Abstraction Agent: LLM-Guided Abstraction for Imperfect-Information Games

Boning Li
IIIS, Tsinghua University
li-bn22@mails.tsinghua.edu.cn

Longbo Huang*
IIIS, Tsinghua University
longbohuang@tsinghua.edu.cn

Abstract

Information abstraction, which groups strategically similar private states into a tractable number of buckets, is essential for scaling game-solving algorithms to large imperfect-information games. Constructing effective abstractions, however, has traditionally required domain-specific evaluators such as hand-strength calculators or equity estimators, which demand expert knowledge and engineering effort and are unavailable for most less-studied games. We propose the *Abstraction Agent*, a zero-shot pipeline that uses a large language model (LLM) to discover continuous strategic features from a natural-language game description, score private states on these features, and cluster them into abstraction buckets, without any game-specific evaluator, training data, or game-tree traversal during abstraction construction. The pipeline runs in four phases: feature discovery with calibration anchors, batched private-state scoring, correlation-based feature selection, and k-means clustering. The resulting abstractions reduce lifted-strategy exploitability by up to 62% relative to an expected-hand-strength baseline on heads-up no-limit Texas hold'em (HUNL) turn endgames, and beat a scalar rank baseline at every granularity on ROVER Trials, an original game absent from any pretraining corpus. Beyond these quantitative benchmarks, the pipeline transfers with unchanged prompts to four-card Pot-Limit Omaha, HUNL preflop and flop, and Riichi Mahjong, where the discovered features track each game's recognized strategic concepts. This is *structured knowledge elicitation*: converting implicit strategic knowledge in LLM parameters into explicit numerical features for downstream algorithmic computation. The full pipeline code is available at https://github.com/lbn187/AbstractionAgent.

* Corresponding author.

1 Introduction

Large language models (LLMs) have shown strong capabilities in reasoning and planning across diverse tasks (Wei et al., 2022; Kojima et al., 2022). Recent work explores LLMs as game-playing agents (Meta Fundamental AI Research Diplomacy Team, 2022; Gandhi et al., 2023; Guo et al., 2023), yet in complex strategic games their play remains limited by imprecise probability estimation and inconsistent decision-making (Huang et al., 2024; Zhuang et al., 2025; Lin et al., 2026). We therefore propose using LLMs not as players, but as *feature engineers* that identify the strategic dimensions along which game states should be compared. This division of labor plays to the LLM's strength of articulating qualitative strategic structure in natural language, while delegating the quantitative equilibrium computation to an exact game solver, where LLMs are unreliable.

In imperfect-information games, where players must act under uncertainty about opponents' private states, *information abstraction* is essential for computational tractability. Abstraction groups strategically similar private states into buckets, enabling solvers like counterfactual regret minimization (CFR) to operate on manageable game sizes (Zinkevich et al., 2007). The quality of this grouping is a major determinant of solution quality: merging states with different strategic roles introduces exploitability.

Constructing effective abstractions has traditionally required domain-specific evaluators. In poker, hands are clustered by equity or expected hand strength (EHS), metrics that require hand evaluators, Monte Carlo rollouts, or expert-designed features (Johanson, 2013). Recent advances such as EVPA (Li and Huang, 2025) and WEVA (Li and Huang, 2026b) reduce the dependency on neural training or handcrafted features, but both still gather their abstraction signal by traversing the

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 26368–26387
October 24-29, 2026 ©2026 Association for Computational Linguistics

game tree. This traversal is computationally expensive for large games and, more fundamentally, presupposes that a game-specific solver or evaluator has already been implemented, which is often not the case for less-studied games. More broadly, many existing methods rely on at least one of: (1) game-specific evaluators, (2) extensive training data from game-tree traversal, or (3) closed-source implementations that make reproduction difficult (Brown and Sandholm, 2018, 2019).

This motivates our central question: *can an LLM, given only a natural-language description of the game rules, construct useful information abstractions without game-tree traversal during abstraction construction?*

We provide initial evidence that this is possible by proposing an *Abstraction Agent*, a zero-shot pipeline that uses LLM reasoning to discover and apply strategic features for information abstraction. By "zero-shot" we mean that the pipeline uses no game-specific training data, rollout evaluator, or game-tree traversal; it receives only a natural-language rules description and textual renderings of private states. The agent operates in four phases: (1) **Feature Discovery**, the LLM proposes continuous strategic features with calibration anchors from a game description; (2) **Private-State Scoring**, each holding is scored on these features via batched LLM calls; (3) **Feature Selection**, redundant features are filtered by correlation analysis; (4) **Clustering**, the resulting feature vectors are partitioned with k-means. The whole pipeline avoids domain-specific evaluators and manually designed scoring functions.

The key insight is that LLMs trained on broad text corpora encode useful strategic regularities about well-studied games. Concepts like "nut potential," "blocker value," and "draw vulnerability" exist in the LLM's knowledge as natural-language descriptions of strategic dimensions. Our pipeline elicits this knowledge as *structured, continuous features* suitable for downstream computation. We call this process *structured knowledge elicitation*: prompting an LLM to convert implicit domain knowledge into explicit, machine-readable numerical variables that can be consumed by non-neural downstream algorithms.

We evaluate on the two publicly available HUNL turn endgames from the Libratus release (Brown and Sandholm, 2018), where the agent's abstraction reduces exploitability relative to EHS by up to 62% at the finest granularity with the advantage growing monotonically in the bucket count. Because poker is heavily represented in pretraining corpora, we also evaluate on ROVER Trials, an original two-player zero-sum game absent from any pretraining corpus: given only the rules, the same pipeline beats the scalar rank baseline at every granularity and a 2D potential-aware baseline at most granularities, indicating that the agent *constructs* features rather than retrieving memorized poker knowledge. The unchanged pipeline also produces strategically coherent abstractions in PLO4, HUNL preflop/flop, and Riichi Mahjong, games where evaluator-based abstraction signals are too expensive to compute or unavailable, at a cost linear in the number of holdings.

Our contributions are:

1. **A zero-shot abstraction pipeline.** We propose the first LLM-based pipeline that converts a natural-language game description into a multi-dimensional information abstraction without hand evaluators, rollout simulators, or game-tree traversal during abstraction construction.

2. **Quantitative gains over standard baselines.** On the two public HUNL turn subgames, the agent's abstraction reduces exploitability relative to EHS by up to 62% at $K{=}100$, beating EHS in all $K{=}100$ runs without any domain-specific evaluator, with the advantage growing monotonically in the bucket count K; the gain requires a frontier backbone.

3. **Construction beyond memorization, with cross-game portability.** On ROVER Trials, an original game absent from any pretraining corpus, the rules-only pipeline beats the scalar rank baseline at every granularity, showing genuine construction rather than retrieval of memorized poker features, and produces strategically coherent abstractions in PLO4, HUNL preflop/flop, and Riichi Mahjong without modification.

2 Related Work

Information Abstraction in Games. Abstraction is a core technique for scaling imperfect-information game solving. In extensive-form games, information sets group states indistinguishable to a player; abstraction further merges information sets deemed strategically similar (Sandholm, 2015). Early work on poker abstraction used hand

rank bucketing (Shi and Littman, 2000; Billings et al., 2003). Modern approaches cluster private hands by expected hand strength (EHS) or equity against a uniform opponent range (Johanson, 2013; Johanson et al., 2013). Potential-aware abstractions (Gilpin et al., 2007; Ganzfried and Sandholm, 2014) additionally consider how hand values change with future public cards. RL-CFR (Li et al., 2024) uses reinforcement learning to optimize *action* abstraction, complementing information abstraction. EVPA (Li and Huang, 2025) trains neural networks on CFR traversal data to estimate per-hand features for online abstraction. WEVA (Li and Huang, 2026b) avoids neural training but still requires game-tree traversal. Our work removes the requirement for game-specific evaluators and game-tree traversal entirely, using LLM-derived features instead.

CFR and Exploitability. Counterfactual regret minimization (CFR) (Zinkevich et al., 2007) and its sampling-based variants (Lanctot et al., 2009; Li et al., 2026b) are standard algorithms for approximating Nash equilibria in two-player zero-sum games. Equilibria of such games can also be computed by linear programming, by first-order saddle-point methods such as the excessive gap technique (Kroer et al., 2018), and by no-regret learning dynamics such as fictitious play; we adopt CFR for its scalability and its ubiquity in modern poker solving. Superhuman poker AI systems (Moravčík et al., 2017; Brown and Sandholm, 2018, 2019) combine CFR with abstraction and real-time subgame solving. Recent work on parallelizing CFR (Li and Huang, 2026d) enables real-time solving on desktop hardware. We use CFR as the downstream solver and evaluate abstractions by the exploitability of the lifted average strategy (Waugh et al., 2009; Johanson et al., 2013) and, where exact best responses are intractable, certify agent performance with anytime-valid estimators (Li et al., 2026a). Downstream of equilibrium computation, agents may also deviate from equilibrium to exploit opponents under certified exploitability budgets (Li and Huang, 2026a), or replace static chip-value heuristics with computed continuations in tournament settings (Li and Huang, 2026c); both directions consume the abstractions studied here.

Domain-Knowledge-Free Abstraction. Deep CFR (Brown et al., 2019) replaces tabular regret with neural function approximation trained from self-play traversals, avoiding manual feature design but requiring large amounts of game-specific sampling. WEVA (Li and Huang, 2026b) eliminates neural training but still requires running CFR iterations on the unabstracted game. Our approach uses pretrained LLM knowledge as the source of abstraction signal, requiring neither game-specific training nor game-tree traversal, making it applicable to games where no solver implementation exists.

LLMs for Strategic Reasoning. We distinguish *strategic* game playing, in which a player reasons about equilibria and adapts to an opponent, from *non-strategic* play that follows fixed heuristics or imitates demonstrations (Gandhi et al., 2023). Recent work has explored LLMs as game-playing agents (Meta Fundamental AI Research Diplomacy Team, 2022; Guo et al., 2023; Wang et al., 2026b), strategic reasoners (Gandhi et al., 2023), and value estimators inside search (Karten et al., 2025). In poker, LLMs have been evaluated as zero-shot players (Gupta, 2023; Huang et al., 2024; Zhuang et al., 2025; Peng et al., 2025) with mixed results, reflecting the difficulty of reliable equilibrium-aware play; systematic diagnoses attribute the failures to heuristic shortcuts and misreadings of the game state (Lin et al., 2026). Concurrent work on structured prompting (Li et al., 2026c; Wang et al., 2026a) demonstrates that carefully structured prompts can activate latent poker strategy in frontier LLMs. While these works use LLMs as *players* or online evaluators, our work uses LLMs as *feature engineers* for offline abstraction construction, a complementary role that feeds into traditional game-solving pipelines.

LLMs as Feature Engineers. Explicit structural features are known to complement learned representations (Cao et al., 2015; Li et al., 2021). The idea of using LLMs to propose features has emerged in tabular learning (Hollmann et al., 2023) and AutoML contexts. More broadly, LLMs can translate natural-language rules into formal game structures: extensive-form game representations (Deng et al., 2025), executable world models for general game playing (Lehrach et al., 2025), and dynamic deduction that prunes actions and states during equilibrium learning (Zhang et al., 2026). Our work applies this paradigm to game-theoretic abstraction, where features must satisfy game-theoretic desiderata (spread, independence, coverage) that differ from standard ML feature quality metrics. The key technical challenge is eliciting *cal-*

ibrated numerical outputs from LLMs; we address this through calibration anchors and constrained output formatting.

3 Preliminaries

We briefly fix the standard game-theoretic objects used throughout the paper.

Extensive-form games. A two-player zero-sum *extensive-form game* is a tuple $\mathcal{G} = \langle \mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, P, \{\mathcal{I}_i\}, \{u_i\} \rangle$. $\mathcal{N} = \{1, 2\}$ is the set of players (a chance player c models stochastic events such as dealing cards). $\mathcal{H}$ is the set of *histories* (sequences of actions from the root); $\mathcal{Z} \subseteq \mathcal{H}$ are *terminal* histories. At a non-terminal history h, the player to act $P(h) \in \mathcal{N} \cup \{c\}$ chooses from the available actions $\mathcal{A}(h)$. Each terminal $z \in \mathcal{Z}$ yields utility $u_i(z) \in \mathbb{R}$ to player i; the game is *zero-sum* when $u_1(z) = -u_2(z)$.

Information sets. Because players act under uncertainty about hidden information (e.g., the opponent's cards), the decision histories of player i are partitioned into *information sets* $\mathcal{I}_i$. Two histories in the same information set $I \in \mathcal{I}_i$ are indistinguishable to i, so the same actions $\mathcal{A}(I)$ are available and i must behave identically at both. An information set is identified by the player's *private type* (its private state, e.g., its hole cards) together with the observed public history.

Strategies and equilibrium. A *behavioral strategy* σ_i assigns to each $I \in \mathcal{I}_i$ a distribution over $\mathcal{A}(I)$. A *strategy profile* is $\sigma = (\sigma_1, \sigma_2)$, and $u_i(\sigma)$ denotes player i's expected utility under σ (taking expectations over chance and both players' randomization). A *best response* for i to the opponent profile σ_{-i} attains $\max_{\sigma_i'} u_i(\sigma_i', \sigma_{-i})$. A *Nash equilibrium* is a profile where neither player can gain by deviating.

Exploitability. The quality of a profile σ in a two-player zero-sum game is measured by its *exploitability*, the average amount an optimal adversary gains against each player:

$$\mathrm{expl}(\sigma) = \tfrac{1}{2} \sum_{i \in \mathcal{N}} \Big(\max_{\sigma_i'} u_i(\sigma_i', \sigma_{-i}) - u_i(\sigma) \Big). \tag{1}$$

A Nash equilibrium has $\mathrm{expl}(\sigma) = 0$; larger values indicate strategies further from optimal. We report exploitability as a fraction of the pot.

Information abstraction. Solving a large game exactly is intractable, so an *information abstraction* ϕ groups the private types reaching a given public state into $K \ll N$ *buckets*, $\phi : \{h_1, \ldots, h_N\} \to \{1, \ldots, K\}$. A solver computes a strategy σ^ϕ on the smaller abstract game, which is then *lifted* back to the original game by forcing all private types in the same bucket to share the same action distribution. Because merged types must behave identically, a poor grouping raises the exploitability of the lifted strategy; the abstraction quality we evaluate is exactly expl of this lifted profile, computed by an exact best response over the unabstracted game.

4 Method

We propose an *Abstraction Agent* that builds an information abstraction for an imperfect-information game through a four-phase pipeline (Figure 1). The agent is game-agnostic: all domain knowledge enters through a declarative `GameConfig` that gives natural-language descriptions of the game rules and context. No hand-strength calculators, equity estimators, or rollout simulators are used.

Problem setting. Building on Section 3, we construct an information abstraction ϕ at a fixed public state c (the community cards in poker, the visible discards in Mahjong). At this public state each player holds one of N *private types* $h_1, \ldots, h_N$; we call a private type interchangeably a *private state* or *holding* (hole cards in poker, concealed tiles in Mahjong). The abstraction $\phi : \{h_1, \ldots, h_N\} \to \{1, \ldots, K\}$ assigns each private type to one of K buckets; a solver (CFR) computes a strategy over the buckets, which is then *lifted* back to the original game as in Section 3, so private types in the same bucket share an action distribution. The agent's goal is a ϕ whose lifted strategy has low exploitability (Eq. 1) *without any game-specific evaluator*.

Design rationale. An alternative is to have the LLM bucket holdings directly, but that requires every holding in one context for consistent grouping and a fresh generation for each K. Scoring against fixed features instead keeps each call local: the anchors pin each feature's endpoints and midpoint to concrete situations, so batches scored independently share a scale, and the global partitioning is left to K-means. The explicit features also remain readable before any solving, and Section 7 traces

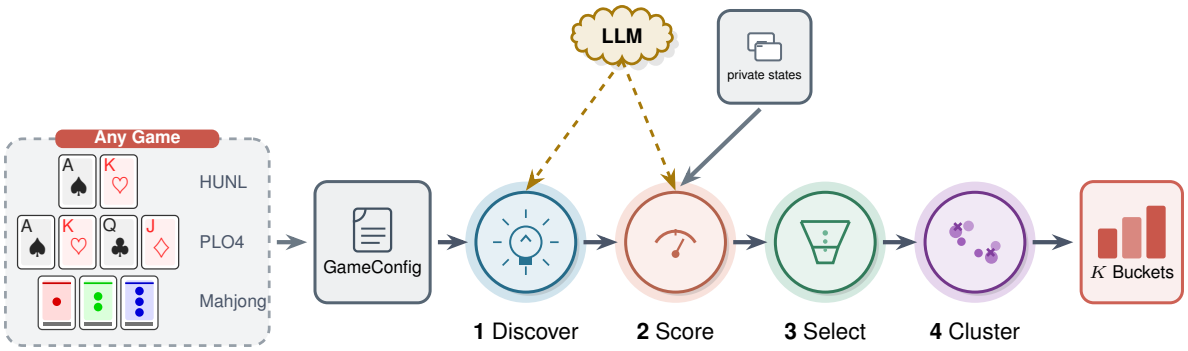


Figure 1: The Abstraction Agent pipeline. Given a textual `GameConfig` and a set of private holdings, an LLM discovers strategic features (Phase 1), scores each holding on them (Phase 2), redundant dimensions are filtered (Phase 3), and holdings are clustered into K buckets (Phase 4). The pipeline uses no hand evaluators, training data, or game-tree traversal, and the same code applies to poker, Omaha, and Mahjong unchanged.

abstraction quality back to individual features.

Phase 1: feature discovery. Given a `GameConfig` with the game rules and strategic context, the agent prompts the LLM to propose a set of continuous strategic features $\{f_1, \ldots, f_M\}$ (typically $M \in [3, 8]$). Each feature f_j carries:

- a **name** (e.g., NUT_CEILING, how close a holding is to the strongest possible hand on the current board);
- a **description** of what it measures;
- **calibration anchors** mapping scores 0.0, 0.5, and 1.0 to concrete game situations.

The prompt asks for features that are continuous in $[0, 1]$, spread across the hand space (std ≥ 0.15), mutually independent, and jointly covering the main strategic axes; when the game has future public cards it additionally requests future-card sensitivity. This phase needs a single LLM call, cached for reuse across K.

Phase 2: private-state scoring. For each private holding h_i the agent prompts the LLM with the public context, the discovered feature definitions and anchors, and a batch of 20–30 holdings, and asks for a score vector $\mathbf{s}_i \in [0, 1]^M$. The anchors provide a shared reference frame so scores stay consistent across batches. Holdings are processed in batches with retry logic and caching; a score that still fails to parse defaults to the neutral midpoint 0.5, placing the holding in a central bucket rather than dropping it. In practice parse rates are at least 97.9%, so the fallback rarely fires. This yields the feature matrix $\mathbf{S} \in \mathbb{R}^{N \times M}$.

Phase 3: feature selection. Not all discovered features are informative or independent. We drop columns with standard deviation below $\tau_\sigma = 0.10$ (the LLM assigned near-constant scores), then, for each surviving pair with Pearson correlation $|r| > \tau_r = 0.90$, drop the lower-variance member. The survivors are rescaled by dividing each column by its standard deviation (no mean-centering, since the anchors already provide a shared zero-point), giving $\mathbf{Z} \in \mathbb{R}^{N \times M'}$.

Phase 4: clustering. We partition the N holdings into K buckets with multi-restart K-means on $\mathbf{Z}$:

$$\phi(h_i) = \arg\min_k \|\mathbf{z}_i - \boldsymbol{\mu}_k\|^2 \qquad (2)$$

using K-means++ initialization, 10 restarts of 50 iterations each, keeping the partition with minimal within-cluster sum of squares. Algorithm 1 summarizes the pipeline.

5 Experimental Setup

HUNL Endgames. We evaluate on the two turn endgames publicly available from the Libratus subgame release (Brown and Sandholm, 2018), which form the complete set of public turn subgames from that release rather than a hand-picked subset. Each subgame specifies a public board (4 community cards with one more to come), pot size, and private hands reaching the decision point. Turn boards are the most strategically complex street: hand rankings can change when the final card is revealed, creating multi-dimensional strategic value (current strength, draw potential, vulnerability) that a single scalar metric cannot capture. Private hands are filtered by reach probability to retain strategically

Algorithm 1 Abstraction Agent Pipeline

Require: Game config $\mathcal{G}$, holdings $\{h_1, \ldots, h_N\}$, public state c, bucket count K
Ensure: Abstraction $\phi : \{h_i\} \to \{1, \ldots, K\}$

Phase 1: Feature Discovery
$\{f_1, \ldots, f_M\} \leftarrow \text{LLM}(\mathcal{G}.\text{game_description})$
Phase 2: Private-State Scoring
for each batch $B \subset \{h_1, \ldots, h_N\}$ **do**
  $\mathbf{s}_i \leftarrow \text{LLM}(c, \{f_j\}, B)$ for $h_i \in B$
end for
$\mathbf{S} \leftarrow [\mathbf{s}_1; \ldots; \mathbf{s}_N] \in \mathbb{R}^{N \times M}$
Phase 3: Feature Selection
Drop columns with std $< \tau_\sigma$
Drop correlated pairs with $|r| > \tau_r$
$\mathbf{Z} \leftarrow$ standardize surviving columns
Phase 4: Clustering
$\phi \leftarrow K\text{-means}(\mathbf{Z}, K)$ with multi-restart
return ϕ

relevant holdings, evaluated at $K \in \{20, 50, 100\}$ buckets. Each condition runs 10 independent seeds (each drawing a fresh feature-discovery sample), and we report the mean, standard deviation, and a 95% confidence interval over the seeds.

ROVER Trials. To test whether the agent *constructs* strategic features rather than retrieving memorized poker knowledge, we introduce *ROVER Trials*, an original two-player zero-sum betting game of hidden information designed for this paper and absent from any pretraining corpus. Each player privately holds a *rover*, described by three integer dials each in $\{0, \ldots, 4\}$: POWER (raw engine strength), GRIP (traction on rough ground), and AFFINITY (a single specialized terrain type), giving $5^3 = 125$ private types drawn uniformly and independently. Both players ante one chip and play two betting rounds; between them a public *terrain* is revealed uniformly from five types (sand/grass are smooth; rock/mud/ice are rough). A rover travels distance $\text{Power} + \text{Grip} \cdot \mathbb{1}[\text{terrain rough}] + b \cdot \mathbb{1}[\text{AFFINITY} = \text{terrain}]$; the farther rover wins the pot. We set the affinity bonus $b = 10$, making a rover a "specialist" that is dominant on its own (one-in-five) terrain but ordinary elsewhere; expected distance alone then cannot separate a volatile specialist from a steady rover of the same mean. The LLM receives only the rules text, with no poker terminology, no feature hints, and no evaluator, so any useful abstraction must be assembled from the rules. We evaluate $K \in \{5, 10, 20, 30\}$ buckets over 10 independent seeds per backbone. Baselines mirror the poker setting: *Rank* buckets rovers by expected distance (the ROVER analog of EHS) and requires the payoff simulator; *PA* clusters the 2D potential-aware features (expected distance, $\text{std}_{\text{terrain}}$), the analog of potential-aware abstraction; *Random* and *full* (no abstraction) complete the comparison. Solving uses CFR+ for 500 iterations, with exploitability computed by exact best response on the unabstracted tree.

Cross-game settings. Three further settings test portability of the unchanged pipeline. In Pot-Limit Omaha (PLO4) preflop, the full hand space of $\binom{52}{4} = 270{,}725$ raw combinations reduces to 16,432 canonical representatives after suit isomorphism, clustered with $K = 30$; no exploitability is computed, and we instead inspect whether the discovered features capture recognized PLO strategic concepts. In HUNL preflop (169 canonical hands, $K = 10$) and flop (1,176 valid holdings, $K = 20$), we test cross-street generalization without pipeline modification. Finally, to move beyond card games entirely, we apply the agent to Japanese Riichi Mahjong (MahjongRepository, 2024), a tile-based game with 136 tiles, chance nodes (random draws from the wall), and no shared structure with poker: we fix a mid-game scenario (turn 8, East round, 14 visible discards) and sample 200 random 13-tile hands ($K = 10$).

Baselines. We compare against the standard information-abstraction baselines used in prior poker-solving work (Johanson, 2013; Ganzfried and Sandholm, 2014), plus a random lower bound and an LLM text-embedding baseline. *EHS* assigns each hand its expected equity against a uniform opponent range over all possible river cards and clusters this scalar with K-means; it requires a hand evaluator and Monte Carlo rollout. *Potential-Aware (PA)* clusters hands in the 2D space of (EHS, $\sqrt{\text{Var}[\text{HS}]}$), with variance computed across all river cards (Ganzfried and Sandholm, 2014); it requires a hand evaluator plus full river enumeration. *LLM-Embed (zero-shot)* has the LLM write a text description of each hand, embeds the descriptions with a sentence encoder, and clusters them with K-means, testing whether raw embeddings capture useful structure without explicit feature discovery. *Random* assigns hands to buckets uniformly at random. Both evaluator-based baselines use domain-specific code that our agent does not.

Evaluation metric. For HUNL endgames, we report exploitability of the lifted strategy as a fraction of the pot. The lifted strategy is obtained by running CFR for 2,000 iterations on the abstract game and mapping the resulting strategy back to the original game. Lower exploitability indicates a better abstraction. We also report the **ratio** of agent exploitability to EHS exploitability; ratio < 1 means the agent outperforms EHS.

Implementation and cost. We use GPT-5.5 via an OpenAI-compatible API for the main results. For the turn subgames, the `GameConfig` supplies the public board, pot, and a summary of the players' range composition (nonzero-hand counts and entropies), all available at solving time without any hand evaluator or solver output; a rules-only variant that omits the range summary behaves identically (Appendix J). To test backbone dependence we run the same pipeline with open-weight Llama-3.3-70B and Qwen-2.5-32B. Feature discovery is a single LLM call per game at temperature 0.7 with high reasoning effort; private-state scoring uses temperature 0.0 with deterministic decoding and low reasoning effort, in batches of 20–30 holdings with 3-attempt retries. We define *parse rate* as the fraction of scoring responses that match the required structured format without manual correction. Features are filtered (min std $= 0.10$, max correlation $= 0.90$) and rescaled before K-means++ clustering (10 restarts, 50 iterations each). CFR uses discounted updates (DCFR; $\alpha = 1.5$, $\beta = 0$, $\gamma = 2$) with alternating regret matching for 2,000 iterations; exploitability is computed exactly by best-response traversal over the unabstracted game tree. Each seed redraws both the feature-discovery sample and the K-means initialization; prompts are identical across games except the `GameConfig` content (full prompts in Appendix C). Clustering stability across seeds yields mean pairwise ARI of 0.77. Per seed the pipeline makes one feature-discovery call and roughly ten scoring batches, on the order of a dozen LLM calls; wall-clock is dominated by API latency (a few to $\sim$15 minutes per seed), while local feature selection and clustering take under a second. Scoring depends only on the discovered features and holdings, not on K, so its cost is amortized across granularities: the $K \in \{20, 50, 100\}$ abstractions for a seed reuse a single scoring pass. Downstream CFR (2,000 DCFR iterations) is the dominant compute cost and grows with K.

Table 1: HUNL turn endgame exploitability (fraction of pot; mean over 10 seeds). **Ratio** = mean Agent / EHS $\pm$ 95% Student-t CI half-width.

Board	K	Agent	EHS	Ratio
7♠9♡9♣T♣	20	0.0437	0.0335	1.30 ± 0.20
	50	0.0142	0.0192	0.74 ± 0.17
	100	0.0036	0.0098	$\mathbf{0.36} \pm 0.04$
T♠6♡A♡7♣	20	0.1359	0.1088	1.25 ± 0.23
	50	0.0483	0.0592	0.82 ± 0.10
	100	0.0102	0.0256	$\mathbf{0.40} \pm 0.08$

6 Results

HUNL turn endgames. Table 1 reports lifted-strategy exploitability on the two turn endgames. For each subgame and granularity K we run the pipeline for 10 independent seeds (each with a fresh feature-discovery sample at temperature 0.7) and report the mean and standard deviation of exploitability, its ratio to the EHS baseline with a 95% confidence interval, and the best run; ratios below 1.0 mean the agent outperforms EHS. All runs use GPT-5.5 and parse at 100%.

As K grows from 20 to 100, the mean exploitability ratio falls from 1.28 to 0.78 to 0.38. At K=100 the agent cuts exploitability by roughly 62% relative to EHS, with a 95% CI below 1.0 on both subgames and all 20 runs beating EHS. Finer abstractions give the multi-dimensional feature space more room to separate hands, while a one-dimensional EHS ordering gains little from extra buckets once the major strength tiers are split. At K=20, by contrast, the agent underperforms EHS: with few buckets and after de-correlation filtering, the surviving feature basis is sometimes too thin to beat a clean equity ordering (Section 7).

Backbone dependence. If the gains come from the backbone's internalized strategic knowledge, weaker models under the same prompt should not reproduce them. Table 2 confirms this: only the frontier GPT-5.5 reliably beats EHS; Llama-3.3-70B is near parity; Qwen-2.5-32B is consistently worse.

Comparison with Other Methods. Table 3 compares the agent with baselines at K=50. It beats a random assignment, the zero-shot LLM-Embed baseline, and EHS, without any domain-specific evaluator. PA, which has access to exact equity calculators and full river enumeration, achieves the lowest exploitability; the agent closes about a third of that gap using only LLM-derived fea-

Table 2: Backbone dependence: mean exploitability ratio (Agent / EHS) over 10 seeds.

K	Backbone	SG1	SG2
50	GPT-5.5	0.74	0.82
	Llama-3.3-70B	1.62	1.86
	Qwen-2.5-32B	1.48	2.01
100	GPT-5.5	**0.36**	**0.40**
	Llama-3.3-70B	1.07	1.36
	Qwen-2.5-32B	1.96	3.70

Table 3: Comparison of abstraction methods at K=50 (exploitability as fraction of pot; agent is the 10-seed mean).

Method	SG1	SG2
Random	0.1314	0.3670
LLM-Embed (zero-shot)	0.0318	0.1847
Abstraction Agent (ours)	**0.0142**	**0.0483**
Domain-specific methods (require hand evaluator):		
EHS	0.0192	0.0592
PA (EHS + Variance)	0.0053	0.0254

tures. We compare directly against methods whose abstraction signal is computable here: Deep CFR (Brown et al., 2019) and WEVA (Li and Huang, 2026b) need CFR iterations on the unabstracted game, infeasible at HUNL turn scale and undefined for PLO/Mahjong without a solver, and potential-aware abstraction (Ganzfried and Sandholm, 2014) needs full river enumeration. Where a computable analog exists, namely EHS and PA in poker and the Rank and 2D potential-aware baselines in ROVER (Section 6), we include it directly.

ROVER Trials: Construction Beyond Memorization. Table 4 reports lifted-strategy exploitability ratios on ROVER Trials, an original game absent from any pretraining corpus; the LLM receives only the rules text and no poker terminology, evaluator, or game-tree traversal. The two baselines are *Rank*, the expected-distance (EHS) analog, and *PA*, the potential-aware 2D analog; both require the payoff simulator, which the agent does not use. GPT-5.5 beats the scalar Rank baseline at every granularity, with all 95% CIs below 1.0, and the margin widens with granularity (from 0.92 at K=5 to 0.66 at K=30). This is the same pattern as on HUNL turn: once buckets are plentiful, a multi-dimensional feature space separates types that a scalar ordering must merge, here volatile terrain specialists from steady rovers of the same expected distance. Against the stronger PA baseline the agent improves at the mean for

Table 4: ROVER Trials: mean exploitability ratio (Agent / baseline) over 10 seeds $\pm$ 95% Student-t CI half-width.

Backbone	K	Agent/Rank	Agent/PA
GPT-5.5	5	0.920 ± 0.074	1.043 ± 0.106
	10	0.804 ± 0.038	0.940 ± 0.067
	20	0.667 ± 0.070	0.913 ± 0.106
	30	0.663 ± 0.065	0.916 ± 0.111
Qwen-2.5-32B	5	1.157 ± 0.085	1.319 ± 0.160
	10	0.969 ± 0.059	1.128 ± 0.054
	20	0.902 ± 0.058	1.238 ± 0.117
	30	0.965 ± 0.078	1.328 ± 0.117
Llama-3.3-70B	5	1.102 ± 0.103	1.254 ± 0.162
	10	0.924 ± 0.060	1.080 ± 0.088
	20	0.760 ± 0.082	1.040 ± 0.116
	30	0.760 ± 0.098	1.046 ± 0.147

$K \in \{10, 20, 30\}$ (ratios 0.91–0.94), but those CIs overlap 1.0, so the advantage is directional: rules-only features roughly match, rather than clearly beat, a simulator-based 2D abstraction. The backbone gradient also mirrors HUNL: Llama-3.3-70B beats Rank at $K \geq 10$ (ratio 0.76 at K=20, 30) but not PA, and Qwen-2.5-32B is below Rank only marginally and only at the finer granularities. The unabstracted full game has exploitability 0.0019 chips, so abstraction still costs something; the relevant question is how much of the gap to evaluator-based abstractions a rules-only pipeline closes.

Because ROVER has no published strategy literature and no per-holding bucketing to copy, these results support the *construction* reading: the LLM assembles useful features from the supplied rules rather than reciting a memorized recipe (Section 7).

Cross-game portability. Beyond HUNL turn, the same pipeline (unchanged code and prompts) yields coherent abstractions in PLO4 preflop (16,432 hands, K=30), HUNL flop (1,176 holdings, K=20), HUNL preflop (169 hands, K=10), and Riichi Mahjong (200 mid-game hands, K=10). The discovered features track each game's recognized strategy: suit coordination and straight connectivity in PLO4 (Billings et al., 2003), draw-versus-made separation on the flop, premium-pair grouping preflop, and shanten-speed / acceptance / draw-swing features in Mahjong, the last reflecting its chance nodes. Parse rates are near 100%. Appendix E lists the features per game and summarizes the cross-game results (Table 8); Appendix H, L, I, F show representative clusters.

These settings also mark the practical boundary

of evaluator-based abstraction. EHS and PA presuppose a hand evaluator with Monte Carlo rollouts or full enumeration of future public cards, which becomes expensive for PLO4's 16,432 canonical four-card hands and is unavailable for Riichi Mahjong, where no equity evaluator or open-source solver exists. The agent's cost is instead linear in the number of holdings, about a dozen LLM calls per seed, which is what makes abstractions for these games obtainable here at all.

7 Analysis

Why multi-feature abstraction outperforms EHS. EHS compresses all strategic information into a single scalar, losing vulnerability, nut potential, and blocker effects. The feature–EHS correlation analysis (Appendix G) shows this: the made-strength feature correlates strongly with EHS ($r \approx 0.87$–0.93), while the improvement- and river-shift features have low or negative EHS correlation and thus capture complementary dimensions. Clustering in multi-dimensional space avoids merging hands with similar EHS but different strategic roles.

Granularity effects. The agent's advantage grows monotonically with granularity: the mean exploitability ratio falls from 1.28 at $K{=}20$ to 0.78 at $K{=}50$ to 0.38 at $K{=}100$ (Table 1). Finer abstractions partition the multi-dimensional space more precisely, while one-dimensional EHS gains less from extra buckets once the major strength tiers are separated. Clustering is stable across seeds: mean pairwise ARI is 0.77 overall, with $K{=}50$ (0.77–0.88) more stable than $K{=}20$ (0.63–0.80). Leave-one-out ablations (Appendix K) confirm that VULNERABILITY_TO_OUTDRAWS and BLOCKER_VALUE carry the most unique clustering signal.

Comparison with potential-aware methods. Potential-aware abstractions (Ganzfried and Sandholm, 2014) extend EHS with future-card distributions; EVPA (Li and Huang, 2025) learns neural features online during CFR. Our IMPROVEMENT_POTENTIAL and VULNERABILITY_TO_OUTDRAWS capture analogous information through natural-language reasoning. PA achieves lower exploitability (Table 3), since it has exact equity calculators; our value is in *beating* the evaluator-based EHS baseline and closing about a third of the gap to PA *without* any game-specific evaluator or game-tree traversal. This zero-shot setting is the realistic case for games without existing solver implementations: no open-source Mahjong CFR solver exists, yet the pipeline produces meaningful strategic clusters in under a minute.

Constructed versus retrieved features. Because HUNL is heavily represented in pretraining corpora, one might worry that the LLM is *retrieving* memorized poker features rather than *constructing* them from the rules. This distinction does not affect the practical contribution: the claim is that useful, evaluator-free abstractions can be elicited zero-shot from a rules description either way. Three observations support the construction reading. First, the same pipeline discovers chance-sensitive features in Riichi Mahjong and suit-coordination features in PLO4, domains with far thinner strategy literature than HUNL and no published per-hand bucketing to copy. Second, the backbone comparison (Table 2) shows that giving the *same* rules and prompt to weaker models does not yield useful abstractions, so the gains track genuine strategic competence rather than surface recall. Third, and most directly, the ROVER Trials (Section 6) are an original game introduced in this paper and absent from any pretraining corpus; the agent receives only its rules text, with no poker terminology, no evaluator, and no feature hints, yet the pipeline still discovers features that beat a scalar rank baseline and a 2D potential-aware baseline across most settings (Table 4). Features that succeed on a game the model has never seen must be constructed from the rules rather than retrieved from memory.

8 Conclusion

We presented an Abstraction Agent that uses LLMs to discover and score strategic features for information abstraction from natural-language game descriptions, without game-specific evaluators or game-tree traversal. On the two public HUNL turn endgames, the agent reduces exploitability relative to EHS by up to 62%, with all $K{=}100$ runs beating EHS and the advantage growing in the bucket count. On ROVER Trials, an original game absent from any pretraining corpus, the rules-only pipeline beats the scalar rank baseline at every granularity, indicating construction from the rules rather than retrieval from memory; the unchanged pipeline also produces coherent abstractions in PLO4, HUNL preflop/flop, and Riichi Mahjong. These results position LLMs as structured interfaces that turn natural-language rules into formal computation.

Limitations

Our work has several limitations. First, the quantitative evaluation is limited to the two HUNL turn endgames that are publicly available from the Libratus subgame release, which form the complete public set of such turn subgames, and the original ROVER game, so broader validation across more game instances and game types remains future work. Second, the agent's advantage is strongest on streets with future cards; on the river, where EHS equals exact equity, multi-dimensional features provide limited additional benefit. Third, the method depends on the backbone's strategic knowledge: a frontier model (GPT-5.5) is needed to reliably beat EHS, whereas open-weight 32B/70B models are not competitive under the identical pipeline (Table 2); making the approach work with weaker or specialized backbones remains open. Relatedly, the advantage is granularity-dependent: at the coarse $K{=}20$ the agent underperforms EHS. Fourth, beyond the HUNL turn endgames and ROVER, the remaining cross-game results (PLO4, HUNL preflop/flop, Mahjong) are qualitative and assess strategic coherence rather than downstream exploitability. Fifth, while the pipeline avoids game-specific evaluators, it still requires a textual game description and rendering functions for private states, which presupposes some understanding of the game's structure.

Ethics Statement

This work proposes a method for constructing information abstractions in imperfect-information games. The primary application domain is poker, which involves gambling. Although our experiments are research-oriented and evaluated on standard academic benchmarks (Libratus subgames), improved poker abstractions could in principle be used in automated gambling systems. We release code for reproducibility in controlled research settings but do not provide deployment tools for real-money play, and we encourage use only in academic contexts. The method itself is game-agnostic and has applications beyond gambling, including security games, negotiation, and multi-agent coordination.

Acknowledgments

This work was supported by the National Natural Science Foundation of China Grant 52494974.

References

Darse Billings, Neil Burch, Aaron Davidson, Robert Holte, Jonathan Schaeffer, Terence Schauenberg, and Duane Szafron. 2003. Approximating game-theoretic optimal strategies for full-scale poker. In *International Joint Conference on Artificial Intelligence*, pages 661–668.

Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. 2019. Deep counterfactual regret minimization. In *International Conference on Machine Learning*, pages 793–802.

Noam Brown and Tuomas Sandholm. 2018. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424.

Noam Brown and Tuomas Sandholm. 2019. Superhuman ai for multiplayer poker. *Science*, 365(6456):885–890.

Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 891–900.

Shilong Deng, Yongzhao Wang, and Rahul Savani. 2025. From natural language to extensive-form game representations. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pages 593–601.

Kanishk Gandhi, Dorsa Sadigh, and Noah D Goodman. 2023. Strategic reasoning with language models. *arXiv preprint arXiv:2305.19165*.

Sam Ganzfried and Tuomas Sandholm. 2014. Potential-aware imperfect-recall abstraction with earth mover's distance in imperfect-information games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Andrew Gilpin, Tuomas Sandholm, and Troels Bjerre Sørensen. 2007. Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of texas hold'em poker. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 50.

Jiaxian Guo, Bo Yang, Paul Yoo, Bill Yuchen Lin, Yusuke Iwasawa, and Yutaka Matsuo. 2023. Suspicion-agent: Playing imperfect information games with theory of mind aware GPT-4. *CoRR*, abs/2309.17277.

Akshat Gupta. 2023. Are chatgpt and GPT-4 good poker players? - A pre-flop analysis. *CoRR*, abs/2308.12466.

Noah Hollmann, Samuel Müller, and Frank Hutter. 2023. Large language models for automated data science: Introducing CAAFE for context-aware automated feature engineering. In *Advances in Neural Information Processing Systems*.

Chenghao Huang, Yanbo Cao, Yinlong Wen, Tao Zhou, and Yanru Zhang. 2024. PokerGPT: An end-to-end lightweight solver for multi-player texas hold'em via large language model. *CoRR*, abs/2401.06781.

Michael Johanson. 2013. Measuring the size of large no-limit poker games. *arXiv preprint arXiv:1302.7008*.

Michael Johanson, Neil Burch, Richard Valenzano, and Michael Bowling. 2013. Evaluating state-space abstractions in extensive-form games. In *International Conference on Autonomous Agents and Multi-Agent Systems*, pages 271–278.

Seth Karten, Andy Luu Nguyen, and Chi Jin. 2025. Pokéchamp: an expert-level minimax language agent. In *Forty-second International Conference on Machine Learning*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*.

Christian Kroer, Gabriele Farina, and Tuomas Sandholm. 2018. Solving large sequential games with the excessive gap technique. In *NeurIPS*, pages 872–882.

Marc Lanctot, Kevin Waugh, Martin Zinkevich, and Michael Bowling. 2009. Monte carlo sampling for regret minimization in extensive games. *Advances in Neural Information Processing Systems*, 22.

Wolfgang Lehrach, Daniel Hennes, Miguel Lazaro-Gredilla, Xinghua Lou, Carter Wendelken, Zun Li, Antoine Dedieu, Jordi Grau-Moya, Marc Lanctot, Atil Iscen, et al. 2025. Code world models for general game playing. *arXiv preprint arXiv:2510.04542*.

Boning Li, Yu Chen, and Longbo Huang. 2026a. Av-aivat: 74x cheaper agent evaluation with certified anytime-valid stopping in imperfect-information games. *arXiv preprint arXiv:2608.06362*.

Boning Li, Yu Chen, and Longbo Huang. 2026b. Correlated chance sampling for monte carlo counterfactual regret minimization. *arXiv preprint arXiv:2607.27035*.

Boning Li, Zhixuan Fang, and Longbo Huang. 2024. Rl-cfr: improving action abstraction for imperfect information extensive-form games with reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 27752–27770.

Boning Li and Longbo Huang. 2025. Efficient online pruning and abstraction for imperfect information extensive-form games. In *The Thirteenth International Conference on Learning Representations*.

Boning Li and Longbo Huang. 2026a. Agents that certify their own exploits: Confidence-scheduled restricted responses for safe opponent exploitation. *arXiv preprint arXiv:2607.28520*.

Boning Li and Longbo Huang. 2026b. Effective, efficient, and general information abstraction for imperfect-information extensive-form games. *arXiv preprint arXiv:2605.10900*.

Boning Li and Longbo Huang. 2026c. Icm out! better tournament strategy from computed continuations, vs. solvers and llms. *arXiv preprint arXiv:2608.09586*.

Boning Li and Longbo Huang. 2026d. Real-time parallel counterfactual regret minimization. *arXiv preprint arXiv:2605.19928*.

Boning Li, Baoxiang Wang, and Longbo Huang. 2026c. Pokerskill: Llms can play expert-level poker without training or solvers. *arXiv preprint arXiv:2605.30094*.

Boning Li, Yingce Xia, Shufang Xie, Lijun Wu, and Tao Qin. 2021. Distance-enhanced graph neural network for link prediction. In *ICML 2021 Workshop on Computational Biology*. Virtual Event.

Minhua Lin, Enyan Dai, Hui Liu, Xianfeng Tang, Yuliang Yan, Zhenwei Dai, Jingying Zeng, Zhiwei Zhang, Fali Wang, Hongcheng Gao, Chen Luo, Xiang Zhang, Qi He, and Suhang Wang. 2026. How far are LLMs from professional poker players? revisiting game-theoretic reasoning with agentic tool use. In *The Fourteenth International Conference on Learning Representations*.

MahjongRepository. 2024. mahjong: Implementation of japanese riichi mahjong hand calculation. https://github.com/MahjongRepository/mahjong. MIT License, version 1.4.0.

Meta Fundamental AI Research Diplomacy Team. 2022. Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074.

Matěj Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. 2017. DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513.

Wenjun Peng, Jing Zhe Lim, Qinghao Liu, and Xinyu Wang. 2025. Can large language models play to win? game-theoretic benchmarks in poker for probabilistic reasoning evaluation. In *Pacific Rim International Conference on Artificial Intelligence*, pages 313–328. Springer.

Tuomas Sandholm. 2015. Abstraction for solving large incomplete-information games. In *AAAI Conference on Artificial Intelligence*.

Jiefu Shi and Michael L Littman. 2000. Abstraction methods for game theoretic poker. In *Computers and Games*, pages 333–345.

Han Wang, Philippe Beardsell, Boning Li, Aaron Sasmita, Shuai Li, Hongyuan Zha, and Baoxiang Wang. 2026a. Solver-guided reasoning

for mixed-equilibrium strategies. *arXiv preprint arXiv:2608.06741*.

Wei Wang, Fuqing Bie, Junzhe Chen, Dan Zhang, Shiyu Huang, Evgeny Kharlamov, and Jie Tang. 2026b. Can large language models master complex card games? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Kevin Waugh, Martin Zinkevich, Michael Johanson, Morgan Kan, David Schnizlein, and Michael H Bowling. 2009. A practical use of imperfect recall. In *SARA*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35.

Yadong Zhang, Xinshu Shen, Yupei Ren, Shangqing Zhao, and Man Lan. 2026. Generative gamer: Learning equilibrium strategy by llm-driven dynamic deduction. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12604–12617.

Richard Zhuang, Akshat Gupta, Richard Yang, Aniket Rahane, Zhengyu Li, and Gopala Anumanchipalli. 2025. PokerBench: Training large language models to become professional poker players. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26175–26182.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. 2007. Regret minimization in games with incomplete information. *Advances in Neural Information Processing Systems*, 20.

A Game Descriptions

This appendix provides the complete game descriptions used as input to the Abstraction Agent. These descriptions constitute the *only* domain knowledge provided to the system; no game-specific evaluation code, hand evaluators, or equity calculators are used.

Heads-Up No-Limit Texas Hold'em (HUNL). HUNL is played with a standard 52-card deck of 13 ranks ($2 < 3 < \cdots < K < A$) and 4 suits ($\spadesuit, \heartsuit, \diamondsuit, \clubsuit$). Each player receives 2 private (hole) cards, and five community cards are dealt face-up in stages: flop (3), turn (1), river (1). At showdown, each player forms the best 5-card combination from their 2 hole cards and 5 community cards (best 5 of 7). Hands are ranked, from strongest to weakest: straight flush (five consecutive ranks, all same suit), four of a kind, full house (three of one rank plus two of another), flush (five of one suit), straight (five consecutive ranks, mixed suits), three of a kind, two pair, one pair, and high card. Betting is no-limit: players may bet any amount up to their remaining stack, with a betting round after each community card stage. In our turn experiments, 4 community cards are visible and one more will be revealed, so hand rankings *can change* with the final card: draws may complete, and currently strong hands may be overtaken.

Pot-Limit Omaha (PLO4). PLO4 uses the same 52-card deck, but each player receives **4 private cards** (compared to 2 in HUNL), with the same five community cards dealt in stages. The critical rule is that at showdown each player **must use exactly 2** of their 4 private cards combined with **exactly 3** of the 5 community cards; players cannot use 1, 3, or 4 private cards, which is the fundamental difference from HUNL. Hand rankings follow the same 9-category hierarchy (straight flush through high card). Betting is pot-limit: the maximum bet is the current pot size. In our PLO4 experiments, no community cards have been dealt, so hand value is entirely about *post-flop potential*. The hand space is $\binom{52}{4} = 270{,}725$ raw combinations, reduced to 16,432 canonical representatives after suit isomorphism.

Riichi Mahjong (Japanese Competitive Mahjong). Riichi Mahjong is played with 136 tiles: 34 unique types $\times$ 4 copies, comprising three numbered suits of 1–9 (Man/Characters, Pin/Circles, Sou/Bamboo) plus honor tiles (4 winds and 3 dragons). A complete hand (14 tiles) consists of 4 sets plus 1 pair, where each set is either a sequence (three consecutive tiles of one suit) or a triplet (three identical tiles); Seven Pairs and Thirteen Orphans are special patterns. Progress toward completion is measured by the *shanten* number, the minimum tile exchanges to reach *tenpai* (shanten 0, one tile from winning). Each turn a player draws one tile randomly from the remaining wall ($\sim$70 tiles initially, $\sim$40–50 mid-game); the draw may complete the hand, reduce shanten, enable higher-value scoring patterns (yaku), or provide no improvement, and it is the primary source of randomness after the initial deal. Hand value depends on yaku such as Riichi, Tanyao, Pinfu, Honitsu, and Chinitsu; more han means exponentially higher payment, with mangan (5+ han) worth 8000 points. Strategically, a hand is characterized by progression speed (shanten plus acceptance width), value potential (yaku routes), tile efficiency, defensive safety (safe discard

availability), and waiting pattern quality. We use the `mahjong` Python library (MahjongRepository, 2024) for shanten calculation and hand evaluation.

ROVER Trials. ROVER Trials is an original two-player zero-sum betting game of hidden information designed for this paper. It appears in no pretraining corpus, so the LLM cannot retrieve memorized strategy knowledge from it. The agent receives only the rules text below, with no feature hints and no evaluator. Each player is privately and independently assigned one *rover*, described by three integer dials each in $\{0, \ldots, 4\}$: POWER (raw engine strength), GRIP (traction on rough ground), and AFFINITY (the single terrain type the rover is specialized for), giving $5^3 = 125$ possible rovers; both players draw uniformly and independently, so they may even hold the same rover. Play proceeds as follows. Both players ante one chip. In betting round 1, before the terrain is known, the first player may check or raise (+2 chips); facing a raise a player may fold, call, or re-raise, with at most 2 raises per round. Then the *race terrain* is revealed publicly: one of five terrain types, each equally likely; sand and grass are *smooth*, while rock/mud/ice are *rough*. Betting round 2 proceeds after the terrain is revealed (raise size +4 chips) with the same betting rules. If neither player folds, both rovers race and the one that travels farther wins the pot; equal distance splits it. A rover travels

$$\begin{aligned}\text{distance} = \textsc{Power} + \textsc{Grip} \cdot \mathbb{1}[\text{terrain rough}] \\ + b \cdot \mathbb{1}[\textsc{Affinity} = \text{terrain}],\end{aligned}$$

where the affinity bonus is set to $b = 10$ in our experiments. A rover is therefore a "specialist" that is dominant on its own (one-in-five) terrain but ordinary elsewhere; expected distance alone cannot separate a volatile specialist from a steady rover of the same mean, which is precisely the regime in which a multi-feature abstraction can beat a scalar baseline. For evaluation we use $K \in \{5, 10, 20, 30\}$ buckets over 10 independent seeds per backbone (GPT-5.5, Llama-3.3-70B, Qwen-2.5-32B). Baselines mirror the poker setting: *Rank* buckets rovers by expected distance (the ROVER analog of EHS) and requires the payoff simulator; *PA* clusters the 2D potential-aware features $(\mathbb{E}[\text{distance}],\ \text{std}_{\text{terrain}})$; *Random* assigns buckets uniformly; and *full* is the unabstracted game. Solving uses CFR+ for 500 iterations; exploitability is computed exactly by best response on the unabstracted tree. The unabstracted full game reaches 0.0019 chips of exploitability, so the abstraction still incurs a cost; the relevant question is how much of the gap to evaluator-based abstractions a rules-only pipeline closes.

B PLO4 Preflop Hand Taxonomy

PLO4 preflop hands are characterized along multiple strategic dimensions; suit structure in particular determines flush potential. Table 5 and Table 6 provide a taxonomy.

Table 5: PLO4 suit pattern classification (canonical hand counts out of 16,432).

Pattern	Count	Description
Double-suited (2+2)	3,081	Two suited pairs, two flush draws.
Single-suited (2+1+1)	7,098	One suited pair; one flush draw.
Triple-suited (3+1)	3,718	Three of one suit, one weaker flush draw.
Rainbow (1+1+1+1)	1,820	All four suits differ; no flush potential.
Monotone (4+0)	715	Same suit; one flush draw (must-use-two).

Key PLO4 Concepts. Four concepts drive PLO4 preflop evaluation. *Nut potential*: because multiway pots are common, hands that can make the *nut* flush (holding the Ace of a suit) or nut straight are far more valuable than hands that can only make non-nut versions. *Card coordination*: all 4 cards should "work together," contributing to the same straights or flushes; a hand like $A\spadesuit K\spadesuit Q\heartsuit J\heartsuit$ (double-suited, connected) has maximum coordination, while $A\spadesuit 7\diamondsuit 3\clubsuit 2\heartsuit$ has a "dangler" (disconnected card) that reduces effective hand strength. *Domination risk*: hands whose flush or straight draws can only make non-nut hands may hit their draw and still lose to a higher version; small pairs without backup draws are particularly vulnerable. *The must-use-two rule*: using exactly 2 private cards fundamentally changes hand evaluation compared to HUNL; holding four cards of one suit does *not* help make a flush (only 2 can be used), and holding trips reduces pair potential (only 1 card of that rank remains in the deck).

C Prompt Templates

This section provides the exact prompt templates used by the Abstraction Agent. All prompts are

Table 6: PLO4 structural hand types with examples and strategic properties.

Type	Example	Strategic Properties
Premium pairs + suited	AA-KK ds	Top pair + flush draws; strongest
Top rundowns	AKQJ, KQJT	Max straight connectivity
Suited connectors	8h7h6s5s	Flush + straight; strong playability
High pairs + danglers	AA-xx rainbow	High pair, limited coordination
Mid rundowns	T987, 9876	Good connectivity, lower nut potential
Suited aces	As-xxx	Nut flush draw potential
Trips	AAA-x	Reduced pair potential; weak in PLO
Low connected	5432, 6543	Straight potential but rarely nut
Disconnected low	7h3d2c9s	No coordination; weakest

game-agnostic; domain knowledge enters only through the `GameConfig` fields.

Feature Discovery Prompt.

```
SYSTEM:
You are a game-theory analyst specializing in
imperfect-information games. Your task is to
propose continuous strategic features for
grouping private holdings.

USER:
## Game Description
{config.game_description}

## Task
Propose {M} continuous features in [0, 1] for
evaluating private holdings in this game.

## Requirements
1. SPREAD: Each feature must produce sufficient
   variance (std >= 0.15) across the hand space.
2. INDEPENDENCE: Features should capture different
   strategic dimensions (pairwise |r| < 0.7).
3. COMPLETENESS: Together, features should cover
   the major axes of strategic variation.
4. CALIBRATION: Provide anchors mapping 0.0, 0.5,
   and 1.0 to concrete game situations.
{5. FUTURE AWARENESS (if has_future_cards):
   Features should capture both current state AND
   potential for change when new information is
   revealed. Note: {future_card_note}}

## Anti-Patterns (DO NOT propose)
- Binary features (e.g., "has a pair: yes/no")
- Features that are subsets of another
- Features requiring external computation

## Output Format (JSON)
[{"name": "FEATURE_NAME",
  "description": "what it measures",
  "anchors": {"0.0": "...", "0.5": "...",
              "1.0": "..."}}]
```

Hand Scoring Prompt.

```
SYSTEM:
{config.game_context}

You will score private holdings on the following
features. Use the calibration anchors as reference
points for consistent scoring.

Features:
{for each feature f:
  - {f.name}: {f.description}
    0.0 = {f.anchors["0.0"]}
    0.5 = {f.anchors["0.5"]}
    1.0 = {f.anchors["1.0"]}}

USER:
{board_to_text(board)}

Score each holding below. Output format:
INDEX: FEAT1=0.xx FEAT2=0.yy ...

{for i, hand in enumerate(batch):
  {i}: {hand_to_text(hand, board)}}
```

Example Scoring Output. For a HUNL turn board $7\spadesuit\, 9\heartsuit\, 9\clubsuit\, T\clubsuit$:

```
0: CURRENT_MADE_STRENGTH=0.85
   IMPROVEMENT_POTENTIAL=0.30
   VULNERABILITY=0.45 NUT_CEILING=0.90
   BLOCKER_VALUE=0.60 RIVER_STABILITY=0.70
1: CURRENT_MADE_STRENGTH=0.20
   IMPROVEMENT_POTENTIAL=0.80
   VULNERABILITY=0.15 NUT_CEILING=0.40
   BLOCKER_VALUE=0.25 RIVER_STABILITY=0.20
...
```

D Experimental Details

Libratus Subgame Specifications. Table 7 provides details of the two Libratus-derived turn subgames used in our quantitative HUNL evaluation. Both are turn boards (4 community cards, 1 more to come), and hands are filtered by reach probability.

Table 7: Libratus subgame specifications.

ID	Board	Street	Pot
1	$7\spadesuit\, 9\heartsuit\, 9\clubsuit\, T\clubsuit$	Turn	500
2	$T\spadesuit\, 6\heartsuit\, A\heartsuit\, 7\clubsuit$	Turn	4780

Board Texture Analysis. Subgame 1 ($7\spadesuit\, 9\heartsuit\, 9\clubsuit\, T\clubsuit$) is a paired board with a club flush draw and open-ended straight draws; the pair creates full house potential, and the connected texture enables many draws. Subgame 2 ($T\spadesuit\, 6\heartsuit\, A\heartsuit\, 7\clubsuit$) is an ace-high board with a heart flush draw, where the disconnected low cards create a wide range of possible hand strengths.

CFR Solver Configuration.

- Algorithm: Discounted CFR (DCFR; $\alpha = 1.5$, $\beta = 0$, $\gamma = 2$) with alternating updates
- Iterations: 2,000
- Exploitability: Exact best-response computation over the unabstracted game tree
- Lifted exploitability: Strategy from abstract game mapped back to original game, then best-response computed

LLM API Configuration.

- Model: GPT-5.5 (primary), Llama-3.3-70B and Qwen-2.5-32B (comparison)
- Temperature: 0.7 (feature discovery), 0.0 (scoring, deterministic)
- Batch size: 20–30 hands per API call
- Retry logic: 3 attempts with exponential backoff
- Caching: All LLM responses cached by (model, prompt hash) for reproducibility

E Discovered Features by Game

Table 8 summarizes all six evaluation settings; exploitability is measured on HUNL turn and ROVER, and the remaining settings are evaluated by inspecting the discovered features and clusters. The rest of this section lists the features the agent discovers in the HUNL turn and PLO4 preflop settings, with their calibration anchors.

Table 8: Cross-game summary of the six evaluation settings.

Game	N	K	Key Result
HUNL Turn	$\sim$1K	20–100	up to 62% $\downarrow$ vs EHS
ROVER Trials	125	5–30	beats baseline
PLO4 Preflop	16,432	30	PLO features
HUNL Flop	1,176	20	Draw/made sep.
HUNL Preflop	169	10	Premium groups
Mahjong	200	10	Chance features

HUNL Turn Features. The following features are consistently discovered across turn subgames:

1. **CURRENT_MADE_STRENGTH** (retained in both subgames)
   Measures the current 5-card hand ranking relative to all possible holdings.
   Anchors: 0.0 = no pair/weak high card; 0.5 = top pair/overpair; 1.0 = set or better.
2. **IMPROVEMENT_POTENTIAL** (retained in both subgames)
   Probability of improving to a significantly stronger hand with the river card.
   Anchors: 0.0 = no draws; 0.5 = gutshot or weak draw; 1.0 = open-ended straight + flush draw.
3. **VULNERABILITY_TO_OUTDRAWS** (retained in both subgames)
   Risk that currently strong hands will be overtaken by opponent draws.
   Anchors: 0.0 = invulnerable (nuts); 0.5 = moderate risk; 1.0 = highly vulnerable to multiple draws.
4. **NUT_CEILING** (retained in both subgames)
   How close the hand is to the best possible hand on this board.
   Anchors: 0.0 = far from nuts; 0.5 = second/third nuts; 1.0 = current nuts.
5. **BLOCKER_VALUE** (retained in both subgames)
   Whether the hand blocks opponent's strong holdings or draws.
   Anchors: 0.0 = no blocking; 0.5 = blocks one draw; 1.0 = blocks multiple strong hands.
6. **RIVER_STABILITY** (retained in 1/2 subgames)
   How stable the hand's relative strength is across possible river cards.
   Anchors: 0.0 = highly volatile; 0.5 = moderate stability; 1.0 = rank unchanged by any river.
7. **RIVER_EQUITY** (sometimes filtered)
   Expected equity after the river card is revealed.
   Often filtered due to high correlation with CURRENT_MADE_STRENGTH.

PLO4 Preflop Features.

1. **RANK_AND_PAIR_STRENGTH** (retained)
   High-card value and pair potential of the 4-card holding.
   Anchors: 0.0 = low unpaired disconnected; 0.5 = medium pair or high-card; 1.0 = AA/KK coordinated.

2. **SUITEDNESS_AND_ FLUSH_POTENTIAL** (retained)
   Suit structure and nut flush draw potential.
   Anchors: 0.0 = rainbow; 0.5 = single-suited; 1.0 = double-suited, ideally ace-high.

3. **STRAIGHT_CONNECTIVITY** (retained)
   How many different straights the hand can make.
   Anchors: 0.0 = widely separated ranks; 0.5 = one or two gaps; 1.0 = 4-card rundown.

4. **ALL_CARD_COORDINATION** (retained)
   Degree to which all 4 cards work together.
   Anchors: 0.0 = severe danglers; 0.5 = 3 coordinated; 1.0 = all 4 coordinated.

5. **NUT_POTENTIAL** (retained)
   How often the holding makes the best (nut) version of strong hands rather than dominated second-best hands.
   Anchors: 0.0 = mostly weak or dominated draws; 0.5 = some nut-relevant components; 1.0 = excellent nut-making capacity.

FLOP_VALUE_SHIFT_POTENTIAL and DOMINATION_RISK_RESISTANCE were also discovered but dropped by the correlation filter ($|r| = 0.91$ with ALL_CARD_COORDINATION and $|r| = 0.97$ with NUT_POTENTIAL, respectively).

F HUNL Preflop Cluster Details

Table 9 shows the complete clustering of 169 canonical HUNL preflop hands into $K = 10$ clusters. The agent discovers 7 features, all retained after filtering: HIGH_CARD_POWER, PAIR_AND_SET_VALUE, SUITED_FLUSH_POTENTIAL, STRAIGHT_CONNECTIVITY, DOMINATION_RESILIENCE, EQUITY_REALIZATION_STABILITY, and RUNOUT_VOLATILITY. Parse rate is 100%. The clusters separate premium pairs, suited connectors, offsuit broadways, and junk hands without any poker-specific code.

The clustering shows strategic coherence: Cluster 8 isolates the strongest starting hands (premium pairs and AKo), Cluster 5 groups premium suited broadways with flush and straight potential, and Cluster 0 groups all small-to-medium pairs together (sharing similar set-mining value). The agent correctly separates suited from offsuit hands (Clusters 1/4/6 vs 7/9/2), reflecting the importance of flush potential in preflop hand evaluation.

G Feature–EHS Correlation

Table 10 reports the Pearson correlation between each discovered feature and EHS on the two turn subgames. The made-strength feature correlates strongly with EHS ($r \approx 0.87$–0.93), confirming that the agent recovers the standard strength signal, while the improvement- and river-shift features have low or negative correlation and thus capture strategic dimensions that a scalar EHS ordering cannot represent.

H PLO4 Preflop Cluster Examples

Tables 11 and 12 show the complete clustering of the 16,432 canonical PLO4 preflop hands into $K = 30$ clusters; the clusters separate hands by rank strength, suitedness, connectivity, and coordination without poker-specific code. Since PLO4 preflop hands are suit-isomorphic (only the *pattern* of shared suits matters, not which specific suits are held), we use abstract suit labels a, b, c, d assigned by first-appearance order. The five canonical suit patterns are: **ds** (double-suited, 2+2): two pairs share suits, e.g., $\mathrm{A}_a\mathrm{K}_a\mathrm{Q}_b\mathrm{J}_b$; **ss** (single-suited, 2+1+1): one pair shares a suit, e.g., $\mathrm{A}_a\mathrm{K}_a\mathrm{Q}_b\mathrm{J}_c$; **r** (rainbow, 1+1+1+1): all four suits differ, e.g., $\mathrm{A}_a\mathrm{K}_b\mathrm{Q}_c\mathrm{J}_d$; **mono** (monotone, 4+0): all same suit, e.g., $\mathrm{A}_a\mathrm{K}_a\mathrm{Q}_a\mathrm{J}_a$; **ts** (triple-suited, 3+1): three share a suit, e.g., $\mathrm{A}_a\mathrm{K}_a\mathrm{Q}_a\mathrm{J}_b$.

Cluster 17 contains exclusively rainbow holdings with premium pair strength (trips/quads such as AAA+ and high pairs), while Cluster 10 groups low-card suited rundowns with maximum straight connectivity and near-zero high-card content, demonstrating that the LLM-discovered features capture the multi-dimensional nature of PLO hand evaluation. Cluster 25 (top rundowns like AKQJ) and Cluster 10 (low rundowns like 9876) are cleanly separated despite both having very high straight connectivity, differing along the rank-strength and nut-potential dimensions.

I HUNL Flop Cluster Examples

Table 13 shows the complete clustering of the 1,176 flop holdings on the board A♠ K♡ Q♢ into $K = 20$ clusters. The agent cleanly separates made-strength tiers: sets in Cluster 5, the nut Broadway straight in Cluster 1, top two pair in Cluster 10, and top-pair hands split by kicker strength; draw-oriented holdings and no-interaction air form their own groups.

Table 9: Complete HUNL preflop clusters ($K = 10$, 169 canonical hands, sorted by ID).

Cluster	Representative Hands	Characteristics
0 (8)	22, 33, 44, 55, 66, 77, 88, 99	Small/medium pairs (set-mining value)
1 (19)	A2s–A5s, 54s–76s	Suited aces + suited connectors
2 (19)	32o–T2o	Offsuit junk (weakest hands)
3 (12)	QTo, KTo, ATo, QJo, KJo, AJo	Offsuit broadways
4 (25)	K2s–K4s, Q2s–Q4s, T4s–J4s	Suited high + low kicker
5 (10)	ATs, QJs, KJs, AJs, KQs	Premium suited broadways
6 (23)	32s–92s, T2s	Suited junk (low cards)
7 (18)	A3o–A5o, 54o–76o	Offsuit aces + connectors
8 (6)	TT, JJ, QQ, KK, AKo, AA	Premium pairs + AKo
9 (29)	J2o–K3o, A2o	Offsuit high + low kicker

Table 10: Pearson correlation between LLM features and EHS across turn subgames.

Feature	SG1	SG2
Current_Made_Str.	0.87	0.93
Improvement_Pot.	0.27	−0.30
Vuln._To_Outdraws	0.68	0.31
Nut_Ceiling	0.68	0.14
River_Stability	−0.36	0.52
Blocker_Value	0.65	0.46

J Prompt-Variant and Rules-Only Robustness

Table 14 reports the mean exploitability ratio (Agent / EHS, 10 seeds, GPT-5.5) for three feature-discovery prompt variants: *rules-only* (the bare game rules), *range-aware* (rules plus the public range summary; used in the main results), and *feature-discovery* (range-aware plus an explicit instruction to separate made strength, draws, blockers, and range interaction). At $K \in \{20, 50\}$ the three variants agree within the per-seed standard deviation (0.10–0.35), so the reported gains are not an artifact of a single prompt wording; in particular, the rules-only variant matches the range-aware one, indicating that the public range summary is not the source of the advantage.

K Feature Ablation Results

Table 15 reports leave-one-out feature importance, measured as the mean adjusted Rand index (ARI) between the clustering with and without each feature: a lower ARI means removing the feature changes the clustering more. Values come from a representative rules-only run, averaged across all subgame/K conditions where the feature is selected. VULNERABILITY_TO_OUTDRAWS and BLOCKER_VALUE carry the most unique clustering signal, while RIVER_EQUITY is largely redundant, consistent with its frequent removal by the correlation filter.

L Riichi Mahjong Cluster Details

Table 16 shows the features discovered by the agent for Riichi Mahjong mid-game clustering: 6 discovered, 5 selected after correlation filtering. Table 17 shows representative clusters (100% parse rate); the clusters separate hands by strategic role: offensive speed vs. defensive resilience vs. value potential.

M Additional Implementation and Analysis Details

Game-Agnostic Design. The pipeline's game-agnosticism comes from a `GameConfig` interface specifying the game rules (for feature discovery), a shorter context (for scoring prompts), and two rendering callbacks (`hand_to_text`, `board_to_text`). To apply the agent to a new game, one writes a `GameConfig` (typically 20–50 lines of natural language) and two rendering functions. No evaluation code, rollout simulators, or domain-specific feature engineering is required.

Prompt Design Principles. Three design choices underlie the pipeline's output reliability: (1) *Calibration anchors* map 0.0/0.5/1.0 to concrete game situations, grounding the LLM in consistent reference points across batches; (2) *Constraint specification* enforces SPREAD, INDEPENDENCE, and COMPLETENESS as explicit requirements for feature discovery; (3) *Batch formatting* uses numbered lists with fixed format ("`0: FEAT1=0.xx FEAT2=0.yy`"), enabling reliable parsing. The calibration anchors serve a dual role: they normalize scores across batches (analogous to few-shot examples) and provide interpretable semantics for each feature's scale.

Feature Families Across Games. Because feature discovery samples at temperature 0.7,

Table 11: Complete PLO4 preflop clusters ($K = 30$, 16,432 canonical hands; abstract suit labels a–d).

Cluster	Representative Hands	Characteristics
0 (590)	$9_a7_b5_a5_b$ (ds), $6_a6_b5_a3_b$ (ds)	Low-rank suited coordination holdings
1 (505)	$A_aJ_bJ_c7_b$ (ss), $A_aJ_bJ_c6_b$ (ss)	High-pair heavy
2 (629)	$K_aT_b9_a5_b$ (ds), $A_aT_a5_b2_b$ (ds)	Low-rank suited coordination holdings
3 (324)	$A_aA_bK_c5_c$ (ss), $A_aA_b9_c7_c$ (ss)	Premium paired / AA-heavy holdings
4 (743)	$Q_a9_a6_b3_b$ (ds), $J_aT_b7_a3_a$ (ts)	Low-rank suited coordination holdings
5 (468)	$Q_aT_b9_c6_a$ (ss), $Q_aJ_b8_a8_c$ (ss)	Mixed suited coordination holdings
6 (168)	$A_aA_b5_a4_b$ (ds), $A_aA_bJ_aJ_b$ (ds)	Premium paired / AA-heavy holdings
7 (194)	$K_aJ_b9_c9_d$ (r), $A_aK_bT_c4_d$ (r)	Rainbow high-card holdings
8 (702)	$A_a9_b7_c4_a$ (ss), $A_aT_bT_c4_a$ (ss)	Mixed suited coordination holdings
9 (482)	$T_aT_bT_c2_d$ (r), $K_aQ_b4_c2_d$ (r)	Medium-pair heavy
10 (314)	$5_a4_a4_b3_a$ (ts), $7_a6_b4_c3_a$ (ss)	Suited connected rundown holdings
11 (436)	$J_a8_b8_c2_d$ (r), $Q_a6_b6_c4_d$ (r)	Low-pair heavy
12 (496)	$7_a6_a4_b2_a$ (ts), $9_a8_a5_b4_a$ (ts)	Low-rank suited coordination holdings
13 (685)	$Q_aT_b7_b6_b$ (ts), $Q_aJ_a7_a5_b$ (ts)	Low-rank suited coordination holdings
14 (401)	$A_a5_a5_b4_b$ (ds), $J_aT_b9_b5_a$ (ds)	Mixed suited coordination holdings
15 (889)	$Q_a9_a9_b9_c$ (ss), $A_aJ_b7_b5_c$ (ss)	Unpaired

Table 12: PLO4 preflop clusters (continued).

Cluster	Representative Hands	Characteristics
16 (133)	$J_a9_b7_c5_d$ (r), $Q_aT_b9_c7_d$ (r)	Rainbow low-to-mid coordination holdings
17 (178)	$K_aK_bT_c4_d$ (r), $A_aA_bQ_c7_d$ (r)	Premium paired / AA-heavy holdings
18 (624)	$Q_aJ_a8_b2_a$ (ts), $5_a5_b2_a2_b$ (ds)	Low-rank suited coordination holdings
19 (387)	$7_a5_b2_c2_d$ (r), $A_a7_b3_c2_d$ (r)	Rainbow low-to-mid coordination holdings
20 (896)	$9_a7_a2_a2_b$ (ts), $Q_a7_a7_b2_a$ (ts)	Low-pair heavy
21 (618)	$A_aK_b9_a9_b$ (ds), $K_aQ_bT_b2_a$ (ds)	Mixed suited coordination holdings
22 (661)	$A_aK_a4_b3_a$ (ts), $A_aK_bT_b5_b$ (ts)	Suited high-card coordination holdings
23 (981)	$9_a8_b5_b3_c$ (ss), $Q_a9_a8_b2_a$ (ts)	Low-rank suited coordination holdings
24 (275)	$9_a8_a6_b5_b$ (ds), $T_a9_a8_a6_b$ (ts)	Suited connected rundown holdings
25 (213)	$A_aK_bQ_aT_a$ (ts), $A_aK_bJ_bT_b$ (ts)	Suited connected rundown holdings
26 (1163)	$J_a8_a6_a3_b$ (ts), $A_a9_b7_b2_c$ (ss)	Low-rank suited coordination holdings
27 (354)	$A_aK_bJ_b4_a$ (ds), $A_aQ_bJ_b9_b$ (ts)	Suited high-card coordination holdings
28 (826)	$7_a4_b3_a3_c$ (ss), $T_a6_b4_c3_c$ (ss)	Low-rank suited coordination holdings
29 (1097)	$Q_aQ_bT_c7_a$ (ss), $K_aJ_b9_c3_c$ (ss)	Mixed suited coordination holdings

the LLM instantiates its features with different names on each run, but the underlying strategic axes are stable. Across seeds and both subgames, the recurring feature families are: current made-hand strength (e.g., CURRENT_MADE_STRENGTH, the family most correlated with EHS at $r \approx 0.87$–0.93), nut potential (NUT_CEILING and variants), blocker effects (BLOCKER_LEVERAGE/BLOCKER_DENIAL), river volatility and downside exposure, and domination / reverse-implied-odds risk. The made-strength family aligns with EHS, while the volatility, blocker, and domination families have substantially weaker EHS correlation (Table 10), which is what lets the multi-dimensional abstraction separate hands that EHS would merge. After correlation filtering, 4–8 features survive per run (median 6).

Parse Rate and Robustness. Across all experiments (HUNL turn/flop/preflop, PLO4, and Riichi Mahjong), the overall parse rate exceeds 99% (100% on turn, preflop, PLO4, and Mahjong; 97.9% on flop). The turn results are also robust to the exact discovery prompt: three prompt variants (rules-only, range-aware, and an explicit feature-discovery instruction) yield mean ratios that agree within the per-seed standard deviation at $K \in \{20, 50\}$ (Appendix J), so the reported gains are not an artifact of a single prompt wording.

N Future Work

Several directions follow directly from this work. First, characterizing the dependence on the LLM backbone: a frontier model is currently needed to beat EHS, and identifying which capabilities drive this gap would clarify when the pipeline transfers to weaker or specialized models. Second, a cheap proxy for abstraction quality that avoids solving the game would close the loop on oracle-free feature-set selection. Third, solver-in-the-loop refinement

Table 13: Complete HUNL flop clusters (board A♠ K♡ Q♢; $K = 20$, 1,176 hands, sorted by ID).

Cluster	Representative Hands	Characteristics
0 (80)	Ad 9s, Ad 9d, Ad 8c	Top pair A, strong kicker
1 (16)	Js Ts, Js Td, Jd Tc	Nut straight (Broadway)
2 (162)	9s 4d, 9s 3c, 9s 2h	Air (9x low kicker, no draw)
3 (138)	Ah 5h, Ah 4h, Ks 9s	Medium made + backdoor draws
4 (25)	9d 9c, 8s 8h, 6s 6d	Underpairs (below board)
5 (8)	Ad Ac, Ks Kd, Qs Qc	Sets (AA/KK/QQ)
6 (86)	8c 7s, 8c 6d, 8c 5c	Junk (8x and below, no interaction)
7 (29)	Td 5d, Td 4s, Td 3c	T-kicker, weak draws
8 (153)	Ks 9d, Ks 8c, Ks 7d	Second pair K, weak kicker
9 (33)	Ks 4d, Ks 3s, Ks 2d	Second pair K, very weak kicker
10 (21)	Ad Ks, Ad Qh, Ac Kd	Top two pair (AK, AQ)
11 (35)	Ac 9s, Ac 9d, Ac 8c	Top pair A + backdoor nut flush
12 (23)	9s 9d, 9s 9c, 8s 8d	Medium pocket pairs (multi-suit)
13 (39)	8d 6s, 8d 5c, 8d 4s	Air + faint backdoor diamond
14 (43)	Jc 9s, Jc 8d, Jc 7c	J-kicker, straight potential
15 (35)	Ks Js, Ks Jd, Ks Ts	Second pair K + strong kicker J/T
16 (161)	9s 8s, 9s 7d, 9s 6c	Largest junk group (below T)
17 (23)	Ts 3c, Ts 2d, Td 9s	T-kicker mixed, bottom straight draw
18 (27)	Ad Js, Ad Td, Ad Tc	Top pair A + strong kicker J/T (diamond)
19 (39)	Jd 7d, Jd 6s, Jd 5c	J-kicker + weak backdoor

Table 14: Prompt-variant robustness: mean Agent / EHS exploitability ratio over 10 seeds (GPT-5.5).

Variant	K=20		K=50	
	SG1	SG2	SG1	SG2
Rules-only	1.20	1.21	0.73	0.91
Range-aware	1.30	1.25	0.74	0.82
Feature-discovery	1.35	1.18	0.76	0.92

Table 15: Feature importance via leave-one-out ablation (mean ARI; lower = larger impact).

Feature Removed	Mean ARI
Vulnerability_To_Outdraws	0.7558
Blocker_Value	0.7607
Current_Made_Strength	0.7638
Nut_Ceiling	0.7716
Improvement_Potential	0.7866
River_Stability	0.7959
River_Equity	0.8122

Table 16: Features discovered for Riichi Mahjong midgame (turn 8, East round).

Feature	Description
CURRENT_SPEED	Shanten + effective tiles
ACCEPTANCE_BREADTH	Tile types improving hand
HAND_VALUE_POTENTIAL	Scoring upside (yaku, dora)
WAIT_QUALITY	Quality of the winning wait
DEFENSIVE_RESILIENCE	Avoid dealing into opponents
NEXT_DRAW_SWING	How one draw changes value (not selected)

could use exploitability feedback to revise the discovered features online, and a principled fallback for unparsable scores would replace the current midpoint imputation.

More broadly, the pipeline instantiates a general recipe for rule-grounded LLM scoring: given a natural-language specification of evaluation criteria and calibration anchors, an LLM scores arbitrary objects on continuous dimensions, and downstream objective signals, here exploitability, verify the quality of those scores. This suggests two wider uses. First, LLM-as-a-judge: the recipe turns any domain whose criteria can be written down into a structured scorer without human preference data, and the verifiable-calibration view offers an alternative to agreement-with-human-labels as the quality signal for such judges. Second, LLM evaluation: the rules-to-features-to-clusters-to-exploitability chain is an automatic, zero-annotation benchmark of an LLM's ability to turn a specification into sound numerical knowledge, complementing game-playing benchmarks that probe decision quality (e.g., PokerBench (Zhuang et al., 2025)) with a direct test of structured knowledge elicitation. Beyond scoring and evaluation, feature discovery is a step toward automated feature engineering, and the game-agnostic design transfers naturally to security games, auctions, and negotiation.

Table 17: Complete Riichi Mahjong clusters ($K = 10$, 200 hands).

Size	Representative Hands	Dominant Trait
27	11578m 156p 13347s GR	Wide acceptance, 3–4 shanten
20	112467899m 4p 16s W	High speed + value
22	11149m 5579p 668s WH	Slow, high defense
16	1129m 39p 4777s WGR	High value, slow
23	1136m 2578p 1455s S	Average hands
33	1119m 1p 157889s WR	Most defensive
14	11378m 5688p 2s GRR	Most value potential
23	1239m 1245p 2667s W	Straight-type acceptance
15	12467m 489p 12339s	Balanced moderate
7	12456m 356p 56s WWR	Fastest, wide waits