

No-Regret Strategy Solving in Imperfect-Information Games via Pre-Trained Embedding

Yanchang Fu^{1,2} Shengda Liu² Pei Xu² Kaiqi Huang^{2*}

¹School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, P.R.China

²The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

{fuyanchang2020, shengda.liu, pei.xu}@ia.ac.cn, kqhuang@nlpr.ia.ac.cn

Abstract

High-quality information set abstraction remains a core challenge in solving large-scale imperfect-information extensive-form games (IIEFGs)—such as no-limit Texas Hold’em—where the finite nature of spatial resources hinders solving strategies for the full game. State-of-the-art AI methods rely on pre-trained discrete clustering for abstraction, yet their hard classification irreversibly discards critical information: specifically, the quantifiable subtle differences between information sets—vital for strategy solving—thus compromising the quality of such solving. Inspired by the word embedding paradigm in natural language processing, this paper proposes the Embedding CFR algorithm, a novel approach for solving strategies in IIEFGs within an embedding space. The algorithm pre-trains and embeds the features of individual information sets into an interconnected low-dimensional continuous space, where the resulting vectors more precisely capture both the distinctions and connections between information sets. Embedding CFR introduces a strategy-solving process driven by regret accumulation and strategy updates in this embedding space, with supporting theoretical analysis verifying its ability to reduce cumulative regret. Experiments on poker show that with the same spatial overhead, Embedding CFR achieves significantly faster exploitability convergence compared to cluster-based abstraction algorithms, confirming its effectiveness. Furthermore, to our knowledge, it is the first algorithm in poker AI that pre-trains information set abstractions via low-dimensional embedding for strategy solving.

Code — <https://github.com/PhilEnchan/EmbeddingCFR>

1 Introduction

Solving large-scale imperfect-information games like no-limit Texas Hold’em remains a pivotal challenge in AI research. Systems including DeepStack (Moravčík et al. 2017), Libratus (Brown and Sandholm 2018), and Pluribus (Brown and Sandholm 2019b) have achieved superhuman performance through game-theoretic equilibrium computation, yet their success hinges on addressing a critical dilemma: handling enormous decision spaces with limited spatial resources.

Counterfactual Regret Minimization (CFR) serves as the foundational algorithm for computing ϵ -Nash equilibrium, but its linear space complexity becomes prohibitive

even for moderate-scale games. Consider heads-up limit Texas Hold’em: with $\approx 10^{13}$ information sets, it demands 523 terabytes of RAM (Johanson 2013)—a scale rendering direct CFR implementation physically impossible, let alone larger games like heads-up no-limit. Existing systems thus trade theoretical rigor for practicality via approximations, with the prevailing “abstraction-solving-translation” paradigm (Gilpin, Sandholm, and Sørensen 2007) as the core approach: compressing the original game into a feasible abstract version, solving for ϵ -Nash equilibrium within it, and mapping the solved strategy back to the original game. One key approach within this paradigm, known as information set abstraction, is a pre-training method that groups similar information sets into discrete equivalence classes prior to strategy solving, and uniform strategies are applied to these classes to reduce spatial resource demands. This approach is validated by the aforementioned superhuman AI systems, which leveraged it to defeat top human players.

However, existing information set abstraction algorithms have critical limitations. Relying predominantly on many-to-one mappings, they often introduce arbitrary classification boundaries. As shown in Figure 1(c), consider two hands in the flop round of Texas Hold’em (see Appendix A.1 for rules)¹: hand $\blacksquare (9_s^p A_s^p 9_h \mathcal{J}_s Q_s)^2$ and hand $\bullet (T_s^p A_s^p T_h \mathcal{J}_s Q_s)$; both hands forming a pair. While superficially similar, hand \bullet can form a straight flush with K_s , making it inherently stronger than hand \blacksquare (which can realistically only form a flush with any additional spade card). Current algorithms face a dilemma: keeping these hands—and the information sets they represent—separate wastes the similar strategies that would naturally follow from their inherent similarities, while grouping them overlooks nuanced differences. This binary, low-granularity approach introduces arbitrariness that degrades strategy quality.

In this paper, we propose **Embedding CFR**, a novel strategy-solving paradigm for information set abstraction, to address the aforementioned limitations. Unlike traditional methods that map multiple information sets to a single equivalence class, it maps each information set to a multi-dimensional vector—embedding coordinates forming

¹All appendix content is available in the extended version.

²h: hearts, d: diamonds, s: spades, c: clubs; superscripts p denote private cards (e.g., $8_h^p 8_d^p$ = player’s 8s in hearts/diamonds).

*Corresponding author.

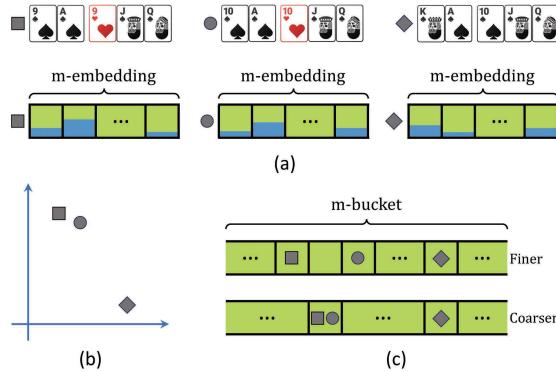


Figure 1: **Behavior comparison of hand representations under Embedding CFR and traditional information set abstraction** for hands \blacksquare , \bullet , and \blacklozenge in Texas Hold’em.

(a) Embedding CFR maps hands to embedding coordinates, which form an m -dimensional probability distribution where the sum of values across all dimensions equals 1.
(b) Schematic 2D projection of embedding coordinates illustrates the geometric topology between hands, highlighting both similarity (closeness of \blacksquare and \bullet) and distinction (separation from \blacklozenge).
(c) Traditional abstraction maps information sets to a fixed number of m abstracted classes, e.g. buckets, forcing binary decisions for these hands: either refining \blacksquare and \bullet into distinct equivalence classes or coarsening them into one. This lack of intermediate states hinders exploitation of inter-information-set similarity for strategy solving.

a probability distribution—thereby enhancing abstraction expressiveness. As Figure 1(a) illustrates, hands \blacksquare , \bullet , and \blacklozenge (the latter a strong straight flush: $K_s^P A_s^P T_s^J Q_s^Q$ at the flop) are assigned embedding coordinates representing their confidence in corresponding dimensions. During strategy solving, updates to information sets preferentially influence high-confidence dimensions, ensuring strategies associated with closer coordinates exhibit greater similarity. Figure 1(b) shows a schematic 2D projection of this embedding, where hand \blacksquare and hand \bullet cluster closely yet remain subtly distinguishable—yielding correspondingly similar solved strategies—while being distinctly separated from the strong hand \blacklozenge , ensuring minimal mutual influence during solving.

This paper makes the following contributions:

1. To our knowledge, at least in poker AI development, ours is the first work to employ a pre-trained embedding approach for information set abstraction.
2. We present a framework for solving strategies using no-regret optimization under the condition of information set embedding, and provide an approximate analysis of its ability to achieve regret decrease.
3. We successfully apply Embedding CFR to poker AI development and propose an embedding construction algorithm for poker.

Experiments are conducted in a poker game, comparing with traditional information set abstraction algorithms such

as EHS (Gilpin and Sandholm 2007a), PaEmd (the information set abstraction algorithm adopted by Libratus, recognized as a state-of-the-art approach in the field) (Ganzfried and Sandholm 2014), and a recent work KrwEmd (Fu et al. 2025). The results show that the strategy solved by Embedding CFR has lower exploitability under the same spatial overhead and update iterations, demonstrating the effectiveness of the proposed algorithm.

1.1 Related Works

Our work lies within the community of solving imperfect-information extensive-form games via no-regret learning, with CFR (Zinkevich et al. 2007) as the core strategy-solving algorithm. Extensions follow two distinct directions: sampling-based methods like MCCFR (Lanctot et al. 2009), and strategy update-focused optimizations including CFR+ (Bowling et al. 2015), DCFR (Brown and Sandholm 2019a), and PCFR+ (Farina, Kroer, and Sandholm 2021).

For strategy solving under limited space constraints, related efforts include pruning and action abstraction techniques: regret-based pruning (Brown and Sandholm 2015), best-response pruning (Brown and Sandholm 2017), dynamic thresholding pruning (Brown, Kroer, and Sandholm 2017), along with reinforcement learning-based online action abstraction methods like RL-CFR (Li, Fang, and Huang 2024) and EVPA (Li and Huang 2025).

Closer to our approach, neural network-based regret estimation methods (DeepStack (Moravčík et al. 2017), ReBeL (Brown et al. 2020), Deep CFR (Brown et al. 2019), Escher (McAleer et al. 2023)) replace tabular storage of regrets and strategies to simplify solving. They excel in model-free scenarios, where reliance on environmental interactions—which simultaneously drive strategy optimization and information set learning—reduces dependence on complex domain knowledge. In model-known settings, however, this dual reliance creates trade-offs, blunting efficiency compared to pre-training-based methods leveraging prior domain knowledge.

Our closest predecessors are information set abstraction works: lossless abstraction applicable to small-scale games (Gilpin and Sandholm 2007b), expectation-based EHS (Gilpin and Sandholm 2007a), potential-aware methods (Gilpin, Sandholm, and Sørensen 2007; Ganzfried and Sandholm 2014) (notably PaEmd), and the history trajectory-based KrwEmd (Fu et al. 2025). We share with these predecessors a pre-training-then-solving paradigm—analogous to NLP pipelines where word embedding or tokenization precedes semantic processing—by first preprocessing information sets before strategy solving.

2 Background and Notation

An imperfect-information extensive-form game (IIEFG) is defined by the tuple $(\mathcal{N}, H, A, \mathcal{P}, u, \sigma_c, \mathcal{I})$. The set $\mathcal{N} = \{1, \dots, N\} \cup \{c\}$ represents the finite set of players; in this work, we focus on the two-player case where $N = 2$, with c being a special **chance** player whose actions model stochastic events. The set H consists of histories (also referred to as nodes), where each $h \in H$ represents a sequence of actions

from the game's start, with the empty sequence h^0 denoting the unique initial history. We write $h \sqsubseteq h'$ if h is a prefix of h' , and $h \sqsubset h'$ if it is a strict prefix; $h \cdot a$ denotes the history formed by appending action $a \in A(h)$ to h . Here, $A(h)$ is the set of available actions for non-terminal histories $h \in H \setminus Z$, with $Z \subset H$ being the set of terminal histories that end the game. The player function $\mathcal{P} : H \setminus Z \rightarrow \mathcal{N}$ assigns a unique acting player to each non-terminal history. The utility function $u = (u_1, \dots, u_N)$ gives each player $i \in \mathcal{N}$ a real-valued payoff $u_i(z)$ for every terminal history $z \in Z$. In the two-player zero-sum setting considered here, $u_1(z) = -u_2(z)$ for all $z \in Z$. The function σ_c models the chance player's behavior by specifying, for each history h where $\mathcal{P}(h) = c$, a probability distribution $\sigma_c(h, a)$ over actions $a \in A(h)$, capturing the game's stochastic nature.

A central concept in imperfect-information extensive-form games is the **information set (infoset)**, which captures the inherent uncertainty a player faces regarding the exact game history when making a decision. Formally, $\mathcal{I} = \bigcup_{i \in \mathcal{N} \setminus \{c\}} \mathcal{I}_i$ represents the collection of infosets, where each \mathcal{I}_i is a partition of the set $H_i = \{h \in H \setminus Z \mid \mathcal{P}(h) = i\}$. That is, each history in which player i takes an action belongs to exactly one infoset. Within any $I \in \mathcal{I}_i$, the player cannot distinguish between the histories $h, h' \in I$, implying the following consistency conditions:

$$\forall h, h' \in I : \begin{cases} \mathcal{P}(h) = \mathcal{P}(h') = i = \mathcal{P}(I), \\ A(h) = A(h') = A(I). \end{cases}$$

For infoset I , we define Z_I as the set of terminal histories that can be reached from some $h \in I$. Throughout this work, we assume the game adheres to the property of **perfect recall**, which stipulates that players retain complete knowledge of all information they have previously encountered. More formally, for player i , if two different histories are not part of the same infoset, then no subsequent continuations of these histories can be grouped within the same infoset for that player. An important implication of this property is that each branch from the root to the leaves of the game tree traverses any given infoset at most once. We use the notation $z[I]$ to denote the history prefix of a terminal history z that passes through infoset I . This notation is undefined (or invalid) if the path from h^0 to z does not traverse I .

A player's strategy $\sigma_i \in \Sigma_i$ assigns a probability $\sigma_i(I, a)$ to each action a at every infoset $I \in \mathcal{I}_i$. A strategy profile $\sigma = (\sigma_1, \dots, \sigma_N)$ specifies a complete strategy for each player. Given σ , the reach probability of a history $h \in H$ is defined as $\pi^\sigma(h) = \pi^\sigma(h^0, h)$, where $\pi^\sigma(h, h') = \prod_{i \in \mathcal{N}} \pi_i^\sigma(h, h')$, with

$$\pi_i^\sigma(h, h') = \begin{cases} 0, & \text{if } h \not\sqsubseteq h' \\ \prod_{h'': h \sqsubseteq h'', h'' \cdot a \sqsubseteq h', \mathcal{P}(h'')=i} \sigma_i(h'', a), & \text{if } h \sqsubseteq h' \end{cases}.$$

The contribution of player j to the reach probability of history h is denoted by $\pi_j^\sigma(h) = \pi_j^\sigma(h^0, h)$. In later sections, we may slightly abuse notation by writing $\sigma_i(h, a)$ to denote the probability assigned to action a at the infoset containing h . We similarly use the subscript $-i$ in both π_{-i}^σ and σ_{-i} to refer to the reach probability and strategy profile, respectively, of all players except player i .

In game theory, the **Nash equilibrium** is a crucial solution concept: a state where no player can enhance their payoff by unilaterally altering their strategy, and to some extent the optimal solution. The payoff for player i under strategy profile σ is $u_i(\sigma) = \sum_{z \in Z} u_i(z)\pi^\sigma(z)$. The **best response** value for player i against σ_{-i} is $b_i(\sigma_{-i}) = \max_{\sigma'_i \in \Sigma_i} u_i(\sigma'_i, \sigma_{-i})$, the max payoff for optimal response. A strategy profile σ^* is an ϵ -**Nash equilibrium** if $\forall i \in \mathcal{N} \setminus \{c\}$, $u_i(\sigma^*) + \epsilon \geq b_i(\sigma_{-i}^*)$. When $\epsilon = 0$, it's a standard Nash equilibrium. In two-player zero-sum games, the **exploitability** of σ is $\epsilon^\sigma = b_1(\sigma_2) + b_2(\sigma_1)$, measuring players' vulnerability to an optimal opponent. Lower exploitability means closer to equilibrium.

2.1 Counterfactual Regret Minimization

Counterfactual Regret Minimization (CFR), also referred to as Vanilla CFR, is an iterative algorithm for computing approximate Nash equilibrium in IIEFGs by minimizing **counterfactual regret** (Zinkevich et al. 2007). For player i , the total regret after T iterations is defined as

$$R_i^T = \frac{1}{T} \max_{\sigma'_i \in \Sigma_i} \sum_{t=1}^T (u_i(\sigma'_i, \sigma_{-i}^t) - u_i(\sigma^t)).$$

In two-player zero-sum IIEFGs, if both players satisfy $R_i^T \leq \epsilon$, then the average strategy forms a 2ϵ -Nash equilibrium (Waugh 2009).

CFR avoids the difficulty of directly minimizing global regret by instead minimizing cumulative counterfactual regret at each infoset. It defines the **counterfactual value**:

$$v_i^\sigma(I) = \sum_{z \in Z_I} \pi_{-i}^\sigma(z[I])\pi^\sigma(z[I], z)u_i(z),$$

and the **immediate counterfactual regret**:

$$r^T(I, a) = v_{\mathcal{P}(I)}^{\sigma_{I \rightarrow a}^T}(I) - v_{\mathcal{P}(I)}^{\sigma^T}(I), \quad (1)$$

where $\sigma_{|I \rightarrow a}$ is a strategy profile identical to σ except that player $\mathcal{P}(I)$ always plays action a at I .

Then, the **cumulative counterfactual regret**

$$R^T(I, a) = \frac{1}{T} \sum_{t=1}^T r^t(I, a) \quad (2)$$

is used to update the immediate strategy via **regret matching** (Hart and Mas-Colell 2000):

$$\sigma_{\mathcal{P}(I)}^{T+1}(I, a) = \begin{cases} \frac{R^T(I, a)_+}{\sum_{a' \in A(I)} (R^T(I, a')_+)}, & \text{if } \sum_{a' \in A(I)} (R^T(I, a')_+) > 0, \\ \frac{1}{|A(I)|}, & \text{otherwise,} \end{cases}$$

where $x_+ = \max(x, 0)$.

This process causes $\sum_{I \in \mathcal{I}_i} (\max_{a \in A(I)} R^T(I, a)_+)$ to converge to zero at a rate of $\mathcal{O}(1/\sqrt{T})$, which is an upper bound of R_i^T . Therefore, the average strategy profile $\bar{\sigma}^T = \langle \bar{\sigma}_1^T, \bar{\sigma}_2^T \rangle$ converges to an ϵ -Nash equilibrium, where

$$\bar{\sigma}_{i=\mathcal{P}(I)}^T(I, a) = \frac{\sum_{t=1}^T \pi_i^{\sigma^t}(I) \sigma_i^t(I, a)}{\sum_{t=1}^T \pi_i^{\sigma^t}(I)}.$$

3 Embedding CFR

This section introduces the Embedding CFR algorithm, a strategy-solving approach that leverages embedding for information set abstraction. We first extend the definition of information set abstraction (Section 3.1), then detail the algorithm’s driving process (Section 3.2), and finally analyze its regret convergence trends in low-dimensional spaces under restricted conditions (Section 3.3).

3.1 Problem Modeling

We begin our approach by redefining information set abstraction to expand its conceptual scope. Specifically, the abstraction takes **info-blocks** as its primary objects of operation. In an IIEFG, an **info-block partition** is defined as a partition of a player i ’s infosets collection \mathcal{I}_i into non-overlapping groups $\mathcal{J} = \{J_1, J_2, \dots, J_K\}$. Each info-block J_k aggregates infosets that are strategically relevant or similar, under the constraint that all infosets $I, I' \in J_k$ share the same action space ($A(I) = A(I')$). The construction of info-block partitions is non-unique and must be predefined manually prior to information set abstraction, requiring game-specific analysis to determine valid groupings; we will discuss the construction of info-block partitions in the context of specific games in Section 4.1.

Definition 3.1 (Extended Information Set Abstraction). For an info-block $J = \{I_1, \dots, I_n\}$ containing n infosets of player i , **extended information set abstraction** is defined as the strategic association of J with a set of m advisors $E = \{e_1, e_2, \dots, e_m\}$. Each advisor $e_p \in E$ is assigned a strategy function $\sigma(e_p, \cdot): A(J) \rightarrow [0, 1]$ forming a valid probability distribution over $A(J)$, where $A(J)$ denotes the shared action space of all infosets in J . The strategy of player i at an infoset $I_q \in J$ is determined by an m -ary aggregation function f_q , such that for any action $a \in A(J)$:

$$\sigma(I_q, a) = f_q(\sigma(e_1, a), \sigma(e_2, a), \dots, \sigma(e_m, a)).$$

Advisors act as fixed-strategy decision-makers, adopting a uniform strategy across all infosets in J ; for each infoset, the player predefines a method to construct its strategy by aggregating these advisor strategies. Extended information set abstraction leverages this structure to reduce storage overhead: instead of maintaining n distinct strategies for each infoset in J , it retains m advisor strategies over E . When $m < n$, this yields a space complexity of $\mathcal{O}(m \cdot |A(J)|)$, outperforming the $\mathcal{O}(n \cdot |A(J)|)$ requirement of traditional abstractions. Notably, this framework generalizes traditional information set abstraction: if, for each $I_q \in J$, the aggregation function f_q reduces to an identity mapping onto a single advisor e_p (i.e., $\sigma(I_q, a) = \sigma(e_p, a)$ for all $a \in A(J)$), the two abstractions coincide.

3.2 Driving Process for Embedding CFR

For ease of discussion, while $J = \{I_1, \dots, I_n\}$ and $E = \{e_1, e_2, \dots, e_m\}$ are originally set-theoretic concepts for player i , we arrange them into column vectors $\mathbf{J} = [I_1, \dots, I_n]^\top$ and $\mathbf{E} = [e_1, \dots, e_m]^\top$ to facilitate the application of linear algebra tools. Furthermore, for any function g , its action on \mathbf{J} or \mathbf{E} is defined element-wise as

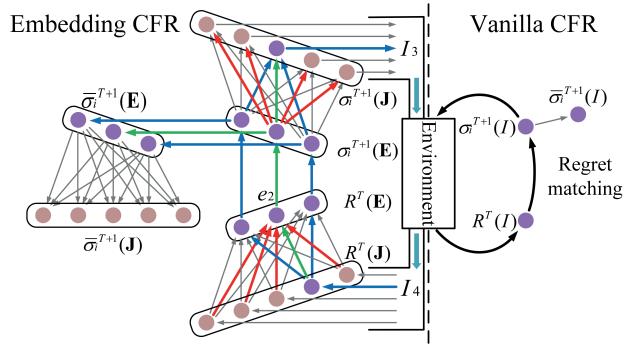


Figure 2: Schematic comparison of driving processes: Embedding CFR vs. Vanilla CFR

$$\begin{cases} g(\mathbf{J}) = [g(I_1), g(I_2), \dots, g(I_n)]^\top, \\ g(\mathbf{E}) = [g(e_1), g(e_2), \dots, g(e_m)]^\top. \end{cases} \quad (3)$$

Building on this notation, we propose **Embedding CFR**, which models the extended information set abstraction problem as a strategy voting framework, where a **embedding matrix** $\Phi = [\phi]_{m \times n}$ is constructed. Each non-negative element $\phi_{p,q} \geq 0$ —with the constraint that $\sum_{p=1}^m \phi_{p,q} = 1$ for each q —represents the confidence in trusting advisor e_p at infoset I_q ; here, $\Phi_{\cdot,q}$ denotes the embedding coordinates of infoset I_q . Given a strategy $\sigma_i^T(\mathbf{E})$ in the advisor space (also called embedding space), decisions in the original strategy space are made via the mapping

$$\sigma_i^T(\mathbf{J}, a) = \Phi^\top \sigma_i^T(\mathbf{E}, a) \quad (4)$$

for each $a \in A(\mathbf{J})$.

To solve games under this abstraction, Embedding CFR operates as a no-regret learning algorithm that draws insights from Vanilla CFR (Zinkevich et al. 2007): as illustrated in the right half of Figure 2 and as we have detailed in Section 2.1, Vanilla CFR updates the cumulative counterfactual regret $R^T(I)$ of the target infoset I at iteration T , then derives the immediate strategy $\sigma_i^{T+1}(I)$ for next-iteration online interaction with the environment via regret matching, after which the regret for the new iteration is updated—forming a driving process. The strategy to be evaluated, which exhibits favorable convergence properties, is the average strategy $\bar{\sigma}_i^T(I)$. Embedding CFR extends this process by addressing how to drive it within the newly introduced advisor space.

As illustrated in the left half of Figure 2, when collecting regrets from interaction with the environment, we first decompose regrets from the original space into the advisor space based on embedding coordinates. Using Equations (1), (2) and (3), we define the embedded immediate regret as $r^T(\mathbf{E}, a) = \Phi \cdot r^T(\mathbf{J}, a)$, and the embedded cumulative regret as $R^T(\mathbf{E}, a) = \frac{1}{T} \sum_{t=1}^T r^t(\mathbf{E}, a)$. From these, it follows that the embedded cumulative regret satisfies:

$$R^T(\mathbf{E}, a) = \Phi R^T(\mathbf{J}, a). \quad (5)$$

Each advisor $p = 1, \dots, m$ then iteratively updates its embedded immediate strategy via regret matching based on its embedded cumulative regret:

$$\sigma_i^{T+1}(e_p, a) = \begin{cases} \frac{R^T(e_p, a)_+}{\sum_{a' \in A(J)} (R^T(e_p, a')_+)}, & \text{if } \sum_{a' \in A(J)} (R^T(e_p, a')_+) > 0, \\ \frac{1}{|A(J)|}, & \text{otherwise,} \end{cases} \quad (6)$$

The embedded immediate strategy is, on one hand, mapped back to the immediate strategy in the original space via Equation (4) and used to interact with the environment to initiate a new iteration; on the other hand, it accumulates into the embedded average strategy in the advisor space:

$$\bar{\sigma}^{T+1}(\mathbf{E}, a) = \frac{T}{T+1} \bar{\sigma}^T(\mathbf{E}, a) + \frac{1}{T+1} \sigma_i^{T+1}(\mathbf{E}, a) \quad (7)$$

The average strategy in the original space is recovered via the formula

$$\bar{\sigma}_i^T(\mathbf{J}, a) = \Phi^\top \bar{\sigma}_i^T(\mathbf{E}, a), \quad (8)$$

which serves as the final learned strategy after the iterative process.

It is important to note that while regret embedding (Equation (5)) and strategy recovery (Equations (4), (8)) involve full matrix multiplications—specifically of sizes $m \times n$ with $n \times m$ and $n \times m$ with $m \times n$ —this might initially suggest unavoidable $\mathcal{O}(n)$ or higher space complexity. Our solution addresses this by framing regret embedding as a sampling process and strategy recovery as a query procedure. Rather than operating on the full set of infosets, we dynamically engage a subset of $l < n$ infosets $I_{(1)}, \dots, I_{(l)}$ (mapping to indices j_1, \dots, j_l in \mathbf{J}) at each iteration. As visualized in Figure 2, this selective focus is concretely illustrated: iteration T activates only the single infoset I_4 , while iteration $T+1$ shifts to I_3 .

This design yields a critical efficiency gain: throughout the process, storage is allocated solely to advisor-space quantities $R^T(\mathbf{E}, a)$, $\sigma_i^T(\mathbf{E}, a)$, and $\bar{\sigma}^T(\mathbf{E}, a)$ for each $a \in A(J)$, with no need to reserve space for their original-space counterparts. To operationalize the sampling-based embedding, we use the $\tilde{R}^T(\mathbf{E}, a)$ in place of $R^T(\mathbf{E}, a)$:

$$\tilde{R}^T(\mathbf{E}, a) = \frac{T-1}{T} \tilde{R}^{T-1}(\mathbf{E}, a) + \frac{1}{T} \sum_{k=1}^l \Phi_{\cdot, j_k} r^T(I_k, a).$$

Correspondingly, the query-based recovery retrieves original-space strategies from the advisor space via:

$$\sigma_i^T(I_{(k)}, a) = (\Phi_{\cdot, j_k})^\top \sigma_i^T(\mathbf{E}, a).$$

An additional consideration is how to efficiently retrieve the embedding coordinates Φ_{\cdot, j_k} for each infoset $I_{(k)}$ —this will be discussed in the next section. Notably, each Φ_{\cdot, j_k} only incurs $\mathcal{O}(m)$ space overhead, which does not significantly add to overall storage requirements. The blue and green arrows in Figure 2 illustrate these dynamic pathways, with purple nodes representing the quantities that need to be stored in this iteration—emphasizing that this paired sampling-and-query mechanism ensures Embedding CFR operates with $\mathcal{O}(m)$ space complexity. Pseudocode for poker-specific Embedding CFR is in Appendix B.

3.3 Convergence Analysis

Analyzing the convergence of the Embedding CFR algorithm presents significant challenges, which we acknowledge upfront.

First, establishing the convergence of cumulative regrets in the original space remains non-trivial: embedding inherently introduces information loss, and the quality of solutions recovered in the original space is tightly coupled with the construction of the embedding matrix Φ . To illustrate this complexity, consider an extreme scenario where all infosets rely on a single advisor p with full confidence. In such cases, the iterative updates would lack the diversity needed to explore the solution space effectively, making it implausible to converge to a high-quality solution.

Second, even in the advisor space, proving that the embedded cumulative regret of a single advisor stably converges to zero remains challenging. This is because each infoset is simultaneously shaped by all advisors—a dynamic where their update processes are mutually interdependent, with adjustments to one advisor rippling through and altering the convergence trajectory of others.

Fortunately, we can show that when an advisor acts in isolation, Embedding CFR ensures its regret decreases. Embedding CFR can be conceptualized as follows: in iteration T , when player i encounters an infoset $I_q \in J$ (J is an info-block relevant to player i) during the game, a random selection is made to decide with probability $\phi_{p,q}$ to act according to the strategy $\sigma_i^T(e_p)$. Consider an approximate sampling scenario in Embedding CFR, each iteration strictly needs to consider all infosets in J ; and the key difference is that for each infoset, a selection must be made among e_1, \dots, e_m —choosing to interact and update according to a specific advisor’s strategy. At a particular iteration T , it happens that player i selects e_p for every infoset I_q (as in Figure 2’s green/red arrows, where all infosets choose e_2); this approximates an advisor’s isolated impact on its regret.

Proposition 1. Let $S_p^T = \sum_{a \in A(J)} (R^T(e_p, a)_+)^2$ and $\Delta_J = \max_{\substack{I \in J \\ a, a' \in A(J)}} |v_{\mathcal{P}(I)}^{\sigma_{|I \rightarrow a}}(I) - v_{\mathcal{P}(I)}^{\sigma_{|I \rightarrow a'}}(I)|$. In the

aforementioned scenario, the following holds: If $\frac{S_p^T}{n|A(J)| \cdot \Delta_J^2 \cdot \|\Phi_{p,\cdot}\|_2^2} \leq \frac{T}{T+1}$, then $S_p^{T+1} \leq \frac{n|A(J)| \cdot \Delta_J^2 \cdot \|\Phi_{p,\cdot}\|_2^2}{T+1}$. Otherwise, $S_p^{T+1} \leq \frac{T}{T+1} S_p^T$ (see Appendix C for the proof).

In Proposition 1, S_p^T quantifies the overall embedded cumulative positive regret level associated with e_p . The proposition shows that when S_p^T is less than a specified threshold, S_p^{T+1} can be bounded by a definite decreasing value. When S_p^T exceeds this threshold, although no explicit upper bound for S_p^{T+1} can be provided, it is guaranteed to be a decreasing quantity relative to S_p^T . This conclusion offers an approximate analysis of regret reduction when a single advisor acts independently. By extension, it provides intuition that even when all advisors act together (with mutual influences), each advisor’s regret tends to exhibit a decreasing trend, meaning Embedding CFR can achieve regret-decreasing learning within the advisor space.

4 Application To Poker Games

We now discuss applying the Embedding CFR algorithm to strategy solving in typical imperfect-information extensive-form games like poker, using domain knowledge.

4.1 Partitioning of Info-blocks by Chance Actions

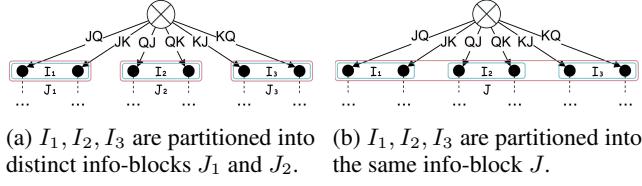


Figure 3: Two info-block partitioning schemes for player 1’s infosets I_1, I_2, I_3 in Kuhn Poker, noting that all sets satisfy action-space consistency ($A(I_1) = A(I_2) = A(I_3)$).

First, we address the unresolved issue of constructing info-block partitions from Section 3.1.

On one hand, as noted, such partitions (exemplified in Kuhn Poker; Figure 3, rules in Appendix A.2) admit non-unique constructions, and not all partitions facilitate strategy solving. Trivially isolating each infoset into individual blocks (Figure 3(a)) undermines the generalization benefits of correlations among similar infosets; intuitively, the framework functions better with more aggregated infosets per block. Conversely, merging all infosets with matching action spaces is flawed: in heads-up limit Texas Hold’em (rules in Appendix A.1), pre-flop and post-flop infosets may share action spaces (call/raise/fold) but are strategically distinct, making merger inappropriate.

On the other hand, algorithmically, Embedding CFR operates per info-block, requiring an embedding matrix Φ for each—a labor-intensive process. If partitions allowed matrices to be constructed for a subset of info-blocks, with others extendable from these, workload would drop markedly.

Poker games possess a key property that facilitates info-block partitioning while addressing both challenges mentioned above: history indistinguishability within an infoset stems from chance actions (e.g., dealing), while non-chance actions (e.g., betting) are public and preserve strategic distinctness. Critically, all histories in the same infoset share an identical non-chance action trace (a result of **perfect recall** in poker). This formalizes our partitioning rule: info-blocks in poker games are defined as groups of infosets whose contained histories share the same non-chance action trace.

Consider a concrete example from HULHE, where a game history h can be decomposed into interleaved chance (card dealing) and non-chance (betting) segments:

$$h = \underbrace{8_h^1 8_d^1 J_s^2 Q_s^2}_{\text{Chance: Private Cards}} \cdot \overbrace{\text{rRC}}^{\text{Non-chance Actions}} \cdot \underbrace{2_s J_s 8_h}_{\text{Chance: Community Cards}} \cdot \overbrace{\text{Cr}}^{\text{Non-chance Actions}} \in I \in \mathcal{I}_2,$$

³

For two additional histories h' and h'' :

³c/r/f (Player 1, small blind) and C/R/F (Player 2, big blind) denote check, raise, and fold, respectively.

- $h' = 7_h^1 7_d^1 J_s^2 Q_s^2 \cdot \text{rRC} \cdot 2_s J_s 8_h \cdot \text{Cr} \in I$,
- $h'' = 7_h^1 5_d^1 8_s^2 4_s^2 \cdot \text{rRC} \cdot J_s Q_s 3_h \cdot \text{Cr} \in I' \neq I$,

all three histories (and their containing infosets I, I') belong to the same info-block J by our partitioning rule, as they share the identical non-chance trace: $\mathcal{H}_{-c}(h) = \mathcal{H}_{-c}(h') = \mathcal{H}_{-c}(h'') = \emptyset \cdot \text{rRC} \cdot \emptyset \cdot \text{Cr}$, where $\mathcal{H}_{-c}(\cdot)$ denotes the operator that retains only non-chance actions (replacing chance segments with \emptyset).

Furthermore, this partitioning rule directly addresses the challenge of constructing embedding matrices for each info-block. For example, consider histories h, h', h'' with the same chance traces as h, h', h'' but a different non-chance trace ($\emptyset \cdot \text{rRC} \cdot \emptyset \cdot \text{C}$)—the infosets to which these histories belong come from another info-block J :

$$\begin{aligned} h &= 8_h^1 8_d^1 J_s^2 Q_s^2 \cdot \text{rRC} \cdot 2_s J_s 8_h \cdot \text{C} \in I \in J, \\ h' &= 7_h^1 7_d^1 J_s^2 Q_s^2 \cdot \text{rRC} \cdot 2_s J_s 8_h \cdot \text{C} \in I \in J, \\ h'' &= 7_h^1 5_d^1 8_s^2 4_s^2 \cdot \text{rRC} \cdot J_s Q_s 3_h \cdot \text{C} \in I' \in J. \end{aligned}$$

Notably, J and J are in one-to-one correspondence: their infosets (e.g., $I \leftrightarrow \dot{I}, I' \leftrightarrow \dot{I}'$) are counterparts linked by identical chance traces. Since history indistinguishability within poker infosets stems solely from such chance actions (not non-chance actions), we construct embedding matrix Φ based on chance actions (i.e., card dealing)—details in Section 4.2. Given that J and J share this chance-based structure via their corresponding infosets, Φ built for J can be directly reused for J , substantially reducing construction workload.

4.2 Embedding Matrices: Construction and Deployment in Poker Games

We now discuss the construction of embedding matrices in poker games and their deployment in Embedding CFR iterations. As established in Section 4.1, these matrices are grounded in chance actions—specifically, players’ hands in poker games.

Poker hands exhibit a comparative structure (not strictly a total order, as not all hands are directly comparable). In the final betting round, hands form a total order enabling definitive comparisons; in earlier rounds, their relative strength is inferred from outcome distributions when rolled out to the final round (simulating remaining card deals yields probabilistic performance measures). This comparative strength captures both distinctions and connections between infosets: hands with similar strength profiles share strong strategic ties, while divergent profiles indicate distinctiveness—forming the basis of our embedding matrix construction.

However, storing a complete Φ would require $m \times n$ space, which is infeasible given poker’s massive hand scale ($n \approx 5 \times 10^{10}$ in Texas Hold’em). Thus, instead of constructing Φ in full, we aim to generate the embedding coordinates $\Phi_{\cdot, q}$ on-the-fly for each infoset I_q .

Neural networks are well-suited for the aforementioned supervised learning and dynamic embedding generation scenarios. To this end, we propose HandEbdNet, an embedding network illustrated in Figure 4. It takes as input a hand



Figure 4: The network architecture of HandEbdNet.

tensor x_q —a structured encoding of a hand’s suits, ranks, and round-wise presence (corresponding to $I_q \in J$)—and outputs a strength tensor y_q capturing comparative strength (e.g., win-rate distributions across rounds).

Functionally, HandEbdNet comprises three core components: (1) a hand feature encoder (HFE)—a composite module of multiple networks—that maps x_q to an m -dimensional intermediate feature vector; (2) a softmax layer transforming this vector into an m -dimensional probability distribution, viewed as $\Phi_{\cdot,q}$; and (3) a final fully connected layer (FC) that transforms this distribution, with supervision from y_q forming the learning objective.

The core insight is using y_q to induce discriminative $\Phi_{\cdot,q}$: hands with similar strength characteristics (reflected in y_q) yield proximate $\Phi_{\cdot,q}$, capturing meaningful structural relationships between infosets in the advisor space. In poker scenarios, traversing all possible hands (rather than storing them) is feasible, and no entirely new hands will be encountered during embedding coordinate generation. Thus, HandEbdNet is not prone to overfitting in both training and deployment. Details on constructing x_q , y_q , and HandEbdNet’s full architecture are provided in Appendix D.

5 Experiment

We conducted experiments in the Numeral211 Hold’em environment, a simplified variant of Texas Hold’em poker proposed by Fu et al. (2025). This game uses a 40-card deck and includes 3 betting rounds, retaining sufficient complexity—particularly in hand diversity—while reducing the overall game scale. This makes it an ideal testbed for researching information set abstraction. Detailed rules are provided in Appendix A.3.

We primarily compare strategies generated by (extended) information set abstraction methods—including the classical EHS, PaEmd (an algorithm successfully applied in DeepStack and Libratus), and the novel KrwEmd proposed by Fu et al. (2025)—when paired with Vanilla CFR for solving, alongside our Embedding CFR, which simultaneously handles information set abstraction and strategy solving.

To ensure a fair comparison, all algorithms are maintained at the logical spatial resources (in terms of the number of abstracted information sets or the size of the advisor space). For the Numeral211 Hold’em environment, the number of hand combinations across its three betting rounds is $\binom{40}{2}$, $\binom{40}{2} \times \binom{38}{1}$, and $\binom{40}{2} \times \binom{38}{1} \times \binom{37}{1}$ respectively. The number of structurally equivalent hand strength classes (via lossless isomorphism (Gilpin and Sandholm 2007b)) across the three rounds is 100, 2250, and 3957, as determined by Waugh (2013). Regarding the logical spatial resources used by abstraction algorithms and Embedding CFR, they are restricted to 225 and 396 in betting rounds 2 and 3 respectively. No

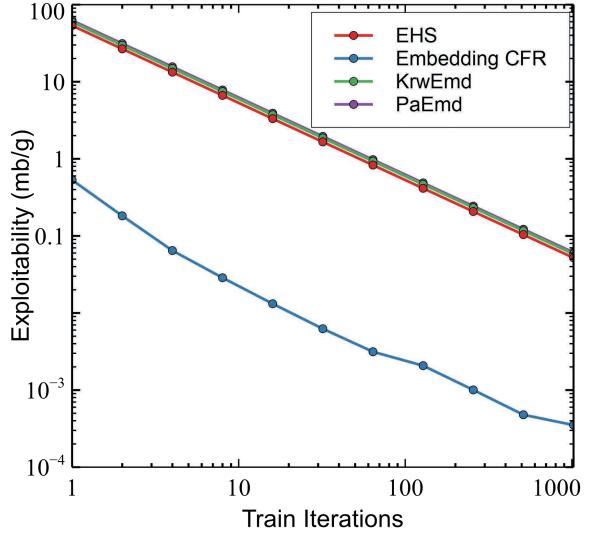


Figure 5: Exploitability convergence comparison of clustering-based algorithm (EHS, PaEmd, KrwEmd) vs. Embedding CFR algorithms in Numeral211 Hold’em.

tably, all algorithms do not apply abstraction or embedding in the first round; instead, they rely on lossless isomorphism.

We compare the convergence of exploitability among these algorithms, and the experimental results are presented in Figure 5. For the experiment, we sampled a number of hands equivalent to all possible hand combinations in a game, which is $\binom{40}{2} \times \binom{38}{1} \times \binom{37}{1}$ (approximately 10^6) as one iteration, and a total of 1024 iterations are conducted to compare the exploitability in various scenarios. The unit of exploitability is milli blind per game (mb/g). For each baseline comparison algorithm, we provide three parameter configurations. The figure shows the optimal experimental curve of the algorithm. More extensive experiments and details of the experimental equipment are provided in Appendix E.

It can be observed that the performance of abstraction methods based on hand clustering (EHS, PaEmd, KrwEmd) varies slightly, but they are at a comparable level. In contrast, the Embedding CFR algorithm demonstrates a significant performance boost in terms of exploitability reduction, with the quality of its strategy being notably superior to that of clustering-based algorithms. This result effectively validates the effectiveness of the Embedding CFR algorithm in the context of information set abstraction for poker games.

6 Conclusion

In conclusion, this paper presents the Embedding CFR algorithm, which innovatively introduces the embedding concept to address the pre-training process of information set abstraction in imperfect information games. This approach marks a significant breakthrough in the field, as it comprehensively outperforms clustering-based methods in our experimental settings.

Acknowledgments

This work was supported by the National Science and Technology Major Project (Grant No. 2022ZD0116403) and the China Postdoctoral Science Foundation (Grant No. 2024M763533).

References

- Bowling, M.; Burch, N.; Johanson, M.; and Tammelin, O. 2015. Heads-up limit hold'em poker is solved. *Science*, 347(6218): 145–149.
- Brown, N.; Bakhtin, A.; Lerer, A.; and Gong, Q. 2020. Combining deep reinforcement learning and search for imperfect-information games. In *In Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020)*, 17057–17069.
- Brown, N.; Kroer, C.; and Sandholm, T. 2017. Dynamic thresholding and pruning for regret minimization. In *In Proceedings of the 31st AAAI Conference On Artificial Intelligence (AAAI-17)*, 421–429.
- Brown, N.; Lerer, A.; Gross, S.; and Sandholm, T. 2019. Deep counterfactual regret minimization. In *In Proceedings of the The 36th International Conference on Machine Learning (ICML 2019)*, 793–802.
- Brown, N.; and Sandholm, T. 2015. Regret-based pruning in extensive-form games. In *In Proceedings of the 29th International Conference on Neural Information Processing Systems (NeurIPS 2015)*.
- Brown, N.; and Sandholm, T. 2017. Reduced space and faster convergence in imperfect-information games via pruning. In *In Proceedings of the The 34th International Conference on Machine Learning (ICML 2017)*, 596–604.
- 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. 2019a. Solving imperfect-information games via discounted regret minimization. In *In Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-19)*, 1829–1836.
- Brown, N.; and Sandholm, T. 2019b. Superhuman AI for multiplayer poker. *Science*, 365(6456): 885–890.
- Farina, G.; Kroer, C.; and Sandholm, T. 2021. Faster game solving via predictive blackwell approachability: Connecting regret matching and mirror descent. In *In Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-21)*, 6, 5363–5371.
- Fu, Y.; Yin, Q.; Liu, S.; Xu, P.; and Huang, K. 2025. KrwEmd: Revising the Imperfect-Recall Abstraction from Forgetting Everything. arXiv:2511.12089.
- Ganzfried, S.; and Sandholm, T. 2014. Potential-aware imperfect-recall abstraction with earth mover's distance in imperfect-information games. In *In Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI-14)*, 682–690.
- Gilpin, A.; and Sandholm, T. 2007a. Better automated abstraction techniques for imperfect information games, with application to Texas Hold'em poker. In *In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, 1–8.
- Gilpin, A.; and Sandholm, T. 2007b. Lossless abstraction of imperfect information games. *Journal of the ACM (JACM)*, 54(5): 25–es.
- Gilpin, A.; Sandholm, T.; and Sørensen, T. B. 2007. Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker. In *In Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI-07)*, 50–57.
- Hart, S.; and Mas-Colell, A. 2000. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5): 1127–1150.
- Johanson, M. 2013. Measuring the size of large no-limit poker games. *arXiv preprint arXiv:1302.7008*.
- Lanctot, M.; Waugh, K.; Zinkevich, M.; and Bowling, M. 2009. Monte Carlo sampling for regret minimization in extensive games. In *In Proceedings of the 23rd International Conference on Neural Information Processing Systems (NIPS 2009)*, 1078–1086.
- Li, B.; Fang, Z.; and Huang, L. 2024. RL-CFR: improving action abstraction for imperfect information extensive-form games with reinforcement learning. In *In Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, 27752–27770.
- Li, B.; and Huang, L. 2025. Efficient Online Pruning and Abstraction for Imperfect Information Extensive-Form Games. In *In Proceedings of 13th International Conference on Learning Representations (ICLR 2025)*.
- McAleer, S. M.; Farina, G.; Lanctot, M.; and Sandholm, T. 2023. ESCHER: Eschewing Importance Sampling in Games by Computing a History Value Function to Estimate Regret. In *In Proceedings of the 11th International Conference on Learning Representations (ICLR 2023)*.
- Moravčík, M.; Schmid, M.; Burch, N.; Lisý, V.; Morrill, D.; Bard, N.; Davis, T.; Waugh, K.; Johanson, M.; and Bowling, M. 2017. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337): 508–513.
- Waugh, K. 2009. Abstraction in large extensive games.
- Waugh, K. 2013. A Fast and Optimal Hand Isomorphism Algorithm. In *The Second Computer Poker and Imperfect Information Symposium, AAAI*.
- Zinkevich, M.; Johanson, M.; Bowling, M.; and Piccione, C. 2007. Regret minimization in games with incomplete information. In *In Proceedings of the 21st International Conference on Neural Information Processing Systems (NIPS 2007)*, 1729–1736.

A Rules of Referenced Imperfect Information Games

A.1 Heads-Up Texas Hold'em

Heads-Up Texas Hold'em is a two-player variant of Texas Hold'em, retaining the core mechanics of community cards and hand rankings while adapting to two-player dynamics. Below outlines the shared rules between its limit and no-limit variants, followed by their distinct betting structures.

1. **Betting Rounds:** Consists of four rounds—preflop, flop, turn, and river—where players can act sequentially after the corresponding cards are dealt.
2. **Deck & Hand Composition:**
 - Employs a standard 52-card deck (excluding Jokers), with each player privately dealt two private cards preflop.
 - Five community cards are revealed sequentially: three cards (the flop) are dealt first, followed by the turn (fourth card) and river (fifth card), forming the strongest possible 5-card hand in combination with the player's private cards.
3. **Hand Rankings:** Follow standard poker hierarchies, from Straight Flush (highest) to High Card (lowest), as detailed in Table 1.
4. **Blinds:** Small Blind (SB) and Big Blind (BB) initiate the pot, with the SB posting half the BB amount.
5. **Action Order:** Preflop action starts with the SB; postflop (flop, turn, river) action starts with the BB.
6. **AI Research Standard Setup:** Commonly used in academic studies, the SB is set to 100 chips, with each player starting with a 20,000-chip stack (200 BB effective stack depth).

Rank	Hand	Prob.	Description	Example
1	Straight Flush	0.0015%	Five cards of consecutive rank, all of the same suit. Ties broken by highest card in the sequence.	$A_s K_s Q_s J_s T_s$
2	Four of a Kind	0.0240%	Four cards of the same rank. Ties broken by the rank of the four cards.	$8_s 8_h 8_d 8_c K_h$
3	Full House	0.1441%	Three of a kind plus a pair. Ties broken by the rank of the three of a kind, then the pair.	$Q_s Q_h Q_d 5_c 5_s$
4	Flush	0.1965%	Five cards of the same suit, not in sequence. Ties broken by comparing the highest card, then the next highest, etc.	$9_h 7_h 6_h 4_h 2_h$
5	Straight	0.3925%	Five cards of consecutive rank, not all the same suit. Ties broken by highest card in the sequence.	$K_s Q_h J_c T_d 9_h$
6	Three of a Kind	2.1128%	Three cards of the same rank. Ties broken by the rank of the three, then the highest remaining card, then the second remaining card.	$7_s 7_h 7_d K_c 4_h$
7	Two Pair	4.7539%	Two different pairs. Ties broken by the rank of the higher pair, then the lower pair, then the remaining card.	$J_s J_h 5_c 5_s 2_h$
8	One Pair	42.2569%	Two cards of the same rank. Ties broken by the rank of the pair, then the highest remaining card, then the next highest, and so on.	$9_s 9_h A_c K_s 3_h$
9	High Card	50.1177%	No pairs or better. Ties broken by comparing the highest card, then the next highest, etc.	$A_s K_h J_c 8_d 3_h$

Table 1: Hand ranks of Texas Hold'em

Heads-Up Limit Texas Hold'em (HULHE) The limit variant imposes fixed constraints on bet sizing and raise frequency, distinguishing it from no-limit play:

1. **Bet Sizing:** - Preflop bets and raises are capped at the big blind amount (e.g., equivalent to the BB value). - Postflop (flop, turn, river) bets and raises double to twice the big blind, maintaining consistent increments.

2. **Raise Limits:** Each betting round allows a maximum of one initial bet plus three raises (e.g., bet → raise → re-raise → cap raise), preventing unbounded betting sequences.

Heads-Up No-Limit Texas Hold'em (HUNL) The no-limit variant removes fixed bet constraints, enabling flexible sizing and strategic depth:

1. **Bet Sizing:** Players may bet any amount from their remaining stack (*ge* the big blind for preflop opens), with raises requiring a minimum size equal to the previous bet.
2. **Raise Flexibility:** No cap on the number of raises per round, allowing complex sequences (e.g., 3-bets, 4-bets) as long as each raise meets the minimum size requirement.
3. **All-In Mechanic:** A player can bet their entire stack at any round, locking the hand until showdown if called.

A.2 Kuhn Poker

Kuhn Poker is a simplified two-player poker variant introduced by Kuhn in 1950, serving as a fundamental model for studying imperfect information games in game theory and artificial intelligence research. Despite its simplicity, it captures key strategic elements such as bluffing and information asymmetry. The game proceeds as follows:

1. **Ante:** Each player antes 1 chip into the pot.
2. **Deck:** The deck consists of three distinct cards: a Jack (J), a Queen (Q), and a King (K).
3. **Private Card:** Each player is dealt one private card face down.
4. **Betting Round:** The player who acts first (determined by a fixed order, typically alternating between hands) can choose to either check or bet 1 chip. The second player responds as follows:
 - If the first player checked, the second player may check (ending the hand with a showdown) or bet 1 chip. If the second player bets, the first player must either call (leading to a showdown) or fold (conceding the pot to the second player).
 - If the first player bet, the second player must either call (leading to a showdown) or fold (conceding the pot to the first player).
5. **Showdown:** A showdown occurs if both players check, or if a bet is followed by a call. The player with the highest-ranked card (as specified in Table 2) wins the pot.

Rank	Hand	Probability
1	K	1/3
2	Q	1/3
3	J	1/3

Table 2: Hand ranks of Kuhn Poker

A.3 Numeral211 Hold'em

Numeral211 Hold'em is a poker variant more complex than Kuhn Poker but significantly simpler than HULHE. With diverse hand possibilities, it serves as an ideal testbed for studying hand abstraction tasks (Fu et al. 2025). The game proceeds as follows:

1. **Ante:** Each player antes 5 chips into the pot at the start of the hand.
2. **Betting Rounds:** Consists of three rounds—preflop, flop, turn—where players can act sequentially after the corresponding cards are dealt.
3. **Deck & Hand Composition:**
 - Derived from a standard 54-card deck (excluding Jokers, Kings, Queens, and Jacks), resulting in 40 cards total. Suits include spades (s), hearts (h), clubs (c), and diamonds (d), with ranks 2 through 9, 10 (T), and Ace (A).
 - Each player receives one private card preflop. Two community cards are revealed sequentially, with one card in flop and one card in turn, respectively. The strongest 3-card hand is formed using the private card and community cards.
4. **Hand Rankings:** Follow hierarchies from Straight Flush (highest) to High Card (lowest), as detailed in Table 3.
5. **Betting Options:** Players may fold, check, call, bet, or raise in all rounds (similar to HULHE). Each round allows a maximum of four total bets.raises, with fixed sizes: 10 chips in preflop and 20 chips in postflop rounds (flop and turn).
6. **Showdown:** If neither player folds, the highest-ranked hand wins the pot; ties result in an even split.

Rank	Hand	Prob.	Description	Example
1	Straight flush	0.321%	Three cards with consecutive rank and same suit. Ties are broken by highest card.	$T_s 9_s 8_s$
2	Three of a kind	1.587%	Three cards with the same rank. Ties are broken by the card's rank.	$T_s T_h T_c$
3	Straight	4.347%	Three cards with consecutive rank. Ties are broken by the highest card rank.	$T_s 9_h 8_c$
4	Flush	15.799%	Three cards with the same suit. Ties are broken by the highest card rank, then second highest, then third highest.	$T_s 8_s 6_s$
5	Pair	34.065%	Two cards with the same rank. Ties are broken by the pair's rank, then the third card's rank.	$T_s T_h 8_c$
6	High card	43.881%	None of the above. Ties are broken by comparing the highest card, then second highest, then third highest.	$T_s 8_h 6_c$

Table 3: Hand ranks of Numeral211 Hold’em

B Pseudocode for Poker-Specific Embedding CFR

Algorithm 1 presents the pseudocode for Poker-Specific Embedding CFR (e.g., tailored for Texas Hold’em), which leverages the unique properties of poker-like games as described in Section 4.1. Specifically, the progression of such games can be partitioned into a sequence of public tree nodes, which depict the game process identifiable to an observer. These nodes correspond to key game events, such as the chance player dealing community cards or other players taking actions like calling, betting, or folding. We denote the set of these public tree nodes as V .

All stochastic factors in the game can be determined prior to traversing the entire public tree—these factors refer to the private cards dealt to players and the community cards to be revealed by the final showdown stage. Traversing the public tree with these pre-determined cards is referred to as chance sampling.

Each hand of cards, when combined with a given public tree node v , uniquely defines an information set. It is straightforward to confirm that there exists a one-to-one correspondence between each public tree node v and an info-block J . We use $J(v)$ to represent the info-block determined by node v .

C Theoretical Proofs

Lemma 1 ((Lanctot et al. 2009), Lemma 7). For any real numbers a and b , define the positive truncation operator $a_+ \triangleq \max(a, 0)$. The following inequality holds:

$$[(a + b)_+]^2 \leq (a_+)^2 + 2(a_+)b + b^2.$$

Proof. We analyze three exhaustive cases based on the signs of a and $a + b$:

Case 1: $a \leq 0$ Since $a_+ = 0$, the right-hand side simplifies to b^2 . Meanwhile:

$$[(a + b)_+]^2 \leq (b_+)^2 \leq b^2,$$

confirming the inequality.

Case 2: $a \geq 0$ and $b \geq -a$ Here, $a_+ = a$ and $(a + b)_+ = a + b$. Expanding both sides:

$$[(a + b)_+]^2 = (a + b)^2 = a^2 + 2ab + b^2,$$

which matches the right-hand side $(a_+)^2 + 2(a_+)b + b^2$.

Case 3: $a \geq 0$ and $b \leq -a$ We have $(a + b)_+ = 0$, and the right-hand side:

$$(a_+)^2 + 2(a_+)b + b^2 = [(a_+) + b]^2 \geq 0 = [(a + b)_+]^2,$$

satisfying the inequality.

Since all cases hold, the lemma is proven. □

Lemma 2. For any info-set $I \in \mathcal{I}_i$ and strategy profile σ , the counterfactual value of player i at I satisfies:

$$v_i(\sigma, I) = \sum_{a \in A(I)} \sigma_i(I, a) \cdot v_i^{\sigma|I \rightarrow a}(I)$$

Algorithm 1: Embedding CFR algorithm for two-player hold'em games

Require: $\tilde{R}_{\text{advisor}}(\cdot)$: A mapping from a public node $v \in V$ to an embedded cumulative regret tensor of size $|A(J(v))| \times m$

Require: $\sigma_{\text{advisor}}(v)$: A mapping from a public node $v \in V$ to an embedded immediate strategy tensor of size $|A(J(v))| \times m$

Require: $\bar{\sigma}_{\text{advisor}}(v)$: A mapping from a public node $v \in V$ to an embedded average strategy tensor of size $|A(J(v))| \times m$

- 1: **for** $t \leftarrow 1$ **to** n **do**
- 2: Deal l batches of final private card tensors $\mathbf{h}_1, \mathbf{h}_2$ (for players 1 and player 2, respectively) and a final community card tensor \mathbf{b}
- 3: $\text{TRAVERSE}(v^0, \mathbf{h}_1, \mathbf{h}_2, \mathbf{b}, \mathbf{1}_l, \mathbf{1}_l)$ ▷ v^0 : public tree root; $\mathbf{1}_l$: a l -dimensional all-ones vector
- 4: **end for**
- 5: **function** $\text{TRAVERSE}(v, \mathbf{h}_1, \mathbf{h}_2, \mathbf{b}, \mathbf{p}_1, \mathbf{p}_2)$
- 6: **if** v is terminal public node **then**
- 7: $[\mathbf{u}_1, \mathbf{u}_2] \leftarrow \text{GETUTILITY}(v, \mathbf{h}_1, \mathbf{h}_2, \mathbf{b})$ ▷ GETUTILITY: Compute utilities from a public node and card tensors
- 8: **return** $[\mathbf{p}_2 \circ \mathbf{u}_1, \mathbf{p}_1 \circ \mathbf{u}_2]$ ▷ “ \circ ” denotes element-wise multiplication
- 9: **end if**
- 10: $\mathbf{r}_1 \leftarrow \mathbf{0}_l; \mathbf{r}_2 \leftarrow \mathbf{0}_l; \mathbf{R} \leftarrow \mathbf{0}_m$ ▷ zero vectors of length l or m
- 11: **for all** $i \in \{1, 2\}$ **do**
- 12: **if** v is player i 's public node **then**
- 13: $\mathbf{x}_i \leftarrow \text{HANDTENSOR}(v, \mathbf{h}_i, \mathbf{b})$ ▷ HANDTENSOR: Truncate card tensors to v 's betting round; construct l -batch hand tensors (see Appendix D.1)
- 14: $[\Phi_{\cdot, \{j_1, \dots, j_l\}}, \cdot] \leftarrow \text{HANDEBDNET}(\mathbf{x}_i)$
- 15: **for all** $a \in A(J(v))$ **do**
- 16: $\sigma_{\text{original}} \leftarrow (\Phi_{\cdot, \{j_1, \dots, j_l\}})^{\top} \cdot \sigma_{\text{advisor}}(v)[a]$
- 17: $\mathbf{p}'_i = \mathbf{p}_i \circ \sigma_{\text{original}}; \mathbf{p}'_{3-i} = \mathbf{p}_{3-i}$
- 18: $[\mathbf{r}'_1, \mathbf{r}'_2] \leftarrow \text{TRAVERSE}(\text{NEXTNODE}(v, a), \mathbf{h}_1, \mathbf{h}_2, \mathbf{b}, \mathbf{p}'_1, \mathbf{p}'_2)$ ▷ NEXTNODE: Next public node after executing action a at v
- 19: $\mathbf{r}_i \leftarrow \mathbf{r}_i + \mathbf{r}'_i \circ \sigma_{\text{original}}; \mathbf{r}_{3-i} \leftarrow \mathbf{r}_{3-i} + \mathbf{r}'_{3-i}$
- 20: $\tilde{R}_{\text{advisor}}(v)[a] \leftarrow \frac{T-1}{T} \tilde{R}_{\text{advisor}}(v)[a] + \frac{1}{T} \Phi_{\cdot, \{j_1, \dots, j_l\}} \cdot (\mathbf{r}'_i \circ \sigma_{\text{original}})$
- 21: $\mathbf{R} \leftarrow \mathbf{R} + \text{TENSORCLAMP}(\tilde{R}_{\text{advisor}}(v)[a], \varepsilon, \infty)$ ▷ TENSORCLAMP: values $< \varepsilon \rightarrow \varepsilon$, otherwise unchanged
- 22: **end for**
- 23: **for all** $a \in A(J(v))$ **do**
- 24: $\sigma_{\text{advisor}}(v)[a] \leftarrow \text{TENSORCLAMP}(\tilde{R}_{\text{advisor}}(v)[a], \varepsilon, \infty) \setminus \mathbf{R}$ ▷ “\” denotes element-wise division
- 25: $\bar{\sigma}_{\text{advisor}}(v)[a] \leftarrow \frac{T-1}{T} \bar{\sigma}_{\text{advisor}}(v)[a] + \frac{1}{T} \sigma_{\text{advisor}}(v)[a]$
- 26: **end for**
- 27: **end if**
- 28: **end for**
- 29: **return** $[\mathbf{r}_1, \mathbf{r}_2]$
- 30: **end function**
- 31: **function** $\text{HANDEBDNET}([\mathbf{x}_q]_{q \in Q})$ ▷ $[\mathbf{x}_q]_{q \in Q}$: $|Q|$ -batch hand tensor; \mathbf{x}_q as described in Section 4.2
- 32: **return** $[[\Phi_{\cdot, q}]_{q \in Q}, [\mathbf{y}_q]_{q \in Q}]$ ▷ $\Phi_{\cdot, q}$ and \mathbf{y}_q as described in Section 4.2
- 33: **end function**

Proof. By definition, the counterfactual value is:

$$v_i^\sigma(I) = \sum_{z \in Z_I} \pi_{-i}^\sigma(z[I]) \cdot \pi^\sigma(z[I], z) \cdot u_i(z)$$

Partition Z_I by actions at I : $Z_I = \bigcup_{a \in A(I)} Z_I^a$, where Z_I^a contains terminal histories passing through I via action a . This allows rewriting the summation as:

$$v_i^\sigma(I) = \sum_{a \in A(I)} \sum_{z \in Z_I^a} \pi_{-i}^\sigma(z[I]) \cdot \pi^\sigma(z[I], z) \cdot u_i(z)$$

For $z \in Z_I^a$, decompose the transition probability: $\pi^\sigma(z[I], z) = \sigma_i(I, a) \cdot \pi^\sigma(z[I] \cdot a, z)$. This decomposition holds because $z[I] \in I$, and I is an info-set where player i acts. Substituting this gives:

$$v_i^\sigma(I) = \sum_{a \in A(I)} \sigma_i(I, a) \cdot \sum_{z \in Z_I^a} \pi_{-i}^\sigma(z[I]) \cdot \pi^\sigma(z[I] \cdot a, z) \cdot u_i(z)$$

For the modified strategy $\sigma|_{I \rightarrow a}$ (which fixes action a at I), note that $\pi^{\sigma|_{I \rightarrow a}}(z[I], z) = \pi^\sigma(z[I] \cdot a, z)$ for all $z \in Z_I^a$. By the definition of counterfactual value under $\sigma|_{I \rightarrow a}$:

$$v_i^{\sigma|_{I \rightarrow a}}(I) = \sum_{z \in Z_I^a} \pi_{-i}^\sigma(z[I]) \cdot \pi^{\sigma|_{I \rightarrow a}}(z[I], z) \cdot u_i(z) = \sum_{z \in Z_I^a} \pi_{-i}^\sigma(z[I]) \cdot \pi^\sigma(z[I] \cdot a, z) \cdot u_i(z)$$

Substituting this equality for the inner summation yields:

$$v_i^\sigma(I) = \sum_{a \in A(I)} \sigma_i(I, a) \cdot v_i^{\sigma|_{I \rightarrow a}}(I)$$

This completes the proof. \square

Proof of Proposition 1. In the proposition's description of the approximate sampling scenario for Embedding CFR, consider iteration T where player i happens to select the strategy e_p for interaction across all information sets $I \in J$.

Key Identity: Cross-Term Vanishing

We first note that under this scenario, the following holds:

$$\sum_{a \in A(J)} (R^T(e_p, a)_+) r^{T+1}(e_p, a) = 0. \quad (9)$$

If $\sum_{a \in A(J)} (R^T(e_p, a)_+) = 0$, the conclusion holds trivially. This is because $(R^T(e_p, a)_+)$ is non-negative, and a sum of non-negative terms equals zero only if each term individually is zero.

Then, when $\sum_{a \in A(J)} (R^T(e_p, a)_+) > 0$:

$$\begin{aligned} & \sum_{a \in A(J)} (R^T(e_p, a)_+) r^{T+1}(e_p, a) \\ &= \sum_{a \in A(J)} (R^T(e_p, a)_+) \sum_{q=1}^n \Phi_{p,q} r^{T+1}(I_q, a) \\ &= \sum_{a \in A(J)} (R^T(e_p, a)_+) \sum_{q=1}^n \Phi_{p,q} \left(v_i^{\sigma^T}(I_q, a) - v_i^{\sigma^T}(I_q) \right) \\ &= \sum_{a \in A(J)} (R^T(e_p, a)_+) \sum_{q=1}^n \Phi_{p,q} \left(v_i^{\sigma^T}(I_q, a) - \sum_{a' \in A(J)} \sigma^T(I_q, a') v_i^{\sigma^T}(I_q, a') \right) \text{ [By Lemma 2]} \\ &= \sum_{a \in A(J)} (R^T(e_p, a)_+) \sum_{q=1}^n \Phi_{p,q} \left(v_i^{\sigma^T}(I_q, a) - \sum_{a' \in A(J)} \sigma^T(e_p, a') v_i^{\sigma^T}(I_q, a') \right) \text{ [Since all info-sets } I_q \in J \text{ use advisor } e_p \text{'s strategy]} \\ &= \sum_{a \in A(J)} (R^T(e_p, a)_+) \sum_{q=1}^n \Phi_{p,q} \left(v_i^{\sigma^T}(I_q, a) - \sum_{a' \in A(J)} \frac{R^T(e_p, a')_+}{\sum_{a'' \in A(J)} (R^T(e_p, a'')_+)} v_i^{\sigma^T}(I_q, a') \right) \text{ [By regret matching (Equation (6))]} \\ &= \sum_{a \in A(J)} \sum_{q=1}^n \Phi_{p,q} (R^T(e_p, a)_+) v_i^{\sigma^T}(I_q, a) - \sum_{q=1}^n \sum_{a' \in A(J)} \Phi_{p,q} (R^T(e_p, a')_+) v_i^{\sigma^T}(I_q, a') \\ &= 0 \end{aligned}$$

Bounding the Instantaneous Regret

To characterize the regret reduction, we then bound $(r^{T+1}(e_p, a))^2$ using the Cauchy-Schwarz inequality:

$$\begin{aligned}
(r^\sigma(e_p, a))^2 &= (\Phi_{p,\cdot} r^\sigma(\mathbf{J}, a))^2 \\
&\leq \|\Phi_{p,\cdot}\|_2^2 \cdot \|r^\sigma(\mathbf{J}, a)\|_2^2 \quad (\text{by Cauchy-Schwarz inequality}) \\
&= \|\Phi_{p,\cdot}\|_2^2 \sum_{q=1}^n (v^{\sigma|I_q \rightarrow a}(I_q) - v^\sigma(I_q))^2 \\
&\leq \|\Phi_{p,\cdot}\|_2^2 \cdot \sum_{q=1}^n \Delta_J^2 \quad (\text{since } |v^{\sigma|I_q \rightarrow a}(I_q) - v^\sigma(I_q)| \leq \Delta_J \text{ for all } q) \\
&= n\Delta_J^2 \cdot \|\Phi_{p,\cdot}\|_2^2.
\end{aligned} \tag{10}$$

Bounding the Instantaneous Regret

Define $S^T = \sum_{a \in A(J)} (R^T(e_p, a)_+)^2$. We have:

$$\begin{aligned}
S^{T+1} &= \sum_{a \in A(J)} (R^{T+1}(e_p, a)_+)^2 \\
&= \sum_{a \in A(J)} \left[\left(\frac{T}{T+1} R^T(e_p, a) + \frac{1}{T+1} r^{T+1}(e_p, a) \right)_+ \right]^2 \\
&\leq \frac{T^2}{(T+1)^2} \sum_{a \in A(J)} (R^T(e_p, a)_+)^2 + \frac{1}{T^2} \sum_{a \in A(J)} (r^{T+1}(e_p, a))^2 + \frac{2T}{(T+1)^2} \overbrace{\sum_{a \in A(J)} (R^T(e_p, a)_+) r^{T+1}(e_p, a)}^{\text{equals zero by Equation (9)}} \quad (\text{by Lemma 1}) \\
&= \frac{T^2}{(T+1)^2} S^T + \frac{1}{(T+1)^2} \sum_{a \in A(J)} (r^{T+1}(e_p, a))^2.
\end{aligned} \tag{11}$$

By Equation (10), $\sum_{a \in A(J)} (r^{T+1}(e_p, a))^2 \leq |A(J)| \cdot n\Delta_J^2 \cdot \|\Phi_{p,\cdot}\|_2^2 \triangleq C$, where C is a constant. Inequality (11) simplifies to:

$$S^{T+1} \leq \frac{T^2}{(T+1)^2} \cdot S^T + \frac{C}{(T+1)^2} \tag{12}$$

We now analyze two scenarios for S^T :

1. Small S^T : Controlled Polynomial Decay

If $S^T \leq \frac{C}{T}$, substituting into the inequality gives:

$$S^{T+1} \leq \frac{T^2}{(T+1)^2} \cdot \frac{C}{T} + \frac{C}{(T+1)^2} = \frac{T \cdot C + C}{(T+1)^2} = \frac{C}{T+1}.$$

This shows small S^T propagate to $S^{T+1} \leq \frac{C}{T+1}$, with decay strictly controlled by the bound $O(\frac{1}{T})$.

2. Large S^T : Uncontrolled Polynomial Decay (Fractional Scaling)

If $S^T > \frac{C}{T}$, the dominant term drives a faster reduction:

$$\begin{aligned}
S^{T+1} &\leq \frac{T^2}{(T+1)^2} S^T + \frac{C}{(T+1)^2} \\
&\leq \frac{T^2}{(T+1)^2} S^T + \frac{T \cdot S^T}{(T+1)^2} \\
&= \frac{T}{T+1} S^T
\end{aligned}$$

This means that when $S^T > \frac{C}{T}$ holds, no explicit bound can be established for S^{T+1} . Even so, it still guarantees a regret-decreasing process. Specifically, the magnitude of the decrease is proportional to S^T itself, yet decays with iteration. \square

D Detailed Implementation of HandEbdNet in Numeral211 Hold'em

D.1 The Construction of the Hand Tensor \mathbf{x}_q

Numeral211 Hold'em, a 3-round poker game with 4 suits and 10 ranks, represents hands in the s -th round as $[4, 10, s]$ image-like tensors (see Figure 6). Each channel is a $10 \times s$ map encoding one suit's cards across rounds, facilitating independent analysis of straight flushes and flushes.

Preprocessing ensures isomorphic hands share identical tensors. Channels are ordered by suit importance: first by card count (more means a higher importance), with lexicographical order breaking ties (e.g., a suit containing an Ace in the first round and a 9 in the second round is considered more important than one with a 10 in the first round and a 0 in the second round). This uniquely distinguishes all suits in Numeral211 Hold'em.

The Figure 6 illustrates examples of such tensor construction. Specifically, the hands $[A_c^p T_c^p, 9_d, 2_c]$ and $[A_h^p T_h^p, 9_s, 2_h]$ share an identical tensor representation, and their predecessors in the second round are also identical.

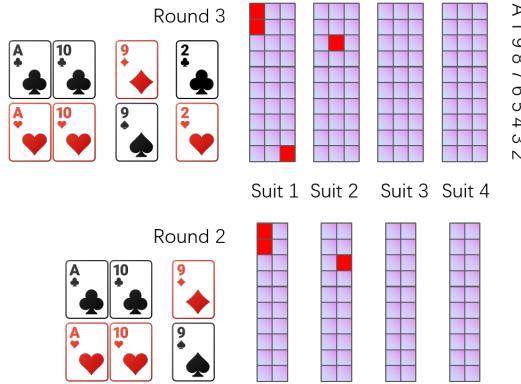


Figure 6: Hand Tensor Examples in Numeral211 Hold'em.

This approach is equally applicable to more complex Texas Hold'em games.

D.2 The Construction of the Hand Strength Tensor \mathbf{y}_q

Let's formalize the construction process of the hand strength tensor \mathbf{y} of a hand x . We largely draw on the ideas by Fu et al. (2025) to record the strengths of x and its predecessors in their respective rounds.

Suppose a poker game consists of a total of S rounds ($S = 3$ in Numeral211 Hold'em). We construct the hand strength tensor for a hand x at round s through the following systematic process:

For the terminal round S : Hands x in poker games naturally form a total order at terminal rounds. We directly compare its strength against all other valid opponent hands x' that share the same community cards at this round. This comparison yields a 3-length outcome vector $\mathbf{w} = [w_l, w_d, w_w]$, where:

- $w_l = \frac{\text{Number of } x' \text{ where } x \text{ loses to } x'}{\text{Total number of } x'}$
- $w_d = \frac{\text{Number of } x' \text{ where } x \text{ draws with } x'}{\text{Total number of } x'}$
- $w_w = \frac{\text{Number of } x' \text{ where } x \text{ wins against } x'}{\text{Total number of } x'}$

These values are inherently normalized such that $w_l + w_d + w_w = 1$.

In Numeral211 Hold'em, for example, during the terminal round, each player holds one private card, and there is one community card on the table. For a specific hand x (composed of the player's private card and the community card), the comparison set includes all other possible hands formed by pairing each of the remaining 38 private cards with the same community card (i.e., x'). The vector \mathbf{w} is thus calculated based on how x performs against these 38 potential opponent hands.

For a non-terminal round s : Since direct strength comparison between hands x is not feasible at non-terminal rounds, we employ a rollout process: each hand x at round s is rolled out to the terminal round S . For each rollout scenario $x \sqsubset x_k$ in S , we obtain a 3-length outcome vector \mathbf{w}_k following the terminal round comparison method. The final outcome vector for round s is calculated as a weighted sum $\mathbf{w} = \sum_{k=1}^K \alpha_k \mathbf{w}_k$, where α_k represents the probability weight of the k -th rollout path, and $\sum_{k=1}^K \alpha_k = 1$.

For a hand x evaluated at round s , we compute the 3-length outcome vectors for each round $1, 2, \dots, s$ using the above methods applied to the predecessors of x . The hand strength tensor $\mathbf{y}_q \in \mathbb{R}^{s \times 3}$ is then formed by stacking these vectors vertically, where each row corresponds to the outcome vector of a specific round, resulting in a 2-dimensional tensor with dimensions $[s, 3]$. This tensor provides a comprehensive strength profile across all relevant game rounds.

D.3 Structure of the Hand Feature Encoder

The hand feature encoder (HFE) is a simple structure. Since we model a hand as an image-like tensor, the HandEbdNet—essentially a regression task mapping x_q to y_q —adopts a CNN architecture to identify hand patterns (see Figure 7). Logically, HFE comprises a convolutional layer followed by a fully connected layer.

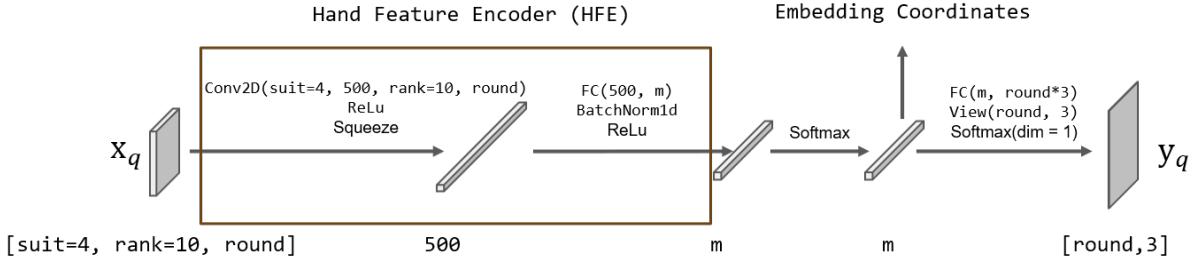


Figure 7: Structure of the HandEbdNet.

Notably, each convolution kernel matches the size of a single channel, mapping one channel’s data to a single value. This enables kernels to recognize suit-specific patterns (e.g., straight flushes, flushes). After ReLU activation and squeezing, a 500-length intermediate vector (hyperparameter) is produced. The subsequent fully connected layer then generates the target m -length vector, while capturing cross-suit hand strengths like pairs and straights.

D.4 Training and Inference

We train a separate network for each round, considering only Round 2 and Round 3. As shown in Figure 7, we need to specify the “round” parameter. The training data consists of all possible hand combinations in the current round: for Round 2, the number of combinations is $\binom{40}{2} \times \binom{38}{1} = 780 \times 38 = 29,640$; for Round 3, it is $\binom{40}{2} \times \binom{38}{1} \times \binom{37}{1} = 1,096,680$. Such data volumes are easily traversable and trainable on a home PC. Since the same data is used during inference, we do not need to consider issues of network robustness or overfitting—this is essentially a simple regression task, which is why a basic network structure suffices to solve this task.

E Experiment

For the EHS and PaEmd algorithms, there are no parameters to set; the only factor that can lead to differences is the initial clustering algorithm. For each algorithm, we generate three initial clusterings and iteratively compute the final clustering results.

For the KrwEmd algorithm, it requires setting an importance hyperparameter, which simply refers to the significance of the winning rate distance in each round. We select three sets of parameters: KrwEmd1, where the winning rate differences in late game are more important; KrwEmd2, where the winning rate differences in early game are more important; and KrwEmd3, where the winning rate differences in all rounds are equally important. One clustering result is generated for each set.

The complete experimental data are shown in the Figure 8. It can be observed that in long-term experiments, the differences in exploitability among the various clustering-based methods are insignificant, yet there exists an essential gap when compared with EmbeddingCFR in terms of exploitability.

Our experiments were conducted on a device equipped with 754GB of memory, an Intel(R) Xeon(R) Gold 6240R CPU (2.40GHz base frequency, up to 4.00GHz, 96 threads), and 7 NVIDIA TITAN RTX GPUs (each with 24GB memory). Each individual experiment takes approximately 34.2 days (with 32 threads).

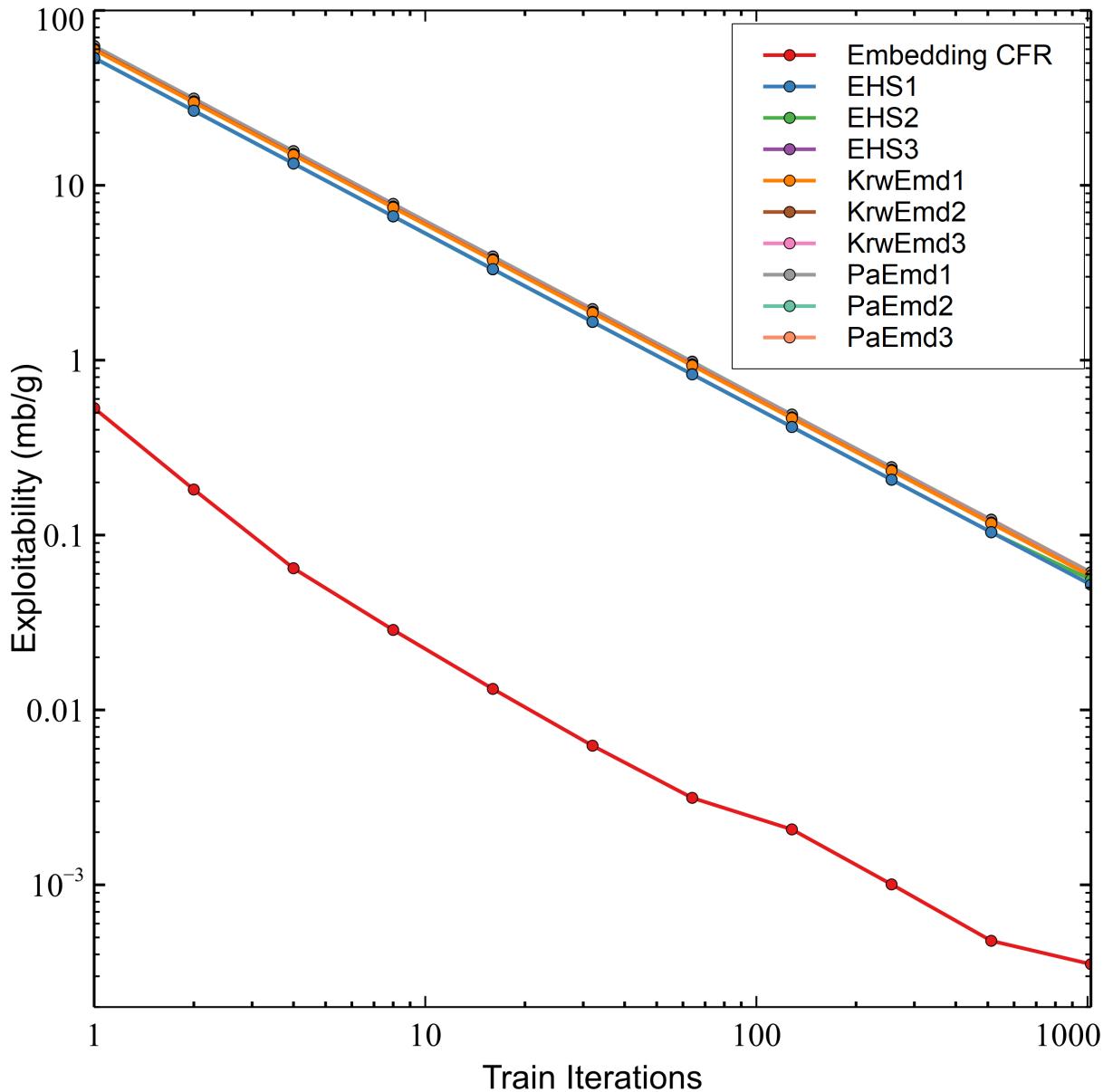


Figure 8: Complete Experimental Data