long as the initial value is small enough. In Appendix F, the curves of the adaptive $\lambda$ in ReCFR with different initial values are given.

### B. Experimental Results of Neural ReCFR-B

The results of all the algorithms on FHP are given in Fig. 2. As we can see, neural ReCFR-B achieves an exploitability of 50.5 mbb/g after touching $1.0 \times 10^{10}$ nodes, whereas deep CFR achieves 47.0 mbb/g after touching $6.0 \times 10^{8}$ nodes, but the value increases to 73.0 mbb/g after touching $1.3 \times 10^{9}$ nodes. Neural ReCFR-B is worse than deep CFR in sample efficiency. It is reasonable because model-free algorithms are less sample efficient by nature. However, our algorithm is faster than the three model-free algorithms: deep OSCFR, DREAM, and NFSP. More importantly, our algorithm is the most training efficient, by more than 25 times faster than deep CFR to reach the exploitability of 100 mbb/g, whereas deep OSCFR and DREAM never reach this value and both DNCFR and NFSP are stuck at this value. Recall that DNCFR also uses bootstrap learning. However, it has a higher exploitability than Neural ReCFR-B. On the other hand, NFSP is a training efficient, but it achieves an exploitability much higher than neural ReCFR-B and deep CFR. Note that NFSP can be regarded as a special case of neural ReCFR-B to some extent, and the latter may degenerate to NFSP when the gap between $u_p^{l\overline{\sigma}^t}(I)$ and $\max_a u_p^{l\overline{\sigma}^t}(I, a)$ at every infoset is too small to be learned by neural networks. As for DREAM, it performs better than deep OSCFR but worse than neural ReCFR-B and deep CFR. This suggests that DREAM is still suffering from high-variance training targets.

We also tried to improve the training efficiency of deep CFR by only reducing the number of SGD steps for training the regret networks. The results in Appendix F show that deep CFRs with fewer SGD steps converge earlier to higher exploitability. On the other hand, as shown in Fig. 3, neural ReCFR-B converges to a similar exploitability even the RSV networks are trained for one epoch per iteration. To further investigate the robustness of neural ReCFR-B, we test it with different settings on FHP. In Fig. 3, the results of neural ReCFR-B with different numbers of epochs for training the RSV networks are given. As we can see, training the RSV networks for more than two epochs can reduce the convergence speed significantly. A possible reason is that the neural networks are overfitting. We conjecture that using memory buffers can partially solve this problem. As shown on the right-hand side in Fig. 3, the problem is alleviated. Overfitting is also observed in deep CFR [18], and it is solved by training the regret networks from scratch at every iteration. In neural ReCFR-B, reducing the number of epochs to alleviate the problem is more appealing. In Fig. 4, the results of neural

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

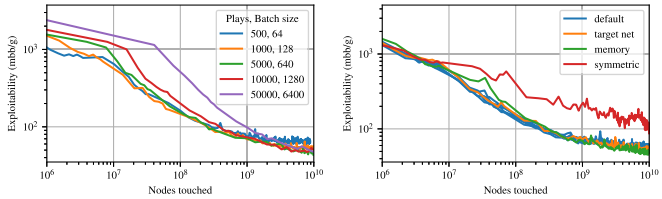


Fig. 5. Left-hand side: Exploitability curves of neural ReCFR-B with different numbers of plays and batch sizes. Right-hand side: Exploitability curves of neural ReCFR-B with additional components. We also plot the exploitability curves of five replicates of neural ReCFR-B (the blue lines) with different random seeds.

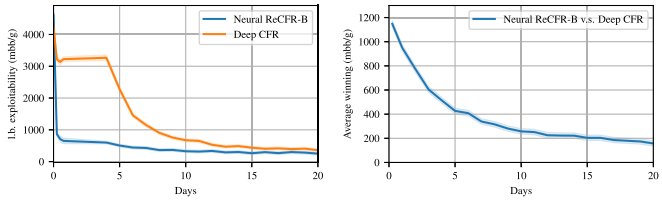


Fig. 6. Comparison of neural ReCFR-B and deep CFR on HULH. The $x$-axes indicate the days for training. Left-hand side: Lower bound on exploitability. Right-hand side: Head-to-head performance. The $y$-axes represent the average winning of neural ReCFR-B (simulated by $10^6$ plays). All the values are presented with 95% confidence interval.

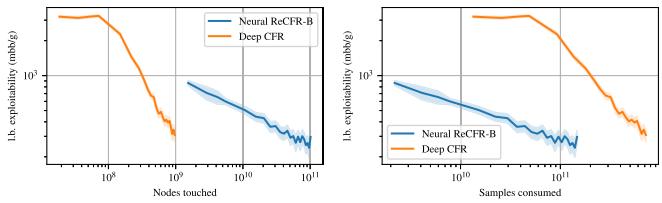


Fig. 7. Curves of lower bounds on exploitability of neural ReCFR-B and deep CFR on HULH. The $x$-axes represent the number of nodes touched and the number of samples consumed, respectively.

ReCFR-B with different numbers of plays per iteration are given. It is shown that neural ReCFR-B is insensitive to the number of plays in sample efficiency. However, performing more plays (collecting more training data) per iteration can increase the training efficiency. This is because of the variance in approximated RSVs is also affected by the dataset size.

On the left-hand side in Fig. 5, we test the scalability of the algorithm by increasing the number of plays per iteration and the batch size simultaneously. As we can see, the algorithm can scale down to 500 plays and up to 50 000 plays per iteration. Performing more plays per iteration and using a larger batch size may reduce the sample efficiency. However, it is easier to parallelize and, thus, may reduce the training time dramatically. In conclusion, neural ReCFR-B is robust to different hyperparameters.

On the right-hand side in Fig. 5, we test neural ReCFR-B with additional components. The default is neural ReCFR-B with asymmetric learning, without memory buffers and target networks. As we can see, neural ReCFR-B with target networks performs identically to the default one. On the other hand, neural ReCFR-B with RSV memory buffers seems to have a slightly lower exploitability. This result suggests that using RSV memory buffers in neural ReCFR-B may increase the sample efficiency. However, neural ReCFR-B with symmetric learning is worse than the default one. The reason might be that symmetric learning cannot guarantee to visit every infoset with a probability greater than zero. Besides, we plot five replicates of neural ReCFR-B with different random seeds (so the neural networks are also initialized differently). As we can see, the results are consistent.

Finally, we evaluate neural ReCFR-B on HULH. Since using RSV memory buffers may increase sample efficiency and avoid overfitting, we use RSV memory buffers in this experiment. We train neural ReCFR-B and deep CFR separately for 20 days on one GPU and 10 CPU cores. On the left-hand side of Fig. 6, we compare the lower bounds on the exploitability of neural ReCFR-B and deep CFR. The lower bounds are estimated using a local best response [39] algorithm. The comparison is performed day by day. So it should reflect both sample efficiency and training efficiency. As we can see, neural ReCFR-B achieves a lower bound around 600 mbb/g only after one day's training, whereas deep CFR reaches this value on the ninth day. After 20 days' training, deep CFR and neural ReCFR-B have similar lower bounds on exploitability, about 300 mbb/g. Besides, Fig. 7 shows that neural ReCFR-B is more training efficient, but deep CFR has a better sample efficiency. Note that the sample efficiency of neural ReCFR-B may be increased when the batch size is reduced at the cost of increasing training time, according to the scalability experiment on FHP. On the right-hand side of Fig. 6, we show the head-to-head performance between neural ReCFR-B and deep CFR. As we can see, neural ReCFR-B beats deep CFR by more than 150 mbb/g. It looks like that both algorithms have not converged yet, but it is safe to conclude that neural ReCFR-B is more training efficient and time efficient.

## VIII. Conclusion

This article proposes a new CFR algorithm, ReCFR, and a model-free neural CFR algorithm, neural ReCFR-B. In these two algorithms, cumulative regrets are replaced by RSVs proposed in [22]. After revisiting RSVs and warm CFR, we prove that ReCFR can converge to a Nash equilibrium at a rate of $O(1/\sqrt{T})$. Thanks to the recursive and noncumulative properties of the RSVs, when bootstrap learning is used, the variance in training targets in neural ReCFR-B should be low. The experimental results show that neural ReCFR-B achieves competitive performance to the state-of-the-art neural CFR algorithms with higher training efficiency.

According to the theoretical results, it is promising to transform other CFR algorithms, e.g., CFR+ [26] and PCFR [40], to new algorithms similar to ReCFR, and develop new neural CFR algorithms based on them. Also, combining neural ReCFR-B with variance-reduction techniques [30], [41] or other improvements [20], [27], [29] may produce more efficient algorithms. Furthermore, there are some other equilibrium-finding algorithms that have fast convergence, e.g., excessive gap technique (EGT) [11] and optimistic follow the regularized lead (OFTRL) [42], and EGT does not even use any cumulative variables. However, it is unclear whether they are compatible

Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.

with sampling and function approximation. So, more research is required in this direction. Besides, it has been shown in [43] that RM is equivalent to follow the regularized lead (FTRL) [44]. In this article, it is shown that ReCFR is a generalization of CFR and XFP. So, there might be some strong connections between CFR, XFP, and FTRL worthy of further study.

With the model-free neural ReCFR-B, we may also apply CFR algorithms to a broader range of IIGs, for example, nonstationary games, nonzero-sum games, multiplayer games, and even video games.

## REFERENCES


[1] G. Tesauro, "Temporal difference learning and TD-Gammon," *Commun. ACM*, vol. 38, no. 3, pp. 58–68, 1995.

[2] R. Coulom, "Efficient selectivity and backup operators in Monte-Carlo tree search," in *Proc. 5th Int. Conf. Comput. Games*. 2006, vol. 4630, pp. 72–83.

[3] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[4] D. Silver *et al.*, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[5] P. I. Cowling, E. J. Powley, and D. Whitehouse, "Information set Monte Carlo tree search," *IEEE Trans. Comput. Intell. AI Games*, vol. 4, no. 2, pp. 120–143, Jun. 2012.

[6] N. Brown and T. Sandholm, "Safe and nested subgame solving for imperfect-information games," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 689–699.

[7] S. Srinivasan *et al.*, "Actor-critic policy optimization in partially observable multiagent environments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3426–3439.

[8] M. Zinkevich, M. Johanson, M. H. Bowling, and C. Piccione, "Regret minimization in games with incomplete information," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, vol. 20, pp. 1729–1736.

[9] J. Heinrich, M. Lanctot, and D. Silver, "Fictitious self-play in extensive-form games," in *Proc. Int. Conf. Mach. Learn.*, 2015, vol. 37, pp. 805–813.

[10] B. Bosanský, C. Kiekintveld, V. Lisý, J. Cermak, and M. Pechoucek, "Double-oracle algorithm for computing an exact Nash equilibrium in zero-sum extensive-form games," in *Proc. Int. Conf. Auton. Agents Multi-Agent Syst.*, 2013, pp. 335–342.

[11] S. Hoda, A. Gilpin, J. Peña, and T. Sandholm, "Smoothing techniques for computing Nash equilibria of sequential games," *Math. Operations Res.*, vol. 35, no. 2, pp. 494–512, 2010.

[12] M. Moravčík *et al.*, "DeepStack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[13] N. Brown and T. Sandholm, "Superhuman AI for heads-up no-limit poker: Libratus beats top professionals," *Science*, vol. 359, no. 6374, pp. 418–424, 2018.

[14] N. Brown and T. Sandholm, "Superhuman AI for multiplayer poker," *Science*, vol. 365, no. 6456, pp. 885–890, 2019.

[15] M. Johanson, N. Burch, R. A. Valenzano, and M. Bowling, "Evaluating state-space abstractions in extensive-form games," in *Proc. Int. Conf. Auton. Agents Multi-Agent Syst.*, 2013, pp. 271–278.

[16] K. Waugh, D. Morrill, J. A. Bagnell, and M. H. Bowling, "Solving games with functional regret estimation," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2138–2145.

[17] D. R. Morrill, "Using regret estimation to solve games compactly," M.S. thesis, Dept. Comput. Sci., Univ. Alberta, Edmonton, AB, Canada, 2016.

[18] N. Brown, A. Lerer, S. Gross, and T. Sandholm, "Deep counterfactual regret minimization," in *Proc. Int. Conf. Mach. Learn.*, 2019, vol. 97, pp. 793–802.

[19] H. Li, K. Hu, S. Zhang, Y. Qi, and L. Song, "Double neural counterfactual regret minimization," in *Proc. Int. Conf. Learn. Representations*, 2020.

[20] E. Steinberger, A. Lerer, and N. Brown, "DREAM: Deep regret minimization with advantage baselines and model-free learning," 2006, *arXiv:2006.10410*.

[21] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Berlin, Germany: Springer, 2009.

[22] N. Brown and T. Sandholm, "Strategy-based warm starting for regret minimization in games," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 432–438.

[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[24] N. Burch, M. Johanson, and M. Bowling, "Solving imperfect information games using decomposition," in *Proc. AAAI Conf. Artif. Intell.*, 2014, pp. 602–608.

[25] N. Brown, T. Sandholm, and B. Amos, "Depth-limited solving for imperfect-information games," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7663–7674.

[26] O. Tammelin, "Solving large imperfect information games using CFR+," 2014, *arXiv:1407.5042*.

[27] E. Steinberger, "Single deep counterfactual regret minimization," 2019, *arXiv:1901.07621*.

[28] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," 2016, *arXiv:1603.01121*.

[29] A. Gruslys *et al.*, "The advantage regret-matching actor-critic," 2020, *arXiv:2008.12234*.

[30] M. Schmid, N. Burch, M. Lanctot, M. Moravcik, R. Kadlec, and M. Bowling, "Variance reduction in Monte Carlo counterfactual regret minimization (VR-MCCFR) for extensive form games using baselines," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 2157–2164.

[31] E. Lockhart *et al.*, "Computing approximate equilibria in sequential adversarial games by exploitability descent," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 464–470.

[32] M. Lanctot, K. Waugh, M. Zinkevich, and M. H. Bowling, "Monte Carlo sampling for regret minimization in extensive games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1078–1086.

[33] M. Johanson, N. Bard, M. Lanctot, R. G. Gibson, and M. Bowling, "Efficient Nash equilibrium approximation through Monte Carlo counterfactual regret minimization," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2012, pp. 837–846.

[34] R. G. Gibson, M. Lanctot, N. Burch, D. Szafron, and M. Bowling, "Generalized sampling and variance in counterfactual regret minimization," *Proc. AAAI Conf. Artif. Intell.*, vol. 26, no. 1, pp. 1355–1361, 2012.

[35] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 279–292, 1992.

[36] F. Southey *et al.*, "Bayes' Bluff: Opponent modeling in poker," in *Proc. Conf. Uncertainty Artif. Intell.*, 2005, pp. 550–558.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.

[38] M. Lanctot *et al.*, "OpenSpiel: A framework for reinforcement learning in games," 2019, *arXiv:1908.09453*.

[39] V. Lisý and M. Bowling, "Eqilibrium approximation quality of current no-limit poker bots," in *Proc. Workshops AAAI Conf. Artif. Intell.*, 2017.

[40] G. Farina, C. Kroer, and T. Sandholm, "Faster game solving via predictive Blackwell approachability: Connecting regret matching and mirror descent," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 5363–5371.

[41] T. Davis, M. Schmid, and M. Bowling, "Low-variance and zero-variance baselines for extensive-form games," in *Proc. Int. Conf. Mach. Learn.*, 2020, vol. 119, pp. 2392–2401.

[42] G. Farina, C. Kroer, and T. Sandholm, "Optimistic regret minimization for extensive-form games via dilated distance-generating functions," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2019, pp. 5222–5232.

[43] K. Waugh and J. A. Bagnell, "A unified view of large-scale zero-sum equilibrium computation," in *Proc. Workshops AAAI Conf. Artif. Intell.*, 2015.

[44] J. D. Abernethy, E. Hazan, and A. Rakhlin, "Competing in the dark: An efficient algorithm for bandit linear optimization," in *Proc. Conf. Learn. Theory*, 2008, pp. 263–274.


Authorized licensed use limited to: New York University. Downloaded on June 30,2025 at 06:23:09 UTC from IEEE Xplore. Restrictions apply.