FROM INTUITION TO DELIBERATION: SELF-EVOLVING REASONING FOR STRATEGIC CARD GAMES

Anonymous authors
Paper under double-blind review

ABSTRACT

Strategic card games, characterized by partial observability and dynamic adversarial interactions, serve as critical benchmarks for evaluating the decision-making intelligence of Large Language Models (LLMs). Most LLM-based approaches prioritize policy performance while overlooking explicit reasoning generation. Achieving transparent decision-making typically relies on high-quality Chain-of-Thought (CoT) data; however, such annotations are scarce and costly in strategic card games, whereas prompt-generated rationales are often ungrounded or unreliable. Moreover, incorporating explicit reasoning into discrete decision-making tasks may incur generative overhead and risk performance degradation. To address these challenges, we propose **Metis**, a self-evolving framework designed to harmonize competitive proficiency with explicit strategic reasoning. Metis comprises three synergistic components: mixed Supervised Fine-Tuning (SFT), structured-reward Reinforcement Learning (RL), and an iterative self-distillation loop. Specifically, mixed SFT alongside structured-reward RL bootstraps initial policy-reasoning alignment and mitigates sparse supervision in discrete action spaces, while the iterative self-distillation loop anchors reasoning logic to high-confidence strategic actions, transforming autonomous expert-level decisions into reliable supervision signals. Extensive experiments on GuanDan and DouDiZhu demonstrate that Metis achieves a synergistic enhancement of competitive card-playing proficiency and explicit strategic reasoning. Our code is available at `https://anonymous.4open.science/r/Metis`.

1 INTRODUCTION

Games, particularly strategy games with well-defined rule sets and long-term adversarial dynamics, have long served as critical benchmarks for evaluating agent capabilities (ALLIS, 1994; Silver et al., 2016; Schrittwieser et al., 2020). Unlike open-domain text tasks, game environments offer a more discernible state-action-reward loop, facilitating the assessment of models in areas such as strategic planning, situational judgment, opponent modeling, and long-term reward optimization (Silver et al., 2016; Schrittwieser et al., 2020; Duan et al., 2024; Light et al., 2024). In recent years, Large Language Models (LLMs) have achieved remarkable success across a wide range of domains, including commonsense reasoning, code generation, and complex problem-solving, owing to their powerful text understanding and generation capabilities (Vaswani et al., 2017; Brown et al., 2020; Yao et al., 2023; Shinn et al., 2023). Given their demonstrated proficiency in language-driven reasoning, it becomes imperative to **evaluate whether LLMs can extend such capabilities to more dynamic and adversarial environments—namely, strategic games.** Consequently, LLM-based Game Agents (LLMGAs) have emerged as a promising research frontier (Hu et al., 2024; Lin et al., 2025), with card games representing a particularly compelling testbed. The inherent partial observability, discrete action constraints, opponent strategy uncertainty, and multi-round game structures in card games pose unique challenges that demand more than raw language understanding; they require strategic deliberation, long-term planning, and adaptive reasoning under incomplete information. These characteristics make card games an ideal yet demanding proving ground for assessing the true strategic limits of LLMs.

Current research on applying LLMs to game decision-making can be broadly categorized into three paradigms: prompt-based zero-shot or few-shot methods (Paglieri et al., 2025; Ruoss et al., 2024), behavior imitation via Supervised Fine-Tuning (SFT) (Wei et al., 2021; Ouyang et al., 2022), and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

policy optimization approaches incorporating Reinforcement Learning (RL) (Schulman et al., 2017; Rafailov et al., 2023; Shao et al., 2024). Despite their progress, these methods face notable challenges when applied to complex card games. Prompt-based approaches, while easy to deploy, are highly sensitive to prompt design and context construction, often resulting in unstable performance and limited generalization in complex decision scenarios (Isaza-Giraldo et al., 2024; Ramesh et al., 2025). In contrast, SFT and RL-based methods can substantially improve win rates; however, they primarily optimize action correctness and reward signals, with limited modeling of the underlying decision rationale (Wang et al., 2025b; Zhang & Lu, 2024). As a result, although strong gameplay performance can be achieved, the internal reasoning process remains largely implicit, hindering decision transparency and systematic analysis of the reasoning process (Wang et al., 2025b).

Consequently, most existing card game agents strictly prioritize win-rate optimization, neglecting explicit strategic reasoning. Integrating human-understandable rationales into decision-making typically relies on high-quality Chain-of-Thought (CoT) supervision to align reasoning steps with actions (Wei et al., 2022). However, such annotations are scarce and costly in complex card games, whereas zero-shot or prompt-generated reasoning remains noisy and unstable. Furthermore, enforcing natural-language generation in non-linguistic control tasks introduces computational overhead and optimization conflicts, presenting a trade-off between reasoning expressiveness and decision performance. **Thus, ensuring competitive performance and decision transparency simultaneously remains a significant challenge.**

To bridge this gap, we propose **Metis**—named after the Greek goddess of strategic wisdom and adaptive cunning—a self-evolving framework to harmonize competitive proficiency with natural-language reasoning traces. Metis operates through a synergistic three-stage pipeline: First, we employ mixed SFT to bootstrap the initial alignment between strategic reasoning and policy execution, providing a robust foundation for joint optimization. Building upon this, we employ a structured reward-enhanced Group Relative Policy Optimization (GRPO) (Shao et al., 2024) algorithm to encourage action accuracy and logical consistency while mitigating sparse supervision in discrete action spaces. Finally, an iterative self-distillation loop anchors reasoning logic to high-confidence trajectories by transforming expert-level decisions into CoT supervision signals. In our experiments, this integrated design yields competitive strategic performance and explicit reasoning traces. The main contributions of this study are summarized as follows:

- We propose **Metis**, an end-to-end self-evolving framework that harmonizes competitive play with explicit natural-language reasoning traces in strategic card games.
- We design a synergistic multi-stage architecture: mixed SFT provides dual policy-reasoning initialization, structured-reward GRPO optimizes action and format targets under sparse supervision, and an internal self-distillation loop component continuously feeds verified trajectories back into the system to drive self-evolution.
- Experiments on GuanDan and DouDizhu demonstrate its effectiveness in the two evaluated games. Furthermore, we analyze the role of reasoning traces in these settings and observe an association between performance and distilled data quality, providing empirical guidance for integrating CoT into game-playing agents.

2 Related Works

Card Game Model. Card games have long served as benchmarks for evaluating strategic decision-making in AI research. Initial approaches relied on Monte Carlo Tree Search (MCTS) (Kocsis & Szepesvári, 2006; Coulom, 2006) and self-play RL. In DouZero (Zha et al., 2021), the Deep Monte Carlo algorithm achieved superhuman performance in DouDizhu. Subsequently, DouRN (Chen et al., 2023) incorporated residual networks to further improve win rates. For GuanDan, landmark frameworks such as DanZero (Lu et al., 2023) and DanZero+ (Zhao et al., 2024) established the first reinforcement learning benchmarks by integrating MCTS with deep neural networks to navigate the game's vast combinatorial action space. Platforms such as RLCard (Zha et al., 2019) and OpenSpiel (Lanctot et al., 2019) have further advanced self-play training infrastructure for imperfect-information games.

LLMs for Games. As LLM capabilities advance, LLMs for Games have emerged as a significant research frontier (Hu et al., 2024). Researchers have systematically evaluated LLM performance

across various game environments (Wu et al., 2023; Schultz et al., 2024; Ma et al., 2024; Yuan et al., 2025; Zhang et al., 2025; Ma et al., 2026). Benchmarks such as GTBench (Duan et al., 2024) and LMGame-Bench (Hu et al., 2026) have revealed core challenges in game decision-making. In strategy-intensive card games, research focus has shifted from general assessment to scenario-specific optimization. Wang et al. (2025b) demonstrated the effectiveness of SFT using high-quality game trajectory data. The Mastermind framework (Wang et al., 2025a) achieved competitive performance in DouDizhu and Go through algorithmic data synthesis. Meanwhile, Yim et al. (2024) enhanced reasoning fluency and analytical depth in Guandan by integrating retrieval augmentation with theory of mind mechanisms.

Reinforcement Learning for LLMs. Reinforcement learning has emerged as a key paradigm for enhancing LLM reasoning and aligning with human intent. The field has undergone rapid evolution, from early RL from Human Feedback (RLHF) (Ouyang et al., 2022) and RL from AI Feedback (RLAIF) (Bai et al., 2022) to methods bypassing reward models like Direct Preference Optimization (DPO) (Rafailov et al., 2023). At the algorithmic level, Proximal Policy Optimization (PPO) (Schulman et al., 2017) has become the de facto standard due to its training stability. More recently, the field has witnessed a surge of advances building upon GRPO (Shao et al., 2024), including DAPO (Yu et al., 2025) and GSPO (Zheng et al., 2025), each introducing targeted improvements to training stability, sample efficiency, and exploration capability.

3 METHOD

3.1 PROBLEM FORMULATION

We formulate card-playing decision-making as a conditional policy learning problem. We define a game state s to describe the environment, including the player's hand, public action history, and contextual information. The state evolves dynamically over time. At each step, the model selects an action $a \in A(s)$ from a state-dependent legal action set. The learning objective of the model is to predict an optimal action under game constraints. Supervised training uses the ground-truth action a^* as the target, which is derived from expert gameplay trajectories.

With CoT supervision, each sample is extended to a triplet (s, c, a), where c denotes the intermediate reasoning process. This introduces an explicit intermediate variable linking the state and action. Accordingly, we consider two modeling paradigms: (1) direct policy modeling, $\pi_\theta(a \mid s)$, and (2) CoT-augmented policy modeling, $\pi_\theta(c, a \mid s)$, which jointly generates reasoning and action. The latter generates actions alongside natural-language rationales, affording surface-level interpretability. However, this joint generation risks interfering with policy optimization. Ultimately, this work investigates the trade-off between task performance and rationale generation.

3.2 OVERVIEW

To reconcile the competitive proficiency and strategic transparency required for complex card games, we propose Metis—a framework designed to cultivate deliberate strategic reasoning through an evolutionary process. By conditioning reasoning on high-confidence strategic actions, Metis encourages consistency between the generated CoT traces and selected actions. As illustrated in Figure 1, the framework consists of three key components: mixed SFT, structured GRPO, and a self-distillation loop paradigm.

Metis follows an iterative generate–filter–train procedure across self-distillation cycles. Given a pool of game states paired with reference actions, the model first generates candidate reasoning traces. These candidates undergo a quality filtering mechanism and are merged with action-only instances to form the augmented training set. Only after this data construction step is the model updated via mixed supervised fine-tuning (SFT) followed by structured-reward GRPO, with the newly updated checkpoint serving as the rationale generator for the next cycle.

This iterative self-distillation paradigm directly mitigates the scarcity of expert CoT supervision. Within each loop, candidate rationales are retained only when their predicted actions strictly align with reference choices, followed by an LLM-as-judge that screens for logical coherence. By feeding these validated, action-consistent samples back into subsequent training cycles, Metis continuously elevates both competitive proficiency and explicit rationale generation.

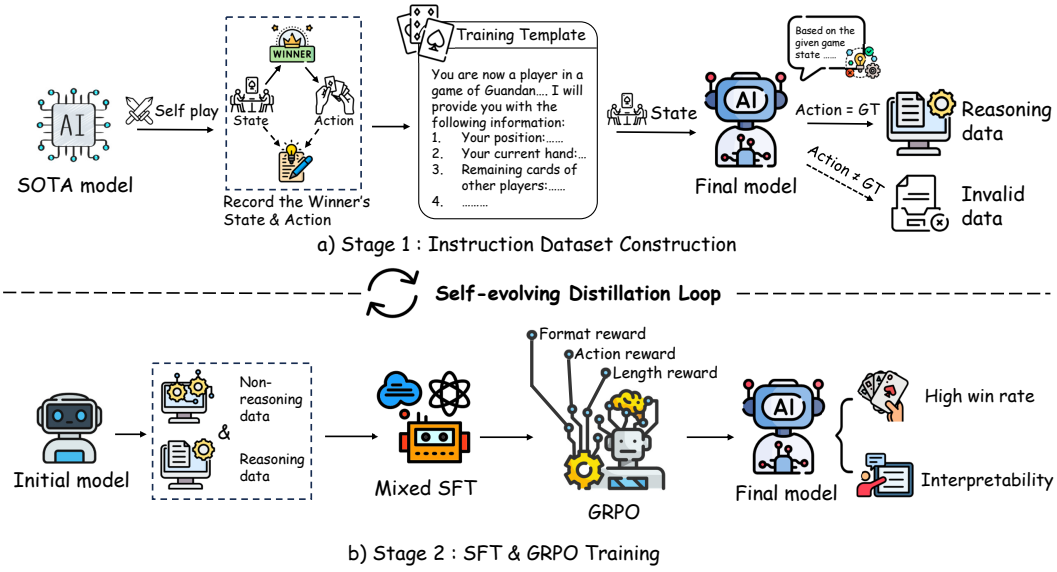


Figure 1: Overview of the Metis training framework

Overall, this framework combines policy learning, explicit rationale generation, and data refinement in an evolutionary process. We evaluate the resulting agent through win rate, action validity, and the availability of natural-language rationales.

3.3 Mixed SFT Training

CoT prompting has been widely adopted to enhance reasoning capabilities and decision transparency in LLMs. However, extending its efficacy to non-linguistic decision-making tasks, such as strategic card games, remains far from straightforward. Unlike symbolic reasoning problems that map neatly to formal logic, card game decisions hinge heavily on implicit strategy learning, belief-state distribution modeling, and real-time adaptation to opponents. Consequently, enforcing explicit linguistic reasoning may induce a mismatch between verbalized rationale and optimal policy execution.

To investigate this issue, we analyze CoT integration strategies in the two evaluated games (Section 4.3.1). In these experiments, configurations that generate full reasoning processes during inference obtain lower win rates and action validity than action-only training. Mixed supervision improves upon pure CoT supervision in the evaluated configurations, suggesting that CoT can be more useful as an auxiliary training signal than as a standalone generation target in these settings.

Motivated by these findings, we propose Mixed SFT, which jointly leverages two complementary supervision formats within a unified framework: (1) action-only samples (s, a^*), providing direct policy learning signals, and (2) reasoning-augmented samples (s, c, a^*), providing structured rationales as an auxiliary training signal. Mixed SFT is intended to reduce interference between the two supervision formats and initialize subsequent RL.

3.4 GRPO with Structured Reward

Following SFT to obtain a foundational decision-making policy, we further refine the model via RL, incorporating a CoT reasoning mechanism. We adopt GRPO as the RL algorithm, which eliminates the need for a separate critic network by computing advantages through intra-group comparisons. To encourage structured reasoning outputs and agreement with the final action, we design a multi-objective dense reward function tailored to game-theoretic contexts.

3.4.1 Multi-objective reward function

The reward function is engineered to provide fine-grained and discriminative feedback by optimizing two primary objectives: format compliance and action accuracy. While the underlying logic

remains unified across GuanDan and DouDizhu, minor task-specific adaptations are implemented to accommodate differences in action spaces and game rules. The overall reward R is defined as a weighted sum of these components, specifically $R = 0.9 \times R_{\text{acc}} + 0.1 \times R_{\text{fmt}}$, where the coefficients are balanced to prioritize strategic correctness while maintaining structural integrity.

The format consistency reward, R_{fmt}, encourages structured and parsable outputs, requiring paired reasoning tags followed by a valid JSON object containing an extractable action field. This binary signal assigns explicit feedback to parseable outputs during early RL.

The action accuracy component, R_{acc} is a hierarchical metric calculated by subtracting a length-based penalty from the raw action matching reward. The matching reward employs a soft-matching scheme to mitigate the sparse supervision challenge inherent in discrete action spaces: a full reward is assigned for an exact match with the ground truth, while partial credit is awarded for correctly identifying card types or key attributes. Additionally, a dedicated penalty is imposed on invalid "PASS" actions to discourage the model from converging toward degenerate passive strategies.

To regulate output length and discourage extreme verbosity or trivial shortcuts, we incorporate length-based regularization into R_{acc}. This includes an over-length penalty to discourage redundant or truncated outputs and an under-length penalty to discourage trivial shortcuts. Length regularization controls output length but does not by itself establish the analytical quality of a CoT trace. The overall reward is defined as:

$$R = 0.9 \times (R_{\text{action}} - R_{\text{length}}) + 0.1 \times R_{\text{fmt}} \tag{1}$$

where R_{action} is the action matching reward based on soft-matching, R_{length} is the length penalty, and R_{fmt} is the format reward. Overall, this hierarchical design provides denser feedback for policy optimization.

3.4.2 GRPO Policy Optimization

During the RL phase, we initialize the policy π_θ using the model checkpoint obtained from the Mixed SFT stage. For each sampled game state s, the policy generates a group of K candidate outputs, where each output $k \in \{1, \ldots, K\}$ consists of a reasoning CoT and its corresponding action, denoted as (c_k, a_k). We then evaluate each candidate using our structured multi-objective reward function to compute a composite reward value R_k. These rewards are utilized to derive the relative advantage estimates within each group, which subsequently drive the policy update via the GRPO objective. This approach reinforces higher-reward reasoning-action pairs. Detailed formulations of the GRPO objective and the specific optimization hyperparameters are provided in Appendix B.

3.5 Self-distillation Loop

To mitigate the scarcity of expert reasoning data, we propose a self-distillation loop paradigm. This approach converts action-validated decisions into filtered CoT supervision signals and is evaluated through strategic performance and explicit rationale generation. The detailed procedure of this self-distillation loop is described in Appendix A.2, with the full algorithm presented in Algorithm 1.

The process begins with the construction of a high-quality action foundation. We utilize state-of-the-art (SOTA) models to conduct extensive self-play, retaining winning trajectories as reference state-action pairs. To bootstrap initial reasoning capabilities, the model synthesizes preliminary CoT traces explicitly conditioned on both game states and their corresponding reference actions, ensuring that the initial rationales are inherently aligned with target decisions.

This initial dataset fuels a multi-stage optimization pipeline: Mixed SFT first balances policy stability with reasoning acquisition, followed by GRPO with structured rewards to refine decision quality and logical consistency. In subsequent self-distillation cycles, the model generates candidate reasoning-action pairs autonomously, given only the game states. To prevent error propagation, generated samples undergo two-tier filtering: a candidate is retained only if its predicted action strictly matches the SOTA reference action, followed by an LLM-based judge that verifies logical coherence before merging validated pairs into the training pool for the next iteration. Across self-distillation cycles, this iterative procedure consistently drives improvements in win rates and action validity, effectively aligning explicit strategic reasoning with high-performing game policies.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Data. We conducted multi-round competitive game play between SOTA models, using DouZero (Zha et al., 2021) for DouDizhu and DanZero (Lu et al., 2023) for GuanDan. For each round, we meticulously recorded the complete card-playing trajectory of the winner. We then engineered domain-specific prompts that used game rules, hand states, and contextual information. This helped us transform the "game state (input) - card-playing action (output)" relationships into a structured key-value format and construct a foundational action dataset. To construct a fine-tuning dataset for CoT reasoning, we introduced CoT prompting into the input and modified the output format: the model first generates its reasoning process, followed by a compliant JSON action. For comparison, we also provided the same state inputs to a rule-based model to generate action labels, creating an additional dataset to examine the effect of data source on training.

Model. In our experiments, we utilized Qwen2.5-7B-Instruct (Yang et al., 2024) as the foundational LLM. Training hyperparameters for our method are detailed in the Appendix B. For comparison, we evaluate small-scale open-source baselines including InternLM2.5-7B-Chat (Cai et al., 2024) and DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025), as well as large-scale closed-source baselines including GLM-4.5, DeepSeek-V3, GPT-4o, Claude-4-Sonnet and DeepSeek-V4-Pro. All models are evaluated under identical prompt templates and inference configurations.

Evaluation. We evaluate model performance using win rate and action validity rate, with distinct adversarial paradigms designed for DouDizhu and GuanDan. Opponents are of two types: (1) random models, which test basic decision-making ability, and (2) rule-based models, which assess strategic optimization. For GuanDan, we adopt Baseline 4 from DanZero (Lu et al., 2023) as the rule-based opponent; for DouDizhu, we use the rule-based model from RLCard (Zha et al., 2019).

In all experiments, we conduct 100 matches per configuration. GuanDan follows a four-player partnership format: the LLM controls one pair of partners, and the opponent model controls the other two. DouDizhu adopts a 1v2 format: the LLM plays as the "landlord," and the opponent model controls the two "peasants." In both settings, the side that first empties its hand wins. Action validity rate is calculated as the proportion of valid card play actions generated by the LLM, adhering to the respective game's meld rules, out of the total number of actions taken.

4.2 MAIN RESULTS

As shown in Table 1, we compare Metis against strong baselines and different training paradigms on the GuanDan and DouDizhu tasks. Overall, the performance gains co-occur with high action validity across the evaluated settings.

The model produces structured natural-language rationales paired with its actions during inference, while attaining high win rates against both random and rule-based opponents. In the evaluated settings, these results show that explicit rationale generation can coexist with competitive decision performance; they do not establish that the rationales faithfully reflect the model's internal computation.

Furthermore, our model surpasses the evaluated general-purpose LLMs, such as GPT-4o, under the prompt-based inference settings in Table 1. One possible explanation is a mismatch between general pretraining data and the requirements of the two rule-constrained card games, where legal-action constraints and implicit strategic information are important. Prompt engineering alone may therefore be insufficient in these particular environments. These results motivate domain-adaptive optimization in the evaluated settings, but do not establish transfer to decision-making tasks beyond them.

Compared with standalone SFT, vanilla GRPO, and a sequential SFT+GRPO baseline, Metis achieves superior win rates and action validity across all evaluated settings. Standard SFT remains constrained by the availability and quality of expert trajectories, whereas vanilla GRPO frequently exhibits optimization instability. Although a sequential SFT+GRPO pipeline yields incremental gains, its performance margin remains modest. In contrast, Metis integrates Mixed SFT and structured GRPO within an iterative self-distillation loop mechanism, yielding learning dynamics that promote stable convergence and sustained policy improvement.

Table 1: **Main Results: Performance comparison under the Reasoning-then-Action (RtA) paradigm.**

Model / Method	GuanDan			DouDizhu		
	Win rate (%)		Action Validity (%)	Win rate (%)		Action Validity (%)
	vs Rule	vs Random		vs Rule	vs Random	
Small-scale Models (under 10B)						
Internlm2.5-7b-chat	18%	51%	52.32%	5%	44%	71.70%
DeepSeek-R1-Distill-Qwen-7B	13%	56%	51.91%	3%	36%	78.80%
Qwen2.5-7B-Instruct	16%	46%	94.48%	4%	21%	90.90%
Large-scale Models						
GLM-4.5	32%	65%	72.06%	7%	40%	95.60%
DeepSeek-V3	45%	82%	97.38%	17%	50%	98.78%
GPT-4o	47%	80%	99.07%	18%	44%	99.04%
Claude-4-Sonnet	69%	84%	98.32%	28%	59%	99.16%
DeepSeek-V4-Pro	48%	86%	97.88%	24%	69%	98.25%
Training Paradigm (Qwen2.5-7B-Instruct)						
SFT	38%	72%	92.04%	27%	76%	91.96%
GRPO	28%	67%	95.65%	10%	56%	95.55%
SFT+GRPO	44%	75%	97.02%	41%	83%	96.81%
Ours	**78%**	**97%**	**99.48%**	**76%**	**97%**	**99.28%**

Table 2: **Performance comparison across different training setups and inference modes for GuanDan and DouDizhu.** AO: Action-only. RtA: Reasoning-then-Action. AtR: Action-then-Reasoning.

Training Setup	Inference Mode	GuanDan			DouDizhu		
		Win rate (%)		Action Validity (%)	Win rate (%)		Action Validity (%)
		vs Rule	vs Random		vs Rule	vs Random	
Non-reasoning model							
AO	AO	69%	86%	99.8%	83%	97%	99.91%
Reasoning Models							
AtR	AtR	41%	81%	94.5%	29%	75%	90.32%
RtA	RtA	38%	72%	92.04%	27%	76%	91.96%
AO & AtR	AO	57%	81%	96.45%	64%	90%	98.8%
	AtR	42%	81%	97.86%	32%	75%	92.5%
AO & RtA	AO	58%	79%	97.5%	61%	93%	99.5%
	RtA	42%	75%	92.73%	30%	81%	92.55%

Experimental results demonstrate that Metis effectively harmonizes competitive card-playing proficiency with explicit strategic reasoning, outperforming diverse baselines in complex card games. **By anchoring CoT reasoning to high-confidence trajectories, the framework generates reasoning traces closely aligned with strategic actions, offering a practical and scalable route toward decision transparency.**

4.3 Ablation Study

4.3.1 Empirical Analysis of CoT

To analyze the impact of CoT on decision-making performance, we design controlled experiments that isolate the effects of CoT intervention and training strategies, while fixing model scale and data volume. As shown in Table 2, we compare five configurations along two dimensions: supervision format and inference paradigm, including Action-only (AO), Reasoning-then-Action (RtA), Action-then-Reasoning (AtR), and two mixed supervision variants. All settings share identical training hyperparameters.

Table 3: Ablation study on reward components for GuanDan and DouDizhu.

Game	Reward variant	Win rate (%)		Action Validity (%)	CoT length
		vs Rule	vs Random		
GuanDan	full reward	36%	68%	97.02%	288
	w/o length reward	38%	65%	98.31%	178
	w/o soft-matching reward	21%	58%	97.43%	286
DouDizhu	full reward	13%	73%	96.81%	285
	w/o length reward	16%	72%	97.56%	222
	w/o soft-matching reward	9%	34%	96.24%	284

Results show clear performance divergence across paradigms. In both GuanDan and DouDizhu, AO achieves the highest performance. In contrast, explicitly introducing CoT during inference is associated with lower performance, with RtA showing the largest observed drop in both win rate and action validity. One possible explanation is an optimization conflict between long-form generation and action selection, although we do not directly test this mechanism.

Across both GuanDan and DouDizhu, mixed training consistently outperforms pure CoT supervision. This indicates that CoT rationales function more effectively as an auxiliary training signal than as an isolated generation objective in complex card-game settings.

A compelling explanation for this phenomenon lies in the verbal overshadowing effect (Liu et al., 2024). Strategic card-game decisions heavily rely on implicit statistical pattern recognition, rapid state abstraction, and dynamic opponent modeling—processes that may not naturally align with linear, explicit linguistic articulation. Forcing the model to strictly verbalize these implicit representations can overemphasize local explanation consistency at the expense of global decision coherence. In dynamic multi-agent card games, unconstrained long-form text generation further introduces risks of hallucination and action invalidity. While our empirical metrics focus on macro-level policy outcomes, this verbal overshadowing framework provides a plausible theoretical account for the superiority of joint policy-rationale supervision.

4.3.2 Impact of Reward Components

To further validate the impact of structured reward design in the RL phase, we conducted ablation studies focusing on soft-matching rewards and length rewards.

Table 3 indicates that removing the soft-matching mechanism led to a significant decrease in win rate, while the action validity rate rate remained largely unchanged. In discrete and highly sparse decision spaces, such as card games, a perfect match reward is overly stringent, resulting in highly sparse effective gradient signals that undermine the continuity and stability of policy optimization. The soft-matching component enables fine-grained differentiation of action structures, hand-type consistency, and local policy proximity, providing the model with clear optimization guidance and thus exerting a decisive influence on the win rate.

Furthermore, omitting the length reward consistently leads to shorter CoT generations. While overall win rates remain largely unchanged and action validity increases marginally by approximately 1%, unconstrained shortening risks compressing intermediate reasoning and discarding strategic detail. Rather than dictating decision accuracy directly, the length reward functions as a structural regulator that governs output verbosity. Consequently, its primary role is to maintain the expressive depth of the CoT within a reasonable token budget, balancing decision transparency with generation conciseness.

In summary, ablation studies confirm the necessity of structured rewards. Soft-matching rewards maintain learnability under sparse supervision; without them, policy performance significantly declines. Length rewards regulate CoT scale, preventing excessive reasoning compression and preserving expressive capacity. **Together, these rewards optimize decision quality and reasoning morphology, promoting win-rate gains during the RL phase while preserving CoT stability.**

4.3.3 Analysis of Iterative Cycles

To examine iterative training in the self-distillation loop, we varied the number of cycles under RL-data distillation. Table 4 shows steady gains through the third cycle in both games, followed by

lower win rates and saturated action validity in cycle 4. The third cycle therefore marks the best performance point across the two domains.

One possible explanation is that policy knowledge is extracted progressively across iterative cycles. Under this view, early cycles primarily distill foundational action validity, local preferences, and basic tactical responses, whereas subsequent iterations progressively expose cooperative coordination, risk-aware decisions, and long-horizon trade-offs through repeated regeneration and filtering. As our present metrics focus on macro-level policy performance, we frame this progressive knowledge transfer as a plausible working hypothesis awaiting deeper qualitative verification.

Table 4: Performance over self-distillation cycles.

Game	#Iterations	Win rate (%)		Action Validity (%)
		vs Rule	vs Random	
GuanDan	0	16%	46%	94.48%
	1	49%	82%	97.38%
	2	68%	91%	99.07%
	3	**78%**	**97%**	**99.48%**
	4	71%	92%	99.44%
DouDizhu	0	4%	21%	90.9%
	1	48%	86%	95.78%
	2	61%	91%	98.9%
	3	**76%**	**97%**	99.28%
	4	72%	93%	**99.32%**

Beyond the third cycle, the student model appears to saturate its capacity regarding the informative signals within teacher trajectories. Further self-distillation yields negligible win-rate gains in our experiments, instead risking overfitting to recurring trajectory motifs rather than acquiring novel supervision. **Among the evaluated configurations, three cycles provide the best observed trade-off between performance and training cost.**

4.4 Distillation Analysis

We investigate how teacher data source relates to distillation performance by contrasting trajectories from RL-based and rule-based models. As shown in Table 5, RL-based distillation continues to improve through three cycles in our evaluation, whereas rule-based distillation saturates after the first iteration. This pattern is consistent with overfitting to recurring rule-based trajectories, although we do not directly measure generalization to atypical game states.

We hypothesize that this disparity arises from fundamental differences in the policy distributions covered by the two teachers. Optimized via long-term returns and adversarial interaction, the RL teacher provides trajectories that expose the student to a broader spectrum of situational trade-offs under partial observability and opponent coupling. Conversely, while rule-based data facilitates the rapid acquisition of action validity, its explicit heuristics inherently cover a narrower strategy distribution with fewer non-local payoff dynamics. As a result, the student exhausts the novel supervisory signals from rule-based patterns much earlier. While our empirical metrics primarily capture win-rate divergence, we present these distributional coverage differences as a compelling explanation for the observed distillation behavior.

Table 5: Performance comparison between Rule model and RL model as teachers.

Game	Teacher model	Win rate (%)		Action Validity (%)
		vs Rule	vs Random	
GuanDan	Rule model	59%	91%	99.34%
	RL model	78%	97%	99.48%
DouDizhu	Rule model	50%	89%	98.92%
	RL model	76%	97%	99.28%

These results demonstrate that RL-teacher trajectories yield higher win rates than rule-teacher trajectories in the evaluated settings. They highlight that both the source and distributional coverage of policy supervision fundamentally set the performance ceiling for self-distilled agents.

5 Conclusion

This study investigates the deployment of LLMs for strategic decision-making in complex card games. By dissecting the functional impact of CoT reasoning in GuanDan and DouDiZhu, we reveal that unconstrained reasoning does not inherently bolster decision efficacy and may even impair it. To address the scarcity of expert CoT annotations, we propose Metis—a self-evolving paradigm that integrates Mixed SFT and structured GRPO within an iterative self-distillation mechanism. Ultimately, Metis demonstrates that competitive proficiency and decision transparency can be effectively harmonized within a unified framework. This provides a scalable paradigm for complex non-linguistic decision-making tasks where balancing performance and explicit rationale generation is paramount.

AI USE STATEMENT

In this work, we used generative AI tools for drafting and editing parts of the manuscript, improving readability, formatting references, suggesting structure, brainstorming, searching for relevant literature, assisting with method and experimental-code implementation. We have not used generative AI tools for generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating or proving mathematical claims, or proposing or refining hypotheses. The remaining required disclosure tasks—designing the research methodology or experiments, cleaning or reformatting datasets, supporting qualitative and thematic data analysis, and interpreting experimental results—are not applicable to this work.

We have reviewed all AI-assisted work. AI-assisted code was verified and tested for correctness by the authors. AI-assisted text was checked against the source data, experimental results, and tables for accuracy. All AI-suggested references were manually verified against the original sources. We take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This study evaluates language-model agents in simulated GuanDan and DouDizhu environments using trajectories generated by game agents. It does not involve human subjects, human annotations, user studies, or personal data. The work is intended to study strategic decision-making and interpretability; limitations and broader impacts are discussed in Appendix E and Appendix F.

REPRODUCIBILITY STATEMENT

Section 4.1 describes the data construction, evaluated models, opponents, and metrics. Appendix B specifies the reward function, GRPO objective, training and inference hyperparameters, hardware, and evaluation protocol; the remaining appendix provides supplementary results and case studies. Anonymous source code is available at `https://anonymous.4open.science/r/Metis`.

REFERENCES

LV ALLIS. Searching for solutions in games and artificial intelligence. *Ph. D. Thesis, University of Limburg*, 1994.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*, 2024.

Yiquan Chen, Yingchao Lyu, and Di Zhang. Dourn: Improving douzero by residual neural networks. In *2023 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 96–99. IEEE, 2023.

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pp. 72–83. Springer, 2006.

Jinhao Duan, Renming Zhang, James Diffenderfer, Bhavya Kailkhura, Lichao Sun, Elias Stengel-Eskin, Mohit Bansal, Tianlong Chen, and Kaidi Xu. Gtbench: Uncovering the strategic reasoning capabilities of llms via game-theoretic evaluations. *Advances in Neural Information Processing Systems*, 37:28219–28253, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Lanxiang Hu, Mingjia Huo, Yuxuan Zhang, Haoyang Yu, Eric P Xing, Ion Stoica, Tajana Rosing, Haojian Jin, and Hao Zhang. lmgame-bench: How good are llms at playing games? In *International Conference on Learning Representations*, volume 2026, pp. 81308–81356, 2026.

Sihao Hu, Tiansheng Huang, Gaowen Liu, Ramana Rao Kompella, Fatih Ilhan, Selim Furkan Tekin, Yichang Xu, Zachary Yahn, and Ling Liu. A survey on large language model-based game agents. *arXiv preprint arXiv:2404.02039*, 2024.

Andrés Isaza-Giraldo, Paulo Bala, Pedro F Campos, and Lucas Pereira. Prompt-gaming: A pilot study on llm-evaluating agent in a meaningful energy game. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2024.

Qiqi Jiang, Kuangzheng Li, Boyao Du, Hao Chen, and Hai Fang. Deltadou: Expert-level doudizhu ai through self-play. In *IJCAI*, pp. 1265–1271, 2019.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Marc Lanctot, Edward Lockhart, Jean-Baptiste Lespiau, Vinicius Zambaldi, Satyaki Upadhyay, Julien Pérolat, Sriram Srinivasan, Finbarr Timbers, Karl Tuyls, Shayegan Omidshafiei, et al. Openspiel: A framework for reinforcement learning in games. *arXiv preprint arXiv:1908.09453*, 2019.

Jonathan Light, Min Cai, Weiqin Chen, Guanzhi Wang, Xiusi Chen, Wei Cheng, Yisong Yue, and Ziniu Hu. Strategist: Self-improvement of llm decision making via bi-level tree search. *arXiv preprint arXiv:2408.10635*, 2024.

Wenye Lin, Jonathan Roberts, Yunhan Yang, Samuel Albanie, Zongqing Lu, and Kai Han. Gamebot: Transparent assessment of llm reasoning in games. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7656–7682, 2025.

Ryan Liu, Jiayi Geng, Addison J Wu, Ilia Sucholutsky, Tania Lombrozo, and Thomas L Griffiths. Mind your step (by step): Chain-of-thought can reduce performance on tasks where thinking makes humans worse. *arXiv preprint arXiv:2410.21333*, 2024.

Yudong Lu, Youpeng Zhao, Wengang Zhou, Houqiang Li, et al. Danzero: Mastering guandan game with reinforcement learning. In *2023 IEEE Conference on Games (CoG)*, pp. 1–8. IEEE, 2023.

Weiyu Ma, Qirui Mi, Yongcheng Zeng, Xue Yan, Yuqiao Wu, Runji Lin, Haifeng Zhang, and Jun Wang. Large language models play starcraft ii: Benchmarks and a chain of summarization approach. *Advances in Neural Information Processing Systems*, 37:133386–133442, 2024.

Yichuan Ma, Linyang Li, Yongkang Chen, Peiji Li, Jiasheng Ye, Qipeng Guo, Dahua Lin, and Kai Chen. Mixing expert knowledge: Bring human thoughts back to the game of go. *Advances in Neural Information Processing Systems*, 38:94643–94672, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Davide Paglieri, Bartłomiej Cupiał, Samuel Coward, Ulyana Piterbarg, Maciej Wołczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, et al. Balrog: Benchmarking agentic llm and vlm reasoning on games. In *International Conference on Learning Representations*, volume 2025, pp. 96666–96702, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Mahesh Ramesh, Aswinkumar Ramkumar, Pavan Thodima, Kaousheik Jayakumar, and Aniket Rege. Are llms generalist hanabi agents? In *Workshop on Scaling Environments for Agents*, 2025.

Anian Ruoss, Fabio Pardo, Harris Chan, Bonnie Li, Volodymyr Mnih, and Tim Genewein. Lmact: A benchmark for in-context imitation learning with long multimodal demonstrations. *arXiv preprint arXiv:2412.01441*, 2024.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

John Schultz, Jakub Adamek, Matej Jusup, Marc Lanctot, Michael Kaisers, Sarah Perrin, Daniel Hennes, Jeremy Shar, Cannada Lewis, Anian Ruoss, et al. Mastering board games by external and internal planning with language models. *arXiv preprint arXiv:2412.12119*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Haolin Wang, Xueyan Li, Yazhe Niu, Shuai Hu, and Hongsheng Li. Empowering llms in decision games through algorithmic data synthesis. *arXiv preprint arXiv:2503.13980*, 2025a.

Wei Wang, Fuqing Bie, Junzhe Chen, Dan Zhang, Shiyu Huang, Evgeny Kharlamov, and Jie Tang. Can large language models master complex card games? *arXiv preprint arXiv:2509.01328*, 2025b.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yue Wu, So Yeon Min, Shrimai Prabhumoye, Yonatan Bisk, Russ R Salakhutdinov, Amos Azaria, Tom M Mitchell, and Yuanzhi Li. Spring: Studying papers and reasoning to play games. *Advances in Neural Information Processing Systems*, 36:22383–22687, 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Yauwai Yim, Chunkit Chan, Tianyu Shi, Zheye Deng, Wei Fan, Tianshi Zheng, and Yangqiu Song. Evaluating and enhancing llms agent based on theory of mind in guandan: A multi-player cooperative game under imperfect information. In *2024 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pp. 461–465. IEEE, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Huining Yuan, Zelai Xu, Zheyue Tan, Xiangmin Yi, Mo Guang, Kaiwen Long, Haojia Hui, Boxun Li, Xinlei Chen, Bo Zhao, et al. Marshal: Incentivizing multi-agent reasoning via self-play with strategic llms. *arXiv preprint arXiv:2510.15414*, 2025.

Daochen Zha, Kwei-Herng Lai, Yuanpu Cao, Songyi Huang, Ruzhe Wei, Junyu Guo, and Xia Hu. Rlcard: A toolkit for reinforcement learning in card games. *arXiv preprint arXiv:1910.04376*, 2019.

Daochen Zha, Jingru Xie, Wenye Ma, Sheng Zhang, Xiangru Lian, Xia Hu, and Ji Liu. Douzero: Mastering doudizhu with self-play deep reinforcement learning. In *international conference on machine learning*, pp. 12333–12344. PMLR, 2021.

Wanpeng Zhang and Zongqing Lu. Adarefiner: Refining decisions of language models with adaptive feedback. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 782–799, 2024.

Yinqi Zhang, Xintian Han, Haolong Li, Kedi Chen, and Shaohui Lin. Complete chess games enable llm become a chess master. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 1–7, 2025.

Youpeng Zhao, Yudong Lu, Jian Zhao, Wengang Zhou, and Houqiang Li. Danzero+: Dominating the guandan game through reinforcement learning. *IEEE Transactions on Games*, 16(4):914–926, 2024.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

A Appendix

A.1 Introduction to Card Games

GuanDan. GuanDan[1], a widely popular four-player cooperative card game in China, is played with two standard decks, totaling 108 cards. Players form two partnerships, competing against each other. The rules integrate elements from games like DouDizhu and upgrades, featuring complex strategic dimensions such as card combinations, an upgrade mechanism, and special rules like "Feng Ren Pei" (matching any suit). The gameplay emphasizes signaling between partners, long-term planning, and dynamic coordination, characterizing it as an imperfect information team game within a high-dimensional discrete action space. Current AI research in this domain is relatively scarce, with primary efforts focused on hybrid approaches combining RL and search. Notably, the DanZero Lu et al. (2023); Zhao et al. (2024), which enhances Monte Carlo methods with neural networks, has surpassed existing algorithms in performance, representing a significant technological advancement in this field.

DouDizhu. DouDizhu[2], also known as Fighting the Landlord, stands as China's most widely played three-player poker game. Utilizing a standard 54-card deck, including two jokers, the game involves two opposing sides: a single "landlord" and two "farmers." Gameplay is structured into bidding and playing phases. The landlord, determined through bidding, secures three hidden cards and operates independently, while the farmers collaborate against them. Victory is achieved when either side successfully plays all their cards first. Characterized by incomplete information, dynamic action spaces, and team-based competition, DouDizhu has emerged as a significant benchmark environment for research in game intelligence. Existing research methodologies encompass heuristic

[1] https://en.wikipedia.org/wiki/Guandan

[2] https://en.wikipedia.org/wiki/Dou_dizhu

rules, RL, and search algorithms Jiang et al. (2019); Zha et al. (2021); Chen et al. (2023). Notably, the DouZero Zha et al. (2021), based on Deep Q-Networks and requiring neither human knowledge nor state abstraction, represents the current state-of-the-art publicly available DouDizhu AI system.

A.2 Description of the Self-Distillation loop

Algorithm 1 outlines the proposed self-distillation loop. The algorithm initializes the model M with M_0 and populates the data pool $\mathcal{D}_{\text{pool}}$ with $\mathcal{D}_{\text{total}}$. In each iteration k, candidate CoT data $\mathcal{D}_{\text{new}}$ is sampled from the current policy π_θ. Specifically, in the first iteration ($k = 1$), reasoning generation is explicitly conditioned on ground-truth actions a^*. In subsequent iterations ($k > 1$), generation proceeds without constraints, retaining only trajectories whose predicted actions match a^*. Across all iterations, the candidate trajectories in $\mathcal{D}_{\text{new}}$ undergo LLM-based filtering to ensure rationale quality. The algorithm then samples an equal number of direct-action instances $\mathcal{D}_{\text{direct}}$ from $\mathcal{D}_{\text{pool}}$, constructs the training set $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{new}} \cup \mathcal{D}_{\text{direct}}$, and removes the selected samples from the pool. Subsequently, the model is updated via Mixed SFT followed by GRPO optimization. This cycle repeats for K iterations, returning the optimized model M_K.

Algorithm 1 Self-distillation Loop

Require: Dataset $\mathcal{D}_{\text{total}}$, base model M_0, iterations K, reward function $\mathcal{R}$
Ensure: Optimized model M_K

$M \leftarrow M_0$
$\mathcal{D}_{\text{pool}} \leftarrow \mathcal{D}_{\text{total}}$ {$\mathcal{D}_{\text{total}}$ consists of winning trajectories from SOTA model self-play}
for $k = 1$ **to** K **do**
  if $k = 1$ **then**
    $\mathcal{D}_{\text{new}} \leftarrow \{(s, c, a) \sim \pi_\theta(c, a \mid s, a^*) \mid (s, a^*) \in \mathcal{D}_{\text{pool}}\}$
  else
    $\mathcal{D}_{\text{new}} \leftarrow \{(s, c, a) \sim \pi_\theta(c, a \mid s) \mid (s, a^*) \in \mathcal{D}_{\text{pool}}\}$
    $\mathcal{D}_{\text{new}} \leftarrow \{(s, c, a) \in \mathcal{D}_{\text{new}} \mid a = a^*\}$
  end if
  $\mathcal{D}_{\text{new}} \leftarrow \text{Filter}(\mathcal{D}_{\text{new}})$ {LLMs Filtering}
  $\mathcal{D}_{\text{direct}} \leftarrow \text{Sample}(\mathcal{D}_{\text{pool}}, |\mathcal{D}_{\text{new}}|)$
  $\mathcal{D}_{\text{train}} \leftarrow \mathcal{D}_{\text{new}} \cup \mathcal{D}_{\text{direct}}$
  $\mathcal{D}_{\text{pool}} \leftarrow \mathcal{D}_{\text{pool}} \setminus (\mathcal{D}_{\text{new}} \cup \mathcal{D}_{\text{direct}})$
  $M \leftarrow \text{Mixed SFT}(M, \mathcal{D}_{\text{train}})$
  $M \leftarrow \text{GRPO}(M, \mathcal{R})$
end for
return M

A.3 Scope of Applicability and Generalizability

In this section, we delineate the applicability boundaries of the Metis framework and discuss its potential extension across domains and foundational model architectures.

Generalizability Beyond Card Games. While our empirical validation focuses on GuanDan and DouDiZhu—two representative multi-player, incomplete-information card games—Metis is fundamentally designed for broader multi-agent strategic decision-making environments. The core mechanics of Metis, specifically the evolutionary alignment loop combining Mixed SFT and structured GRPO, are task-agnostic. The framework is applicable to any sequential control or strategic setting where reliable reference actions or high-confidence trajectories can be acquired (e.g., via domain experts, search algorithms, or high-performing rule-based systems). Future work will extend the empirical evaluation of Metis to non-game sequential decision tasks and real-world multi-agent control problems.

Model Agnosticism and Compatibility with Reasoning Models. In our primary experiments, we instantiated Metis using Qwen2.5-7B-Instruct to strike an optimal balance among open-source accessibility, computational efficiency, and strong baseline reproducibility. However, the architectural design of Metis is strictly model-agnostic. In principle, the framework can be seamlessly transferred to LLMs explicitly post-trained for reasoning (e.g., reasoning-centric models trained via large-scale RL). We anticipate that initializing Metis with such reasoning-trained models will provide

stronger intrinsic CoT priors and richer candidate search spaces, further accelerating the evolutionary distillation process. Comprehensive validation across diverse model families remains an exciting direction for future research.

B EXPERIMENTAL DETAILS

Reward Function Design. For both GuanDan and DouDizhu, the reward combines format and accuracy scores as $0.9 \times R_{\text{acc}} + 0.1 \times R_{\text{fmt}}$. Format reward requires paired `<think>` and `</think>` tags followed by a JSON object containing an `"action"` field. Accuracy reward employs soft matching: full reward (1.0) for exact action match; partial reward (0.3) for type match (e.g., card type in GuanDan, or list length match in DouDizhu); an additional 0.2 for rank match (or first card match in DouDizhu). Outputting PASS when the ground truth is not PASS incurs a 0.5 penalty. Missing the closing `</think>` tag incurs a penalty of 0.5.

Length penalties differ by game. For GuanDan, outputs shorter than 100 characters incur a penalty of 0.4; between 100 and 200 characters incur 0.1; outputs exceeding 1400 characters incur 0.01 per additional 100 characters (capped at 0.5). For DouDizhu, outputs shorter than 250 characters incur 0.4; between 250 and 450 characters incur 0.1; outputs exceeding 1600 characters incur 0.01 per additional 100 characters (capped at 0.5). The overall reward is defined as:

$$R = 0.9 \times (R_{\text{action}} - R_{\text{length}}) + 0.1 \times R_{\text{fmt}} \tag{1}$$

where R_{action} is the action matching reward based on soft-matching, R_{length} is the length penalty, and R_{fmt} is the format reward.

GRPO Policy Optimization Details. To prevent catastrophic forgetting and ensure training stability by constraining the distributional shift from the reference model, we optimize the following objective function:

$$\begin{aligned}\mathcal{J}_{GRPO}(\theta) =& \mathbb{E}_{s\sim\mathcal{D},\{o_k\}_{k=1}^{K}\sim\pi_{\theta_{\text{old}}}} \\ & \left[\frac{1}{K}\sum_{k=1}^{K}\left(\min\left(\rho_k(\theta)\hat{A}_k, \text{clip}(\rho_k(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_k\right) - \beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\right)\right]\end{aligned} \tag{2}$$

where o_k denotes the reasoning-action pair (c_k, a_k), $\rho_k(\theta) = \frac{\pi_\theta(o_k|s)}{\pi_{\theta_{\text{old}}}(o_k|s)}$ is the importance sampling ratio, and $\hat{A}_k$ is the group-relative advantage derived by standardizing R_k within the sample set.

Experimental Settings. During the SFT phase, we trained the model on a dataset of 300k samples using a learning rate of 5e-5, a batch size of 128, and a total of 3 epochs. For the GRPO phase, we used 100k samples for training, setting the learning rate to 2e-5, the batch size to 128, the number of sampled actions per state to $G = 4$, and a total of 2,000 training steps. All experimental procedures were conducted on a server equipped with 8 A100 GPUs.

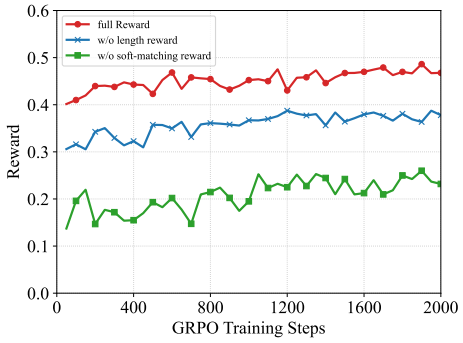
Evaluation Details. In both GuanDan and DouDizhu, the LLM generates actions using the vLLM framework Kwon et al. (2023) with temperature set to 0.5, top-p=1.0, top-k=50, repetition penalty=1.0, and a maximum generation length of 8192 tokens. The prompt follows the template described in the main paper. For GuanDan, we implement a WebSocket-based client that interacts with the game server. For DouDizhu, the evaluation is conducted on an offline dataset of pre-recorded game states, with each episode initialized from a stored state and simulated by the GameEnv. In both environments, if the model output fails to parse or yields an illegal action, a random legal action is selected as fallback to ensure the game proceeds. Both environments use the same model checkpoint and share identical inference hyperparameters to ensure fair comparison.

C SUPPLEMENTARY RESULTS

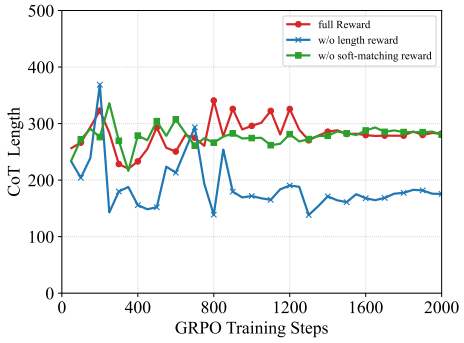
C.1 GRPO TRAINING CURVES

As shown in Figures 2 and 3, across both GuanDan and DouDizhu, the full reward configuration achieves higher and more stable convergence, whereas removing either the length penalty or the soft-matching reward yields sparser feedback signals and lower overall rewards. Regarding CoT

length, omitting the length penalty leads to significantly truncated reasoning chains, which impairs decision transparency and rationale completeness. These empirical findings demonstrate the necessity of each reward component for optimizing both learning stability and explanation quality.

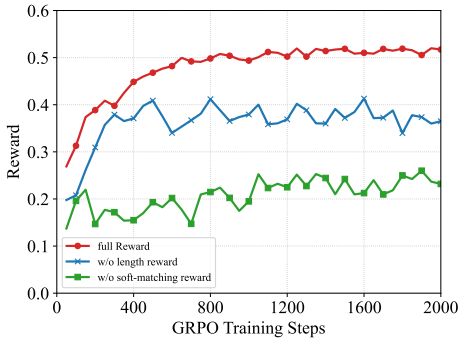


(a) GRPO reward curves

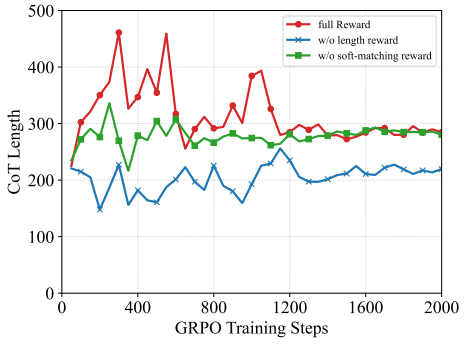


(b) CoT length curves

Figure 2: Ablation Study on Reward Function Design for GuanDan.



(a) GRPO reward curves



(b) CoT length curves

Figure 3: Ablation Study on Reward Function Design for DouDizhu.

C.2 Iterative stage breakdown of SFT and GRPO.

Table 6 provides a per-iteration breakdown of SFT and GRPO stages. Iteration 0 corresponds to the untrained baseline, while subsequent iterations separate SFT and GRPO.

C.3 Comparison Against SOTA Action-Only Models

Table 7 compares our reasoning-enhanced model against strong action-only SOTA baselines, including SFT (AO), SFT+GRPO (standard GRPO applied to the SFT model), and domain-specific SOTA models (Danzero for GuanDan and Douzero for DouDizhu). On GuanDan, our model achieves a 78% win rate against rule-based opponents and 97% against random opponents, with 99.48% action validity. While the action-only SOTA model (Danzero) attains higher raw win rates (93%/99%), it operates as a black box without generating explicit reasoning traces. In contrast, our method maintains competitive performance while simultaneously producing semantically meaningful CoT rationales. A similar pattern emerges in DouDizhu: our model achieves 76% (rule-based) and 97% (random) win rates with 99.28% validity, remaining competitive with the action-oriented SOTA (Douzero: 88%/99%) while providing decision transparency. These results demonstrate that our framework successfully bridges the gap between high-performance game-playing and explicit strategic reasoning, achieving strong competitive results without sacrificing decision transparency.

C.4 Head-to-Head Evaluation

We provide head-to-head evaluation results below, including comparisons against the state-of-the-art model DouZero. Columns denote landlord policies, rows denote peasant-team policies, and each

Table 6: Iterative performance breakdown of SFT and GRPO stages across different iterations.

Game	Iteration	Stage	Win rate (%)		Action Validity (%)
			vs Rule	vs Random	
GuanDan	0	–	16%	46%	94.48%
	1	SFT	42%	75%	92.73%
		GRPO	49%	82%	97.38%
	2	SFT	55%	86%	98.56%
		GRPO	68%	91%	99.07%
	3	SFT	71%	94%	99.26%
		GRPO	**78%**	**97%**	**99.48%**
	4	SFT	69%	91%	99.34%
		GRPO	71%	92%	99.44%
DouDizhu	0	–	4%	21%	90.9%
	1	SFT	30%	81%	92.55%
		GRPO	48%	86%	95.78%
	2	SFT	52%	89%	97.63%
		GRPO	61%	91%	98.9%
	3	SFT	66%	96%	99.3%
		GRPO	**76%**	**97%**	99.28%
	4	SFT	75%	91%	99.15%
		GRPO	72%	93%	**99.32%**

Table 7: Performance comparison of our reasoning-enhanced model against SOTA action-only baselines

Game	Method / Model	Win rate (%)		Action Validity (%)
		vs Rule	vs Random	
GuanDan	*Action Models*			
	Danzero	93%	99%	100%
	SFT	69%	86%	99.8%
	SFT+GRPO	79%	90%	99.92%
	Reasoning Model			
	Ours	78%	97%	99.48%
DouDizhu	*Action Models*			
	Douzero	88%	99%	100%
	SFT	83%	97%	99.91%
	SFT+GRPO	76%	99%	99.94%
	Reasoning Model			
	Ours	76%	97%	99.28%

Table 8: Head-to-head evaluation results. Columns denote landlord policies; rows denote peasant-team policies. Each cell reports the landlord win rate. "—" indicates that the policy is not evaluated against itself.

Peasant team \ Landlord	Random	RLCard	DouZero	Qwen2.5-7B-Instruct	Metis
Random	—	89%	98%	21%	97%
RLCard	2%	—	78%	4%	76%
DouZero	0%	11%	—	0%	21%
Qwen2.5-7B-Instruct	47%	95%	96%	—	93%
Metis	0%	35%	68%	1%	—

Table 9: Peasant-team evaluation results. "vs RLCard" and "vs Random" report the peasant-team win rate against the corresponding landlord policy. "Action validity" reports the proportion of legal actions generated by each policy.

Peasant evaluation	Qwen2.5-7B-Instruct	Loop 1	Loop 2	Metis
vs RLCard	5%	60%	63%	65%
vs Random	53%	96%	99%	100%
Action validity	86.96%	98.76%	98.82%	99.56%

cell reports the landlord win rate. The diagonal entries are marked with "—" because a policy is not evaluated against itself.

The relative performance ranking in this head-to-head evaluation is consistent with the results in Table 8: Metis achieves competitive performance in both roles, ranking second only to DouZero. DouZero, specifically designed and trained for this game, achieves the strongest performance. However, it does not generate natural-language CoT traces. Metis aims to achieve competitive decision performance while additionally providing natural-language reasoning processes underlying its actions.

C.5 Peasant-Team Evaluation Results

We additionally evaluate the LLM as the peasant team and report the results below. Columns denote different policies, and rows denote the evaluation setting. "vs RLCard" and "vs Random" report the peasant-team win rate against the corresponding landlord policy, while "Action validity" reports the proportion of generated actions that are legal in the game environment.

The matrix confirms that Metis's advantage is not role-specific: Metis achieves competitive performance whether controlling the landlord or the peasant team. The trend is consistent across roles and aligns with the main results in Table 9. In particular, Metis attains the highest peasant-team win rate against both RLCard and Random, and also the highest action validity, indicating that its improvements are not confined to the landlord role. The LLM baseline, by contrast, remains substantially weaker as the peasant team, especially against RLCard, where its win rate is only 5% and its action validity is 86.96%. These results justify evaluating the LLM primarily as the landlord in the main experiments while confirming that Metis's gains generalize across roles.

D Case Studies

D.1 GuanDan

Input Prompt The input segment is architected using modular components, leveraging structured formats like strings, lists, and dictionaries. Game rules are embedded in natural language to ensure the model comprehends global constraints. Player meta-information encompasses position, level, ranking cards, and wild cards. Hand and public information include current hands, remaining cards of other players, and the count of cards each player possesses. Historical action records are listed independently per player, detailing recent moves and played cards. The legal action space enumerates

all permissible card combinations via nested lists, with each action adhering to a [Type, Rank, Cards] triplet format. The output is a dictionary strictly conforming to JSON, keyed by "action," with its value being a list of triplets isomorphic to the input's legal actions, ensuring direct executability.

The superiority of this design lies in its modularity and parsability, which deconstruct the input into independent semantic units, facilitating layered model inference. The output, being isomorphic to the input action space, eliminates ambiguities arising from format conversions. Comprehensive card encoding (e.g., S2 for Spades 2, SB/HR for Jokers) covers all 108 cards. The card type triplet, with its fixed type, representative card rank, and specific card list, eradicates ambiguity. The separation of rule descriptions from game state allows for easy modification of ranking or wild cards by merely updating corresponding fields, ensuring extensibility. Historical actions are recorded independently per player and supplemented with remaining card distributions, aiding in backtrace debugging. The explicit provision of the legal action space prevents the model from generating illegal actions, thereby reducing post-processing costs. Furthermore, structured numerical data (hands, remaining card counts) can be directly utilized for heuristic evaluations or as features for RL extraction. The JSON output is readily embeddable within automated game execution pipelines.

Figure 4: Input Prompt of GuanDan

```
You FIRST think about the reasoning process as an internal monologue
    and then provide the final answer.
The reasoning process MUST BE enclosed within <think> </think> tags.
    The final answer MUST BE in json format.
You are now a player in a game of GuanDan. The game rules are as
    follows:

1. The game is played by four players in partnerships, sitting
    opposite each other.
2. The deck consists of two standard international decks with Jokers,
    totaling 108 cards.
3. The objective is to play higher combinations of cards to empty your
     hand before your opponents.
4. If your team completes the game first, you will advance in levels;
    the ultimate goal is to win on Level A.
5. Card ranks in increasing order are: 2, 3, 4, 5, 6, 7, 8, 9, 10, J,
    Q, K, A.
6. There are four suits (Spades, Hearts, Diamonds, Clubs) and four
    Jokers (two red, two black).
7. Players take turns in counterclockwise order, starting from a
    player who plays any combination of cards.
8. Other players must play higher cards of the same type or a higher
    combination, or they must pass.
9. The game continues until three players have finished their cards.
10. Players are given titles based on the order they finish: Banker,
    Follower, Third, and Dweller.

The special cards and comparison are as follows:
1. Level Cards: The level number of the leading team determines the
    level cards. The level cards rank above aces but below jokers. For
     example, if the leading team is at level 6, then sixes are the
    level cards and rank above A.
2. Wild Cards: During the round, they can be played as any card,
    except jokers, to form a combination with other cards. However,
    they only count as normal, non-wild cards when played as a single
    card. Wild card can form Bomb, StraightFlush, Straight,
    ThreeWithTwo, ThreePair, TwoTrips, Pair, or Trips, but never
    Single or JokerBomb.
3. Normal Comparison: The normal comparison of the cards is from high
    to low in the order of small Joker, big joker, A, K, Q, J, 10, 9,
    8, 7, 6, 5, 4, 3, 2. It applies when comparing with a Single, Pair,
     Trips, ThreePair, TwoTrips, Straight, Bomb, and StraightFlush.
```

```
    Specially, ThreeWithTwo compares the Trips in the combination only.

4. Bomb Comparison: Bomb depends on its number of cards. The smallest
    is a 4-bomb of 2s and the largest is an 8-bomb of aces. However, a
     5-bomb of 2s is larger than a 4-bomb of aces. A bomb ranks above:
     Single, Pair, Trips, ThreePair, TwoTrips, ThreeWithTwo, Straight.
     A StraightFlush is regarded as a bomb that ranks above a 4 or 5-
    card bomb, except the JokerBomb. A bomb with 6 or more cards ranks
     above a StraightFlush. StraightFlushes rank according to their
    largest card regardless of suits. The JokerBomb is the largest
    bomb in the game.

The representation of cards and card types is as follows:
1. **Cards**: Represented by a two-character string, such as 'S2'
    which means Spade 2. Detailed description below:
   - **Suits**: Spades, Hearts, Clubs, and Diamonds are represented by
     the characters S, H, C, and D respectively. Specifically, the
    suit for the small Joker is S, and for the big Joker, it is H.
   - **Ranks**: A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K are represented
     by A, 2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K respectively. That is,
    the rank 10 is represented by the character T. Specifically, the
    rank for the small Joker is represented by the character B, and
    for the big Joker, it is represented by the character R.
   For example, 'S2' represents Spade 2, 'HQ' represents Heart Q; 'SB'
     represents the small Joker, 'HR' represents the big Joker, 'PASS'
     indicates a pass.

2. **Card Types**: [Type, Rank, Cards]
   A card type is represented by a list of three fixed parts: Type,
    Rank, and Cards.
   - **Type**: The type of card combination, represented as a string
    with possible values of ['Single', 'Pair', 'Trips', 'ThreePair', '
    ThreeWithTwo', 'TwoTrips', 'Straight', 'Bomb', 'PASS', '
    StraightFlush'].
   - **Rank**: The rank of the highest card or representative rank in
    the combination, with possible values of ['A', '2', '3', '4', '5',
     '6', '7', '8', '9', 'T', 'J', 'Q', 'K', 'B', 'R', 'PASS'].
   - **Cards**: The actual cards involved in the combination,
    represented as a list.
   Examples:
   - A single Diamond 5 is represented as: ['Single', '5', ['D5']].
   - A pair of 4s is represented as: ['Pair', '4', ['H4', 'C4']].
   - PASS: ['PASS', 'PASS', 'PASS'].

Your task is to make the best decision in each playing round. I will
    provide you with the following information:

1. Your position:
0

2. Your current hand:
["D3", "H4", "S4", "C4", "C4", "H5", "S5", "H7", "H7", "D7", "D7", "S9
    ", "DT", "DJ", "SQ", "DQ", "HK", "SA", "SB", "HR"]

3. Remaining cards of other players:
["S2", "D2", "C3", "D3", "S4", "S5", "C5", "C5", "D5", "D5", "S6", "S6
    ", "C6", "D6", "D6", "S7", "S8", "C8", "D8", "H9", "C9", "C9", "D9
    ", "ST", "ST", "CT", "DT", "HJ", "HJ", "SJ", "CJ", "DJ", "CK", "CK
    ", "HA", "SA", "CA", "DA", "DA", "SB"]

4. Last action of other players:
["D8", "D9", "CT", "CJ", "CQ"]
```

```
5. Last action of the teammate:
["H6", "C7", "C8", "S9", "HT"]

6. Number of cards left for other players:
{"0": 20, "1": 23, "2": 7, "3": 10}

7. All cards already played by the down player:
["C6", "CQ", "SK", "SK"]

8. All cards already played by the teammate:
["C2", "C2", "H3", "S3", "C3", "H4", "D4", "D4", "H6", "C7", "C8", "S9
    ", "HT", "HT", "SQ", "DQ", "HK", "DK", "DK", "HR"]

9. All cards already played by the up player:
["H3", "S3", "H5", "H6", "S7", "C7", "H8", "D8", "H9", "D9", "CT", "CJ
    ", "HQ", "HQ", "CQ", "HA", "CA"]

10. Self rank:
"Q"

11. Wild card:
"HQ"

12. Current rank:
"Q"

13. Legal actions:
[["PASS", "PASS", ["PASS"]], ["Straight", "9", ["S9", "DT", "DJ", "SQ
    ", "HK"]], ["Straight", "T", ["DT", "DJ", "SQ", "HK", "SA"]], ["
    Bomb", "4", ["H4", "S4", "C4", "C4"]], ["Bomb", "7", ["H7", "H7",
    "D7", "D7"]]]

Please tell me your action in JSON format based on the provided
    information. The JSON should contain an "action" key with a value
    chose from legal actions.

Output format examples:
<think>
[Your reasoning here]
</think>
{"action": ["action_type", "value", ["cards"]]}

Please provide the corresponding JSON action based on the given
    information.
```

Generated CoT Output The model's output unifies structured rationale generation with deterministic decision-making, offering a two-fold advantage. First, the pre-action reasoning process is explicitly articulated in natural language across key strategic dimensions—including hand strength evaluation, wild card optimization, opponent tempo disruption, hand structure adaptation, bomb risk mitigation, and tactical option comparison—thereby establishing a transparent and traceable decision trail conducive to offline debugging and trust evaluation. Second, the final action strictly conforms to the game-defined legal action triplet format (e.g., `["Straight", "T", ["DT","DJ","SQ","HK","SA"]]`), achieving complete structural isomorphism with the action space. This eliminates formatting ambiguities, enabling direct execution by the game engine without requiring heuristic post-processing or conversion. By bridging explicit reasoning with execution-ready output, this dual-layered architecture satisfies the demand for decision transparency in complex game scenarios while ensuring high reliability in automated gameplay.

Figure 5: Generated CoT Output of GuanDan

```
<think>
Based on the provided information, the current situation is as follows
    :
- Your hand contains cards like D3, H4, S4, C4, C4, H5, S5, H7, H7, D7
    , D7, S9, DT, DJ, SQ, HK, SA, SB, HR.
- Your teammate has played a strong combination, completing a straight
     with H6, C7, C8, S9, and HT.
- The down player and the up player have played cards that do not seem
     to form a strong combination yet.
- Your rank is Q, and the wild card is HQ.
- The legal actions include passing, playing a straight or bomb.

Given the current game state, the best move is to play a Straight with
     a high value, specifically a "T" (10) straight, which includes DT,
     DJ, SQ, HK, and SA. Here's the strategic reasoning:

1. **High Straight Value**: Playing a straight with a high value like
    T (10) is strong and can be challenging for the opponents to beat.
     The sequence DT, DJ, SQ, HK, SA is a powerful combination that
    ranks above most other cards in the game.

2. **Utilizing Wild Card**: The wild card HQ can be used to enhance
    the straight. The wild card can be used to complete the sequence,
    making it harder for others to match or beat. However, in this
    case, the straight is already strong enough without needing to use
     the wild card explicitly.

3. **Breaking Opponent's Momentum**: The teammate has already played a
     strong combination, completing a straight. By playing a higher
    straight, you can break the momentum of your teammate and
    potentially force the down and up players to play weaker cards or
    pass.

4. **Card Structure**: Your hand has several 7s and 9s, but not enough
     to form a strong combination. The available cards include D3, H4,
     S4, C4, C4, H5, S5, H7, H7, D7, D7, S9, which do not form a
    strong combination on their own. Therefore, playing a straight is
    a better use of your hand.

5. **Avoiding Bomb**: While you could play a bomb, the straight is a
    more versatile and powerful combination. A bomb, especially a 4-
    bomb or 7-bomb, is powerful but can be beaten by a higher straight.
     The straight has the added advantage of not being a bomb, which
    can be limited in the game and might not be as universally feared.

6. **Passing Strategy**: Passing is always an option, but given the
    strength of the straight and the opportunity to disrupt the game
    flow, it is a better move to play a strong combination.

In conclusion, playing a Straight with a high value T (10) is the
    optimal move considering the hand structure, opponent's cards, and
     game strategy. This move maximizes the potential of your cards
    and sets a strong tone for the rest of the round.
</think>
{"action": [\"Straight\", \"T\", [\"DT\", \"DJ\", \"SQ\", \"HK\", \"SA
    \"]]}
```

D.2 DOUDIZHU

Input Prompt This prompt is architected with a highly modular and domain-specific structured representation, designed to furnish decision-making models with a comprehensive, unambiguous, and inferable game state description. The input comprises several semantically explicit components:

Firstly, natural language embeddings articulate the game's rules, encompassing role definitions, turn order, win/loss conditions, and card-comparison logic, ensuring the model possesses a grasp of global constraints. Secondly, discrete fields delineate the current player's role, the numerical values of their hand, the union of other players' hands, their discards, and remaining card counts, establishing a quantitative portrayal of the game's status. Thirdly, a sequential history of actions, organized by temporal order and player role, facilitates the model's tracking of game dynamics. Finally, the legal action space is explicitly enumerated as a nested list of all possible card plays (including singles, pairs, three-of-a-kind, straights, bombs, etc.), with each action directly represented as a numerical array. The salient advantages of this compositional design lie in its separation of rules and state, which enhances scalability; its numerical representation of card faces (e.g., J=11, Red King=30), which eliminates suit ambiguity and simplifies comparisons; the pre-computation of legal actions, which prevents the model from generating invalid outputs; and the integration of historical and remaining card counts, which provides a robust data foundation for strategic inference, such as card tracking and hand strength evaluation. Collectively, this input format balances human readability with machine parsability, making it suitable for training and evaluating "DouDizhu" agents based on RL or heuristic search.

Figure 6: Input Prompt of DouDizhu

```
You FIRST think about the reasoning process as an internal monologue
    and then provide the final answer.
The reasoning process MUST BE enclosed within <think> </think> tags.
    The final answer MUST BE in json format.
You are now a player in a game of DouDizhu (Fight the Landlord). The
    game rules are as follows:

1. The game is played by three players with a standard 54-card deck
    including a red joker and a black joker.
2. There are three roles in the game: landlord, landlord_down (farmer
    down of landlord), and landlord_up (farmer up of landlord).
3. After bidding, one player becomes the "landlord" who receives an
    extra three cards. The other two players are the "peasants" who
    work together to defeat the landlord.
4. In each round, the starting player must play a card or a valid
    combination of cards.
5. The other two players can choose to either follow with a higher-
    ranked card or combination, or pass.
6. If two consecutive players pass, the round ends and the player with
     the highest rank in that round starts the next round.
7. The objective of landlord is to be the first player to get rid of
    all the cards in hand.
8. The objective of landlord_down or landlord_up is to be the first
    player to get rid of all the cards in hand or to help the teammate
     to get rid of all the cards in hand.

The cards and comparison are as follows:
1. Individual cards are ranked. Red Joker > Black Joker > 2 > Ace (A)
    > King (K) > Queen (Q) > Jack (J) > 10 > 9 > 8 > 7 > 6 > 5 > 4 > 3.

2. The Rocket (Red Joker and Black Joker) and the Bomb are groups of
    cards that work differently in terms of game play.
3. Compare only the same Category. Compare only the Chains with the
    same length. Compare the rank in the Primal cards only. Jokers and
     2 are non-consecutive cards.
4. The type of card combination: Solo, Solo Chain (5), Pair, Pair
    Chain (3), Trio, Trio Chain (2), Trio with Solo, Trio Chain with
    Solo, Trio with Pair, Trio Chain with Pair, Bomb, Four with Dual
    solo, Four with Dual pair.
5. For ease of comparison, all cards are represented numerically: 3-10
     remain as their face value. Red Joker, Black Joker, 2, A, K, Q,
    and J are represented by 30, 20, 17, 14, 13, 12, and 11,
    respectively.
```

```
Your task is to make the best decision in each playing round. I will
    provide you with the following information:

Turn number:
6

1. Your role:
"landlord"

2. Your current hand cards:
[6, 7, 8, 9, 10, 11, 11, 12, 12, 12, 14, 14, 14, 14]

3. The union of the hand cards of the other two players:
[4, 5, 5, 6, 7, 11, 11, 12, 13, 13, 13, 17, 20, 5, 6, 7, 8, 8, 10, 10,
     13]

4. The played cards so far for each player:
{"landlord": [3, 3, 3, 9, 9, 30], "landlord_up": [4, 6, 8, 10], "
    landlord_down": [4, 4, 17, 17, 17, 3, 5, 7, 9]}

5. The number of cards left for each player:
{"landlord": 14, "landlord_up": 13, "landlord_down": 8}

6. The number of bombs played so far:
0

7. The historical actions of all players (left-to-right: earliest to
    latest; order: landlord, landlord_down, landlord_up):
[[], [3], [4], [], [5], [6], [], [7], [8], [], [9], [10], [30], [],
    []]

8. The legal actions for the current move:
[[6], [7], [8], [9], [10], [11], [12], [14], [11, 11], [12, 12], [14,
    14], [12, 12, 12], [14, 14, 14], [14, 14, 14, 14], [6, 12, 12, 12],
     [6, 14, 14, 14], [7, 12, 12, 12], [7, 14, 14, 14], [8, 12, 12,
    12], [8, 14, 14, 14], [9, 12, 12, 12], [9, 14, 14, 14], [10, 12,
    12, 12], [10, 14, 14, 14], [11, 12, 12, 12], [11, 14, 14, 14], [12,
     14, 14, 14], [12, 12, 12, 14], [11, 11, 12, 12, 12], [11, 11, 14,
     14, 14], [12, 12, 14, 14, 14], [12, 12, 12, 14, 14], [6, 7, 8, 9,
     10], [7, 8, 9, 10, 11], [8, 9, 10, 11, 12], [6, 7, 8, 9, 10, 11],
     [7, 8, 9, 10, 11, 12], [6, 7, 8, 9, 10, 11, 12], [6, 7, 14, 14,
    14, 14], [6, 8, 14, 14, 14, 14], [6, 9, 14, 14, 14, 14], [6, 10,
    14, 14, 14, 14], [6, 11, 14, 14, 14, 14], [6, 12, 14, 14, 14, 14],
     [7, 8, 14, 14, 14, 14], [7, 9, 14, 14, 14, 14], [7, 10, 14, 14,
    14, 14], [7, 11, 14, 14, 14, 14], [7, 12, 14, 14, 14, 14], [8, 9,
    14, 14, 14, 14], [8, 10, 14, 14, 14, 14], [8, 11, 14, 14, 14, 14],
     [8, 12, 14, 14, 14, 14], [9, 10, 14, 14, 14, 14], [9, 11, 14, 14,
     14, 14], [9, 12, 14, 14, 14, 14], [10, 11, 14, 14, 14, 14], [10,
    12, 14, 14, 14, 14], [11, 11, 14, 14, 14, 14], [11, 12, 14, 14, 14,
     14], [12, 12, 14, 14, 14, 14], [11, 11, 12, 12, 14, 14, 14, 14]]

Please tell me what cards you want to play in JSON format based on the
     provided information. The JSON should contain an "action" key
    with a value chose from legal actions.
If you choose to play cards, the value should contain the array of
    cards you want to play; if you choose to pass, the value should be
     empty array.

Output format examples:
Playing cards:
<think>
[Your reasoning here]
```

```
</think>
{"action": [Your chosen action here]}
Passing:
<think>
[Your reasoning here]
</think>
{"action": []}

Please provide the corresponding JSON action based on the given
    information.
```

Generated CoT Output The model's superior performance in complex card game decision-making is primarily attributed to its integrated architecture, which unifies structured state parsing, explicit chain-of-thought reasoning, and execution-ready outputs. Initially, the model efficiently extracts state features from modular, unambiguous input representations, incorporating player roles, current hand cards, opponents' remaining card distributions, historical action sequences, and pre-calculated legal action spaces. It then achieves precise evaluation of card strengths via numerical encoding. Subsequently, prior to action generation, the model explicitly outputs a natural-language CoT reasoning trace covering key strategic dimensions—such as positional awareness, card suppression, tempo control, bomb risk mitigation, and tactical play. This establishes a transparent and traceable decision trail conducive to offline error analysis and strategy optimization. Finally, the model strictly respects the legal action space, outputting a directly executable action array in `JSON` format, thereby eliminating formatting ambiguities and illegal action risks associated with post-processing. In multi-player, incomplete-information environments like DouDizhu, this architecture successfully harmonizes strategic depth, decision transparency, and engineering robustness, offering a scalable paradigm for transparent decision agent systems.

Figure 7: Generated CoT Output of DouDizhu

```
<think>
Given the current game state, I am the landlord, and I have a hand
    consisting of several 12s, 14s (Jokers), and a mix of other cards.
     My opponents have a mix of low to mid-range cards, and the
    landlord_down has played a 9 and a 10, while the landlord_up has
    played a 10. The landlord has played a 3 and a 9, and the
    landlord_down has played a 9. The landlord_up has played a 10 and
    a 30 (red joker).

Considering the cards I have, the best strategic move is to play a
    sequence of cards from 6 to 10. This move is significant for
    several reasons:

1. **Strategic Positioning**: Playing a sequence of cards (6, 7, 8, 9,
     10) is a strong move that can put pressure on the landlord_down
    and landlord_up. It creates a high-ranking played sequence that is
     hard to match or beat, especially considering the remaining cards
     in their hands.

2. **Card Distribution**: The landlord_down and landlord_up have
    relatively lower cards (4, 5, 6, 7, 8, 9, 10, and 11), and playing
     a higher sequence can force them to respond with higher cards if
    they choose to follow. The landlord, who has played a 3 and a 9,
    is less likely to have a higher sequence that can beat this play.

3. **Controlling the Game Flow**: By playing a high-ranking sequence,
    I can control the flow of the game. If the landlord_down or
    landlord_up decide to follow with a sequence, they might have to
    play higher cards, which I can then play again to continue putting
     pressure on them.
```

```
4. **Avoiding Immediate Bomb Threat**: Since there are no bombs played
    yet, playing a sequence gives me the flexibility to respond to
   any bomb that might be played. If the landlord_down or landlord_up
    have a bomb, they would need to play it against this sequence,
   which could be disadvantageous for them.

5. **Psychological Aspect**: Playing a high-ranking sequence can also
   have a psychological impact on my opponents. They might be
   hesitant to play a higher card if they think it will be beaten by
   a sequence.

In conclusion, playing [6, 7, 8, 9, 10] is a strategic move that
   leverages the current game state and my hand composition to put
   pressure on both opponents and control the game flow. It is a
   strong and calculated move that aligns with the objectives of the
   landlord role.

Chosen Action: {"action": [6, 7, 8, 9, 10]}
</think>
{"action": [6, 7, 8, 9, 10]}
```

E LIMITATIONS

Our study presents several limitations that suggest avenues for future research. First, the model's performance trajectory remains bounded by the quality and diversity of the initial expert trajectories, which may constrain policy exploration beyond human domain knowledge. Second, we have not yet evaluated the framework's robustness across different prompt formulations, leaving the sensitivity of CoT generation to prompt variations as an open question. Third, while the generated CoT rationales display strong logical consistency, their strategic depth in exceptionally rare game states warrants further refinement. Future work will investigate data-free self-play evolutionary mechanisms, explore prompt-agnostic reasoning strategies, and develop more lightweight parameterized frameworks.

F BROADER IMPACT

Our work evaluates the learning capabilities of large language models through strategic card games, focusing on improving decision transparency and decision-making performance. This research does not involve generating harmful content or privacy violations. However, training LLMs to play strategically in competitive, zero-sum games may foster self-interested decision-making tendencies that optimize individual goals at the expense of broader cooperative outcomes. We therefore encourage future work to pair strategic capability development with explicit safeguards for cooperative and socially aligned behavior. On balance, this research contributes to the safe and transparent deployment of LLMs in complex decision-making environments.