UNO Arena for Evaluating Sequential Decision-Making Capability of Large Language Models

Zhanyue Qin[1], **Haochuan Wang**[1], **Deyuan Liu**[1], **Ziyang Song**[1], **Cunhang Fan**[2], **Zhao Lv**[2], **Jinlin Wu**[3,4], **Zhen Lei**[3,4,5], **Zhiying Tu**[1], **Dianhui Chu**[1], **Xiaoyan Yu**[6], **Dianbo Sui**[1] *

[1] Harbin Institute of Technology, [2] Anhui University,
[3] CAIR, HKISI-CAS, [4]CASIA, [5]UCAS,
[6] Beijing Institute of Technology
✉johnneyqin@gmail.com, suidianbo@hit.edu.cn

Abstract

Sequential decision-making refers to algorithms that take into account the dynamics of the environment, where early decisions affect subsequent decisions. With large language models (LLMs) demonstrating powerful capabilities among various tasks, we cannot help but ask: *Can Current LLMs Make Sequential Decisions Effectively?* In order to answer this question, we propose the UNO Arena based on the card game UNO for evaluating the sequential decision-making capability of LLMs and explain in detail why we choose the UNO game. In the UNO Arena, we also involve some novel metrics based on Monte Carlo methods for evaluating the sequential decision-making capability of LLMs dynamically. Besides, we set up random players, DQN-based reinforcement learning players, and LLM players (e.g. GPT-4, Gemini-pro) for comparison testing. Furthermore, in order to improve the sequential decision-making capability of LLMs, we propose the **TUTRI** player, which can involve enabling LLMs to reflect on their actions with the summary of game history and the game strategy. Various experimental results demonstrate that the **TUTRI** player can achieve a notable breakthrough in the performance of sequential decision-making compared to the vanilla LLM player. [1]

1 Introduction

In artificial intelligence, sequential decision-making refers to algorithms that take the dynamics of the world into consideration (Frankish and Ramsey, 2014), and it can be described as a procedural approach to decision-making, or as a step by step decision theory. As a consequence, sequential decision-making has the intertemporal choice problem, where earlier decisions influences the later available choices (Amir, 2014).

In recent years, Large language models (LLMs) are gaining increasing popularity in both academia and industry, owing to their unprecedented performances in various applications (Chang et al., 2023), ranging from chatbots to medical diagnoses (Wang et al., 2023a) to robotics (He et al., 2022). From robots handling complex tasks (Amiri et al., 2020) to entrepreneurial action (McMullen, 2015), sequential decision-making permeates diverse domains. Hence, an interesting question arises: *Can Current LLMs Make Sequential Decisions Effectively?*

To answer this question, we need to design a benchmark to evaluate the sequential decision-making ability of LLMs. However, evaluating LLMs' abilities is not trivial. Many studies have been proposed to test LLMs' performances on either a large-scale static benchmark such as MMLU (Hendrycks et al., 2021), or with A/B tests judged by humans (Ganguli et al., 2023). One common and evident limitation of these methods, however, is that the environment for LLMs to be tested is static (Aiyappa et al., 2023; Zhou et al., 2023), which can not reflect the domino effect in sequential decision-making. Besides, data contamination (Sainz et al., 2023; Zeng et al., 2024; Xu et al., 2024), which means the inclusion of test data examples and labels in the pre-training data, also challenges the efficacy of these static benchmarks in differentiating model capabilities.

Unlike static evaluation, dynamic evaluation by treating LLMs as game-playing agents attracted more and more attention of researchers recently, such as beauty contests and private-value second price auctions (Guo et al., 2024a), Warewolf (Xu et al., 2023), Avalon (Wang et al., 2023b; Light et al., 2023) and Leduc Hold'em (Guo et al., 2023). However, current attempts do not account for sequential decision-making, and these games are either challenging to evaluate for intermediate results (such as Werewolf) or have too few decision points

*Dianbo Sui is the corresponding author.

[1]The code is publicly available at: https://github.com/JohnneyQin/UNO-Arena.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 7630–7645
November 12-16, 2024

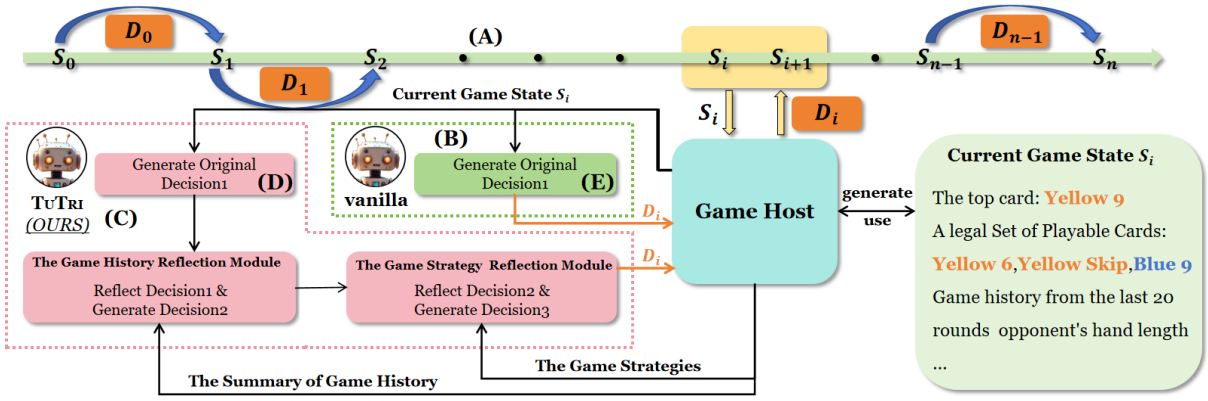

Figure 1: In this figure, (A) demonstrates the sequential decision-making process in the UNO Arena, (B) presents the execution process of the vanilla LLM player, and (C) shows the execution process of the TuTri player. Note that, the Module (D) and the Module (E) are completely identical.

per round (such as Leduc Hold'em). Meanwhile, we should also note that studies of dynamically evaluating sequential decision-making capability in the reinforcement learning, such as games like Go (Silver et al., 2017), Dou Di Zhu (You et al., 2019), and Mahjong (Li et al., 2020). However, these games present an excessively large action space. For instance, in Dou Di Zhu, players can use any combination of their cards in each round, posing significant challenges for current LLMs (Zhai et al., 2024).

Considering the above aspects, we make the following efforts in this paper:

First, we build the UNO Arena to dynamically evaluate the sequential decision-making capability of current LLMs. In the UNO Arena, we allow LLMs to participate as players in the UNO game[2], aiming to play all the cards in their hand as quickly as possible. Compared to games like Leduc Hold'em, which have fewer moves per game, UNO features an average of dozens of moves per game, making it an ideal testbed for sequential decision-making (Pfann, 2021). Additionally, unlike common games in the reinforcement learning, legal actions in the UNO Arena are limited, only including drawing cards, playing cards, selecting colors to convert, and choosing whether to challenge the wild draw four card. Furthermore, to monitor the behaviours of LLMs in the UNO Arena, we propose some real-time quantitative evaluation metrics by leveraging the Monte Carlo method (Kroese et al., 2014) as the reference, which provide a window to observe the intermediate results and various phenomenons (like domino effect) in LLMs' sequential decision-making.

Second, based on the proposed UNO Arena, we set up a family of strong and representative players. In detail, we first build the random player, which makes decisions based on chance rather than a specific, consistent plan, without considering the game's current state or potential outcomes. Then, we built heuristic players based on a conservative heuristic UNO play strategy and implement the reinforcement learning based player, which leverages DQN (Mnih et al., 2013) to develop sophisticated strategies for playing UNO. Finally, to probing the capability of LLMs in sequential decision-making, we provides the task description and then prompt LLMs, like GPT-4 (Achiam et al., 2023) and Gemini-pro (Team et al., 2023), to generate their reasoning steps that lead to the final action.

Third, to unleash the fully potential capability of LLMs in sequential decision-making, we propose the **TuTri** player with reflection mechanism (Shinn et al., 2024), which can enable LLMs to analyze their own actions based on the game history and the game strategy. In detail, the proposed agent framework consists of two key reflection modules: the game history reflection module and the game strategy reflection module. In the game history reflection module, we provide the statistical data of game history and then prompt the LLMs to rethink their decision, which simulates the process of card memorization by humans when playing UNO. In the game strategy reflection module, LLMs further take into account the game strategy, like saving wild draw four, and proceed to make the final decision, which simulates the use and adherence to strategies by humans when playing UNO.

[2] https://en.wikipedia.org/wiki/Uno_(card_game)

In the experiment, we comprehensively evaluate some mainstream LLMs' ability of sequential decision-making, including GPT-3.5 (OpenAI, 2022), GPT-4 (Achiam et al., 2023), Gemini-pro (Team et al., 2023), Llama 2 (Touvron et al., 2023), and ChatGLM3 (Du et al., 2021; Zeng et al., 2022). Our experimental results show that among these LLMs, GPT-4 is the most effective sequential decision-maker.

In summary, our contributions are as follows:

- We propose a dynamic evaluation method, named UNO Arena, for assessing the sequential decision-making capability of large language models (LLMs) based on the card game UNO. This method supports the evaluation of 2-10 LLM players, reinforcement learning players, heuristic players, or random players engaged in a single UNO game.
- We introduce multiple unique evaluation metrics based on the Monte Carlo method for evaluating the sequential decision-making capabilities of players in the UNO Arena.
- To improve the sequential decision-making capabilities of LLMs and enhance their performance in the highly dynamic and complex UNO game, we have developed the **TUTRI** player and compared it horizontally with the vanilla LLM player.

2 Related Work

Evaluate LLMs Dynamically with Game: LLMs have presented increasingly emerging ability on game-playing (Brookins and DeBacker, 2023; Akata et al., 2023) in recent development and iterations. Wang et al. (2023b) use the Avalon, which contains elements of deception, to evaluate the capability of LLMs to recognize and handle deceptive information. Gong et al. (2023) leverage the Cuisine World and Minecraft to assess the planning and emergency cooperation capabilities of LLMs. Guo et al. (2024a) employ beauty contests and auction games to evaluate the rationality, strategic reasoning capability, and adherence to instructions of LLMs. Xu et al. (2023) use the Werewolf game to evaluate the capability of LLMs to infer player roles. Despite evaluating with game becoming a popular trend, exploring into sequential decision-making capability is still of scarcity in current works.

Development of Agent Framework: LLM agents have been perceived as a promising way to realizing Artificial General Intelligence(AGI) (Xi et al., 2023) and recently have shown emergent abilities to execute various tasks in the complex environment (Wei et al., 2022). SiLLM (Guo et al., 2024b) merges large language models with synchronous machine translation, using policy decision agents and translation agents. LLM-Vectorizer (Taneja et al., 2024) uses multiple agents to generate vectorized code by leveraging large language models and test-based feedback. We tailor a special framework for UNO in this paper, featuring self-refinement and iterative thinking.

Sequential Decision-Making Capability: Sequential decision-making refers to the process of making a series of decisions over time, where each decision may impact future choices and outcomes (Amir, 2014). Though certain algorithms or reinforcement learning provide solutions for some sequential decision-making problems (Littman, 1996), LLM-based sequential decision-making are only employed in limited field like recommendation (Wang et al., 2023c). In our work, we utilize the UNO game, which is not an easy one even for human (Demaine et al., 2014), to explore the sequential decision-making ability of LLMs. With certain methods like integrating past experiences and expert advice or demonstrations (Chen et al., 2023), we make efforts to maximally leverage the decision making ability as possible in a sequential manner.

3 The UNO Arena

In this section, we first provide a brief overview of the version of UNO we adopt in the subsection §3.1. Then, we present the four different types of players in the UNO Arena in the subsection §3.2. Next, we detail how to use Monte Carlo methods to determine whether a player has made an optimal decision in subsection §3.3. In the end, we introduce our evaluation metrics in subsection §3.4.

3.1 The UNO Game

We select the UNO as the foundation within our arena due to its widespread popularity, simplicity and mathematical value. There are various versions of the UNO game. In this section, we briefly introduce the rules of the version we adopt in this work.

UNO Cards: A deck of UNO cards comprises a total of 108 cards. UNO cards are divided into three types: number cards, function cards, and wild

cards. A number card is composed of a color (Red, Blue, Yellow and Green) and a number (ranging from 0 to 9). A function card is composed of a color (Red, Blue, Yellow and Green) and a function (Skip, Reverse, Draw Two). The wild cards has no color and is only composed of Wild cards and Wild Draw Four cards. The effects of the function cards and wild cards are shown in the Table 1.

UNO Process: First, deal each player 7 initial cards in clockwise order, then continue drawing cards until a number card is drawn and set as the top card of the initial discard pile. All players take rounds playing cards in clockwise order (it will be reversed by a reverse card) until a player runs out of his cards or the draw pile is exhausted, signaling the end of the game.

UNO Action: From the beginning to the end of the game, players continuously take actions in UNO. In our work, UNO includes the following types of actions:

- **Select Card**: When a player comes his playing round, he need to play a card that matches the color, number, or function of the top card in the discard pile, or play a Wild card. If he does not have a card to play, he must draw one card.
- **Select Color**: After a player plays a Wild card or a Wild Draw Four card, the player need to change the color of the current top card to one of Red, Yellow, Blue or Green.
- **Select ChallengeFlag**: After a player's previous opponent plays a Wild Draw Four card, the player need to decide whether to challenge the legality of the previous opponent's Wild Draw Four card.

For more details about the UNO games, please refer to Appendix A. The Figure 1 (A) shows the workflow diagrams of UNO Arena.

3.2 Players in the UNO Arena

In the UNO Arena, we initially involve three types of players: random player, reinforcement learning based player, vanilla LLM player. To further unleash the potential capability of LLMs in sequential decision-making, we propose TUTRI player, which involves reflection mechanism.

Random Player: As like its name suggests, the random player performs all actions randomly, such as randomly selecting a regulative card to play when it is his turn. The random player can be considered the baseline of the UNO Arena, mainly serving to maintain the flow of the UNO game. If some players outperform the random player, we can infer that these players are consciously playing UNO with an understanding of the game rules.

Card	Sample	Effect
Skip		The next player in sequence misses a round.
Reverse		Order of play switches directions (clockwise to counterclockwise, or vice versa).
Draw Two		The next player in sequence draws two cards and misses a round.
Wild		Player declares the next color to be matched (it can be used on any round even if the player has any card of matching color).
Wild Draw Four		Player declares the next color to be matched. The next player in sequence draws four cards and misses a round. May be legally played if the player has cards of the current color.

Table 1: The effects of function and wild cards.

Heuristic Player: The heuristic player here is a player who follows the following strategy: (1) The number cards are matched first, and since the numbers 0 and 9 are half the number of other number cards, 0 number cards and 9 number cards have higher priority than other number cards. (2) Keep the function cards as much as possible, first match the type of function cards, and the priority is: Reverse cards, Skip cards and Draw Two cards. (3) Keep the wild cards as much as possible, and choose the most color in your hand when using the wild card to change the color of the top card in the current discard pile.

Reinforcement Learning Based Player: Previous research has sought breakthroughs in the UNO game by using reinforcement learning models (Pfann, 2021). We built our reinforcement learning player with DQN (Mnih et al., 2013) model based on the open-source project RLcard (Zha et al., 2019).

Vanilla LLM Player: During the vanilla LLM player's turn, the game host transmits all publicly available information through a prompt to the LLM. The LLM then returns a JSON containing the decision and reasoning as required by the prompt. The Figure 1 (B) shows the workflow diagrams of vanilla LLM player.

TUTRI Player: While LLMs do not always generate the best output on their first try just as hu-

man (Madaan et al., 2023), iterative feedback and refinement could be a necessity for a better agent framework. Moreover, human-like thinking patterns, such as introspective reflections foster divergent thinking processes (Zhang et al., 2023), inspire us to propose the TUTRI player. This advanced framework is designed to navigate the intricacies of UNO game play, offering a more structured approach to strategic sequential decision-making. The original decision for TUTRI player is exactly the same as the vanilla LLM player's decision, after that are two additional reflection modules.

- **The Game History Reflection Module**: In this module, we provide statistical information about game history to the TUTRI player, and the player is told to *reflect the action you just selected* with these auxiliary information. Just like human thinking when playing UNO, if there is a large number of green cards that have already been played in the game's history, it is very advantageous for the player to play a green card. After reflection, the player need to output both reflection thoughts and the updated action.
- **The Game Strategy Reflection Module**: In this module, we provide additional useful game strategies to the TUTRI player, and the player is again told to *reflect the action you just selected* based on game strategies. For example, since wild cards can be played at any situations and disrupt other players, saving the wild cards in your hand as long as possible is a very useful game strategy. After reflection, the player should output both reflection thoughts and updated action (the final action).

It must be emphasized that the TUTRI player should work in a conversational manner, with exactly 3 times Q&A per round. Moreover, the TUTRI player may keep their original decision, in other words, literally updating the action is not a necessity, nevertheless, the reflection process, instead of simple I-O prompting of interaction, providing more opportunities for mistake correcting and divergent thinking. The Figure 1(C) shows the workflow diagrams of TUTRI player.

3.3 Monte Carlo Simulation Method for Monitoring Players' Behavior

In the game play, the change in each player's winning rate after making a decision is the key for tracking. In the classical combinatorial games, like Nim (Bouton, 1901) or Wythoff's Game (Wythoff, 1907), positions space are limited and thus computationally affordable, while UNO is more intricate, where positional space exponentially increases as cards number increases and the calculation gets tougher (Demaine et al., 2014).

To make a plausible ranking mechanism of the candidate decisions, we define the concept of **optimal decision**, meaning the state transferred by the decision from last state, has a highest winning rate concerning all subsequent outcomes, and thus adopt Monte Carlo Simulation (Mooney, 1997) to calculate the estimated winning rate.

Detailedly, with S_i representing the state of the game after the i-th step taken, $D_{i,j}$ representing the j-th legal decision candidates at state S_i, $\mathcal{T}$ representing the state transfer function, $\mathcal{E}$ representing the estimate function of state, thereby we have the definition of the optimal decision $D_{i,opt}$ at the i-th step where

$$opt = \arg\max_j \mathcal{E}(\mathcal{T}(S_{i-1}, D_{i,j})) \quad (1)$$

In calculation of $\mathcal{E}(S_i)$, we massively randomly generate the subsequent decision sequence $\{D_{i+1}, D_{i+2}, \dots\}$ and thus obtain the subsequent state sequence $\{S_{i+1}, S_{i+2}, \dots\}$. Then $\mathcal{E}(S_i)$ is assigned to the ratio of number of sequences where the player plays the state S_{i-1} comes as the winner, to the total number of sequences simulated.

As the times we simulate the subsequent sequence increases, the approximate value $\mathcal{E}(S_i)$ gets more precise, though we could not enumerate all the possible situations. To balance the time expenditure and the precision of the metrics, we control the simulation times in a certain range. Additionally, a threshold parameter p is set to identify critical decisions. We say a decision $D := D_i$ is **critical** if among its all candidate choices D_j

$$\max \mathcal{E}(\mathcal{T}(S, D_j)) - \min \mathcal{E}(\mathcal{T}(S, D_j)) \geq p \quad (2)$$

Actual decisions made on critical positions may have a huge effect on the winning rate, which is consistent with the game nature.

3.4 Evaluation Metrics in the UNO Arena

In our work, we design three evaluation metrics, including WR, ODHR@K and ADR@K, in conjunction with the UNO game to comprehensively evaluate the sequential decision-making capability of LLMs. Among these metrics, ODHR@K and

ADR@K can off a glimpse into the intermediate results in the sequential decision-making of LLMs.

Winning Rate (WR). WR denotes the proportion of player wins to total game innings, and can be represented as:

$$\text{WR} = \frac{N_{\text{Win}}}{N_{\text{Game}}} \tag{3}$$

where N_{Win} represents the total number of times the given player has won, and N_{Game} denotes the total game innings.

Optimal Decision Hit Rate at K Decision Points (ODHR@K) : This metric measure the proportion of times players make the best decision to all decision times, when facing K decision points:

$$\text{ODHR@K} = \frac{N_{\text{Hit@K}}}{N_{\text{Decision@K}}} \tag{4}$$

where $N_{\text{Hit@K}}$ is the number of times the player makes the optimal decision when facing K optional decision points, and $N_{\text{Decision@K}}$ represents the total number of times the agent player makes decision when it faces K optional decision points.

Average Decision Rank at K Decision Points (ADR@K). This metric looks at the rank of output decision made by the player, and can be denoted as:

$$\text{ADR@K} = \frac{\sum_{i=1}^{N_{\text{Decision@K}}} \text{Rank}(D_i)}{N_{\text{Decision@K}}} \tag{5}$$

where $\text{Rank}(D_i)$ represents the rank from best to worst among all legal decisions in its decision-making process, and $N_{\text{Decision@K}}$ represents the total number of times the agent player makes decision when it faces K optional decision points.

For metrics ODHR@K and ADR@K, according to the characteristics of UNO, we only focus on the situations where K is equal to 2, 3 or 4, because the vast majority of decisions in UNO do not exceed 4 (Pfann, 2021).

4 Experiments

In this section, we first conduct preliminary experiments with vanilla LLM players, RL players, and random players in subsection §4.1. Then, we have multiple different LLM-based vanilla players compete in UNO Arena to identify the best LLM in subsection §4.2. Next, we test the superiority of the TUTRI players compared to the vanilla LLM players in subsection §4.3. Finally, we perform ablation experiments on the TUTRI player in subsection §4.4.

Metrics	Vanilla LLM Players & RL Player with DNQ					
	GPT-3.5	GPT-4	Gemini-Pro	Llama 2	ChatGLM3	DNQ
WR (↑)	55.80	**63.20**	53.80	53.60	48.80	57.40
ODHR@2 (↑)	57.34	**61.47**	53.94	53.69	49.75	54.96
ADR@2 (↓)	1.427	**1.385**	1.461	1.463	1.503	1.450
ODHR@3 (↑)	32.15	**39.30**	34.42	33.84	34.45	35.98
ADR@3 (↓)	2.010	**1.904**	2.017	1.994	2.034	1.947
ODHR@4 (↑)	27.20	**36.99**	31.05	27.39	25.36	37.74
ADR@4 (↓)	2.399	**2.142**	2.331	2.436	2.460	2.247

Table 2: Statistical results of random player VS vanilla LLM player or RL player with DNQ. The decision threshold p for critical decision in ODHR@K and ADR@K is 0.15. Bold indicates the best result, underline the second best result, and the Table 3 below follows this pattern.

To ensure the generalization of the experiments, we take the mainstream LLMs mentioned in the introduction: (1) gpt-3.5-turbo-16k-0613 (OpenAI, 2022); (2) gpt-4-1106-preview (Achiam et al., 2023); (3) Gemini-pro (Team et al., 2023); (4) Llama-2-7b-chat (Touvron et al., 2023); (5) ChatGLM3-6b (Du et al., 2021; Zeng et al., 2022).

4.1 1v1 UNO Arena between vanilla LLM players, RL players and random players

In order to verify the rationality of using UNO Arena to evaluate the sequential decision-making ability of LLMs, we first conduct experiments on vanilla LLM players, RL players and random players in 1V1 UNO Arena. We randomly generate 500 sets of UNO initial decks. Each vanilla LLM player or RL player have to play with the random player in these 500 initial decks. In addition, the random players are the first to play cards in all games. The results are shown in the Table 2.

Metrics	Vanilla LLM Players				
	GPT-3.5	GPT-4	Gemini-Pro	Llama 2	ChatGLM3
WR (↑)	22.80	**24.20**	20.40	20.00	15.60
ODHR@2 (↑)	52.57	**54.77**	49.88	54.08	50.52
ADR@2 (↓)	1.474	**1.452**	1.501	1.459	1.495
ODHR@3 (↑)	39.56	**41.41**	33.14	34.78	33.13
ADR@3 (↓)	1.889	**1.885**	2.034	1.978	2.043
ODHR@4 (↑)	26.75	**29.03**	25.74	24.90	25.04
ADR@4 (↓)	2.407	**2.366**	2.516	2.471	2.477

Table 3: Statistical results of competition among 5 vanilla LLM players in UNO Arena. The decision threshold p for critical decision in ODHR@K and ADR@K is 0.00.

From the Table 2, we can find that (1) Except for

LLM	WR (↑)	ODHR@2 (↑)	ADR@2 (↓)	ODHR@3 (↑)	ADR@3 (↓)	ODHR@4 (↑)	ADR@4 (↓)
GPT-3.5 (vanilla)	48.00	53.05	1.4695	34.97	1.9508	34.47	2.2340
GPT-3.5 (TUTRI)	52.50 (+4.50%)	54.01 (+0.06%)	1.4599 (-0.06%)	43.13 (+8.16%)	1.8563 (-4.73%)	32.92 (-1.55%)	2.2667 (+1.09%)
GPT-4 (vanilla)	49.00	56.27	1.4373	39.38	1.9375	36.24	2.2140
GPT-4 (TUTRI)	51.00 (+2.00%)	56.60 (+0.33%)	1.4340 (-0.33%)	40.14 (+0.76%)	1.8592 (-3.92%)	36.33 (+0.09%)	2.1510 (-2.10%)
Gemini-pro (vanilla)	44.00	50.62	1.4938	37.04	2.0159	25.44	2.4737
Gemini-pro (TUTRI)	56.50 (+12.50%)	53.64 (+3.02%)	1.4636 (-3.02%)	34.13 (-2.91%)	1.9461 (-3.49%)	30.36 (+4.92%)	2.3482 (-4.18%)
Llama 2 (vanilla)	47.00	49.54	1.5046	33.11	1.9595	29.11	2.3944
Llama 2 (TUTRI)	54.00 (+7.00%)	55.07 (+5.53%)	1.4493 (-5.53%)	37.31 (+4.20%)	1.8507 (-5.44%)	26.75 (-2.36%)	2.4650 (+2.35%)
ChatGLM3 (vanilla)	47.00	55.82	1.4418	29.05	2.0541	31.84	2.2935
ChatGLM3 (TUTRI)	54.00 (+7.00%)	57.24 (+1.42%)	1.4276 (-1.42%)	39.51 (+10.46%)	1.8642 (-9.50%)	30.62 (-1.22%)	2.4689 (+5.85%)

Table 4: Statistical results of vanilla LLM players VS TUTRI players. The decision threshold p for critical decision in ODHR@K and ADR@K is 0.00. Red annotations indicate favorable experimental results, while blue annotations indicate unfavorable experimental results.

LLM	WR (↑)	ODHR@2 (↑)	ADR@2 (↓)	ODHR@3 (↑)	ADR@3 (↓)	ODHR@4 (↑)	ADR@4(↓)
Gemini-pro (TUTRI)	56.50	53.64	1.4636	34.13	1.9461	30.36	2.3482
Gemini-pro + TUTRI′	52.50 (-4.00%)	54.33 (+0.69%)	1.4567 (-0.69%)	29.88 (-4.25%)	2.0610 (+5.75%)	31.25 (+0.89%)	2.4219 (+2.46%)
Gemini-pro + TUTRI″	53.50 (-3.00%)	54.59 (+0.95%)	1.4541 (-0.95%)	31.95 (-2.18%)	2.0384 (+4.62%)	27.59 (-2.77%)	2.3824 (-1.14%)

Table 5: Statistical results of the ablation study. Where TUTRI′ represents the TUTRI player which remove the game history reflection module, and TUTRI″ represents the TUTRI player which remove the game strategy reflection module. The decision threshold p for critical decision in ODHR@K and ADR@K is 0.15. Red annotations indicate favorable experimental results, while blue annotations indicate unfavorable experimental results.

ChatGLM3, the WR of other vanilla LLM players and RL players are all above 50.00%; (2) The performance of GPT-4 is the best, and GPT-4 performs excellently on the 7 evaluation metrics. Especially, the WR of GPT-4 is 63.20%, 13.20% higher than 50.00%.

4.2 5-players UNO Arena with 5 LLMs

To find the best LLM, we place 5 LLMs in a 5-players UNO Arena to compete against each other. We fix the initial playing order of UNO Arena in the sequence of GPT-3.5, GPT-4, Gemini-Pro, Llama 2, and ChatGLM3. We conduct experiment on 200 decks generated randomly. All players are the vanilla LLM players. The results are shown in the Table 3.

From the Table 3, we can find that (1) GPT-4 has the best performance, with a WR of 24.20%, 4.2% higher than the average (20.00%) and 1.4% higher than the second highest ranked GPT-3.5. Not only that, GPT-4 also performs the best in other 6 evaluation metrics; (2) ChatGLM3 has the worst performance, with a WR of 15.60%, which is 4.4% lower than the average (20.00%) and 8.6% lower than the highest ranked GPT-4. Not only that, ChatGLM3 also performs the worst in ODHR@2, ADR@2, ADR@3, ODHR@4, and ADR@4.

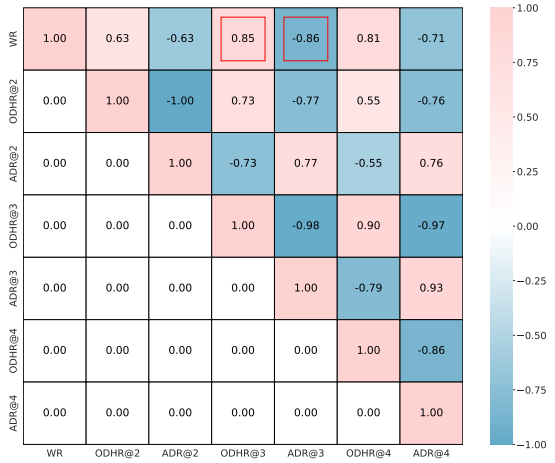

Figure 2: The Pearson Correlation Heatmap among WR, ODHR@K (K=2,3,4), and ADR@K (K=2,3,4).

4.3 Validation of the superiority of the TUTRI player compared to the vanilla LLM player

To verify that our TUTRI player can improve the sequential decision-making ability of LLMs, we compare the vanilla LLM players (baseline) with TUTRI players. We let 5 LLMs serve as the backend LLMs for both the vanilla LLM players and TUTRI players, and play two-players UNO Arena on 200 decks generated randomly. The results are shown in the Table 4.

From the Table 4, we can find that: (1) All LLMs (the TUTRI player) are better than LLM (the vanilla LLM player) on WR, ODHR@2, and ADR@3.

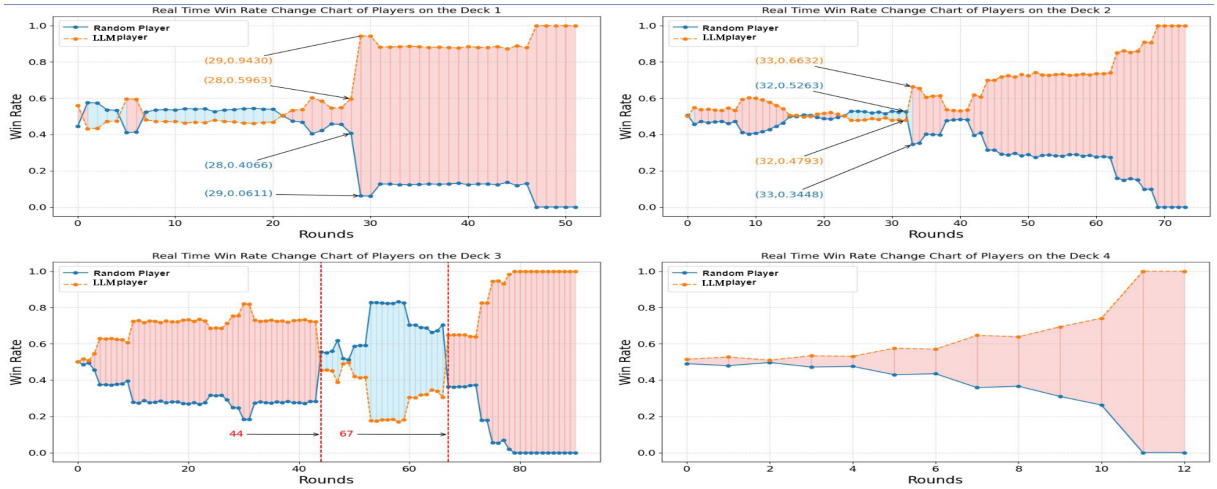

Figure 3: GPT-4 (the vanilla LLM player) real-time winning rate variations on 4 decks.

Gemini-Pro (the TUTRI player) has a 12.50% higher than Gemini-Pro (the vanilla LLM player) on WR; (2) For ODHR@3, except for Gemini-Pro which performed slightly worse (-2.91%), the other 4 LLMs achieved good results. For ODHR@4 and ADR@4, GPT-4 and Gemini-Pro both performed well. It can be seen that the TUTRI player based on reflection can significantly improve its abilities of sequential decision-making after two rounds of reflection on the summary of game history and the game strategies. The experimental results strongly support the superiority of our TUTRI player based on the reflection mechanism over the vanilla LLM player.

4.4 Ablation studies on TUTRI player

To illustrate the necessity of the two reflection modules in the TUTRI player, we further conduct some ablation studies. We remove the game history reflection module and the game strategy reflection module from the TUTRI players, and conduct two-players UNO Arena with vanilla LLM player respectively. The results are shown in the Table 5.

From the Table 5, we can find that: (1) After removing the game history reflection module, the WR of decreased by 4%, the ODHR@3 of decreased 4.25%, and the ADR@3 of increase by 5.75%. (2) After removing the game strategy reflection module, the WR of decreased by 3%, the ODHR@3 of decreased 2.18%. and the ADR@3 of increase by 4.62%. The game history holds significant potential information for incomplete information games. Therefore, removing the game history reflection module has a greater adverse impact on the TUTRI player.

5 Discussion

5.1 Further Exploration of ODHR@K and ADR@K

To better analyze the relationship between our unique evaluation metrics (ODHR@K and ADR@K), and the evaluation metric WR, we conduct a Pearson correlation analysis of the experimental results from the Table 2. The results are shown in the Figure 2. From the Figure 2, we can find that (1) WR shows a positive correlation with ODHR@K (K=2,3,4), and simultaneously, WR shows a negative correlation with ADR@K (K=2,3,4); (2) The strongest positive correlation, reaching 0.85, exists between WR and ODHR@3, while the strongest negative correlation, reaching -0.86, exists between WR and ADR@3. Overall, our unique ODHR@K and ADR@K have a good correlation with WR, so they can serve as reference evaluation metrics for evaluating LLMs in the UNO Arena.

5.2 Case Study

In order to more intuitively see the advantages of LLM versus random player, we conduct a case study. We utilized GPT-4 as the backend LLM for the vanilla LLM player to engage in the game across 4 decks generated randomly, with the random player plays first. We recorded all decision points (for both the vanilla LLM player and the random player) and employed the Monte Carlo method to calculate the real-time percentage change in winning rate for both sides following each decision point. The results are shown in the Figure 3.

From the Figure 3, we can find that: (1) In the

Player	WR (↑)	ODHR@2 (↑)	ADR@2 (↓)	ODHR@3 (↑)	ADR@3 (↓)	ODHR@4 (↑)	ADR@4(↓)
Heuristic Players	63.60	62.68	1.373	37.55	1.931	37.03	2.227
Random Players	36.40	37.12	1.630	28.92	2.153	16.64	2.615
Heuristic Players	60.80	59.74	1.402	36.37	1.932	34.81	2.252
Vanilla LLM Players	39.60	38.31	1.618	29.12	2.148	17.23	2.651

Table 6: The experimental results of heuristic players compared to random players and vanilla LLM players.

UNO Arena, winning rates fluctuate significantly. For example, in deck 1, from round 28 to 29, the random player's winning rate dropped by 34.5%, while the vanilla LLM player by 34.67%; (2) Turning points, like rounds 44 and 67 in deck 3, show shifts in dominance. Initially, the vanilla LLM player leads until round 44, then loses advantage until round 67, before regaining control; (3) Brief game durations occur, notably in deck 4, where the agent player consistently makes exceptional decisions, steadily increasing its winning rate until achieving victory. These findings underscore LLM's adeptness at identifying crucial decision junctures and exploiting its capabilities, highlighting its potential in sequential decision-making scenarios.

5.3 Exploring the Capabilities of Heuristic Players

We conduct two experiments: one where heuristic players face random players in 500 different UNO decks, and another where heuristic players compete against vanilla LLM players using Llama-2 in the same 500 decks. The experimental results are shown in the Table 6.

From the Table 6, we can find that: (1) Heuristic players significantly outperform random players across all evaluation metrics. (2) Heuristic players also demonstrate strong capabilities compared to vanilla LLM players (Llama 2), although the performance of vanilla LLM players is still superior to that of random players.

5.4 Concerns about Data Contamination

As previous studies suggest (Topsakal et al., 2024; Karvonen, 2024), we usually assume LLMs are not trained explicitly for games, like UNO and Tic-Tac-Toe. The LLMs know the description/rules of the games but the experimental results show that even the strongest LLM, like GPT-4, struggles. Besides, As we mention in the introduction section, there are static methods and dynamic methods (such as UNO Arena and Warewolf) in evaluating LLMs' ability. Compared to static methods, UNO Arena is less prone to data contamination due to vast game sampling space and vast game state (for a given game) space. Specifically, we sample 500 UNO games and find that among these 500 games, the maximum game state is 1.610^{34}, and the average game state is 5.310^{31}. Considering the enormous game state space and almost infinite game space, LLMs are extremely difficult to learn or overfit genuinely useful patterns or strategies. Therefore, we don't need to worry about data contamination potentially skewing the results.

6 Conclusion

In conclusion, LLMs possess the capability for sequential decision-making, as evidenced by the experimental results of LLMs playing the UNO game. Our proposed UNO Arena and unique evaluation metrics enable LLMs to compete with each other in the same UNO Arena game, thereby providing a better dynamic assessment of LLMs' sequential decision-making abilities. Furthermore, we propose that the TUTRI player effectively addresses how to enhance LLMs' sequential decision-making abilities for better performance in playing UNO Arena.

Acknowledgements

This work is supported by the National Key R&D Program of China (Grant No. 2023YFB3307500). This work is also supported by the National Natural Science Foundation of China (Grant No. 62306087, No. 62472121 and No.62306313), the Natural Science Foundation of Shandong Province (Grant No. ZR2023QF154), Special Funding Program of Shandong Taishan Scholars Project, the Open Project of Anhui Provincial Key Laboratory of Multimodal Cognitive Computation, Anhui University (Grant No. MMC202420) and the InnoHK program. Besides, we sincerely thank the anonymous reviewers for their valuable feedback.

Limitations

The method of dynamically evaluating the sequential decision-making ability of LLMs using the UNO Arena, as well as the TUTRI player, is only applicable to LLMs that support *chat*. The unique evaluation metrics, ODHR@K and ADR@K, introduced in this paper are only applicable to games or tasks with a limited action space.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Rachith Aiyappa, Jisun An, Haewoon Kwak, and Yong-Yeol Ahn. 2023. Can we trust the evaluation on chatgpt?

Elif Akata, Lion Schulz, Julian Coda-Forno, Seong Joon Oh, Matthias Bethge, and Eric Schulz. 2023. Playing repeated games with large language models. arXiv preprint arXiv:2305.16867.

Eyal Amir. 2014. Reasoning and decision making. The Cambridge handbook of artificial intelligence, pages 191–212.

Saeid Amiri, Mohammad Shokrolah Shirazi, and Shiqi Zhang. 2020. Learning and reasoning for robot sequential decision making under uncertainty. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 2726–2733.

Charles L Bouton. 1901. Nim, a game with a complete mathematical theory. The Annals of Mathematics, 3(1/4):35–39.

Philip Brookins and Jason Matthew DeBacker. 2023. Playing games with gpt: What can we learn about a large language model from canonical strategic games? Available at SSRN 4493398.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology.

Liting Chen, Lu Wang, Hang Dong, Yali Du, Jie Yan, Fangkai Yang, Shuang Li, Pu Zhao, Si Qin, Saravan Rajmohan, et al. 2023. Introspective tips: Large language model for in-context decision making. arXiv preprint arXiv:2305.11598.

Erik D Demaine, Martin L Demaine, Nicholas JA Harvey, Ryuhei Uehara, Takeaki Uno, and Yushi Uno. 2014. Uno is hard, even for a single player. Theoretical Computer Science, 521:51–61.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. Glm: General language model pretraining with autoregressive blank infilling. arXiv preprint arXiv:2103.10360.

Keith Frankish and William M Ramsey. 2014. The Cambridge handbook of artificial intelligence. Cambridge University Press.

Deep Ganguli, Nicholas Schiefer, Marina Favaro, and Jack Clark. 2023. Challenges in evaluating AI systems.

Ran Gong, Qiuyuan Huang, Xiaojian Ma, Hoi Vo, Zane Durante, Yusuke Noda, Zilong Zheng, Song-Chun Zhu, Demetri Terzopoulos, Li Fei-Fei, et al. 2023. Mindagent: Emergent gaming interaction. arXiv preprint arXiv:2309.09971.

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

Shangmin Guo, Haoran Bu, Haochuan Wang, Yi Ren, Dianbo Sui, Yuming Shang, and Siting Lu. 2024a. Economics arena for large language models. arXiv preprint arXiv:2401.01735.

Shoutao Guo, Shaolei Zhang, Zhengrui Ma, Min Zhang, and Yang Feng. 2024b. Sillm: Large language models for simultaneous machine translation. arXiv preprint arXiv:2402.13036.

Zexue He, Yu Wang, Julian McAuley, and Bodhisattwa Prasad Majumder. 2022. Controlling bias exposure for fair interpretable predictions. arXiv preprint arXiv:2210.07455.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

Adam Karvonen. 2024. Emergent world models and latent variable estimation in chess-playing language models.

Dirk P Kroese, Tim Brereton, Thomas Taimre, and Zdravko I Botev. 2014. Why the monte carlo method is so important today. Wiley Interdisciplinary Reviews: Computational Statistics, 6(6):386–392.

Junjie Li, Sotetsu Koyamada, Qiwei Ye, Guoqing Liu, Chao Wang, Ruihan Yang, Li Zhao, Tao Qin, Tie-Yan Liu, and Hsiao-Wuen Hon. 2020. Suphx: Mastering mahjong with deep reinforcement learning. arXiv preprint arXiv:2003.13590.

Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. 2023. Avalonbench: Evaluating llms playing the game of avalon.

Michael Lederman Littman. 1996. Algorithms for sequential decision-making. Brown University.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. arXiv preprint arXiv:2303.17651.

Jeffery S McMullen. 2015. Entrepreneurial judgment as empathic accuracy: A sequential decision-making approach to entrepreneurial action. Journal of Institutional Economics, 11(3):651–681.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

Christopher Z Mooney. 1997. Monte carlo simulation. 116. Sage.

OpenAI. 2022. Introducing chatgpt. https://openai.com/blog/chatgpt. Accessed: 2023-09-30.

Bernhard Pfann. 2021. Tackling the uno card game with reinforcement learning. Towards Data Science: tackling-uno-card-game-with-reinforcement-learning.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In Findings of the Association for Computational Linguistics: EMNLP 2023, pages 10776–10787, Singapore. Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems, 36.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. nature, 550(7676):354–359.

Jubi Taneja, Avery Laird, Cong Yan, Madan Musuvathi, and Shuvendu K Lahiri. 2024. Llm-vectorizer: Llm-based verified loop vectorizer. arXiv preprint arXiv:2406.04693.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. arXiv preprint arXiv:2312.11805.

Oguzhan Topsakal, Colby Jacob Edell, and Jackson Bailey Harper. 2024. Evaluating large language models with grid-based game competitions: An extensible llm benchmark and leaderboard.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971.

Sheng Wang, Zihao Zhao, Xi Ouyang, Qian Wang, and Dinggang Shen. 2023a. Chatcad: Interactive computer-aided diagnosis on medical image using large language models. arXiv preprint arXiv:2302.07257.

Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. 2023b. Avalon's game of thoughts: Battle against deception through recursive contemplation.

Yu Wang, Zhiwei Liu, Jianguo Zhang, Weiran Yao, Shelby Heinecke, and Philip S Yu. 2023c. Drdt: Dynamic reflection with divergent thinking for llm-based sequential recommendation. arXiv preprint arXiv:2312.11336.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. arXiv preprint arXiv:2206.07682.

Willem A Wythoff. 1907. A modification of the game of nim. Nieuw Arch. Wisk, 7(2):199–202.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The rise and potential of large language model based agents: A survey. arXiv preprint arXiv:2309.07864.

Cheng Xu, Shuhao Guan, Derek Greene, and M-Tahar Kechadi. 2024. Benchmark data contamination of large language models: A survey.

Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. 2023. Exploring large language models for communication games: An empirical study on werewolf.

Yang You, Liangwei Li, Baisong Guo, Weiming Wang, and Cewu Lu. 2019. Combinational q-learning for dou di zhu. arXiv preprint arXiv:1901.08925.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. arXiv preprint arXiv:2210.02414.

Zhongshen Zeng, Pengguang Chen, Shu Liu, Haiyun Jiang, and Jiaya Jia. 2024. Mr-gsm8k: A meta-reasoning revolution in large language model evaluation.

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

Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. 2024. Fine-tuning large vision-language models as decision-making agents via reinforcement learning.

Jintian Zhang, Xin Xu, and Shumin Deng. 2023. Exploring collaboration mechanisms for llm agents: A social psychology view. arXiv preprint arXiv:2310.02124.

Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. 2023. Don't make your llm an evaluation benchmark cheater.

Appendix

A UNO Game

In this section of appendix, you will learn what UNO Game is, and in order to facilitate the evaluation of LLMs with the UNO Game, we have made some slight modifications to it.

A.1 Game Objective

In the modified UNO game, We simply set the game objective as to be the first player to clear out the hand. Players play alternatively(2 players) or in circle manner(3 or more players) and strive to achieve the unique goal. It should be noted that if the cards in the deck are exhausted by players, the player with the fewest number of cards in hand wins, so there may be multiple winners in the same game.

A.2 UNO Cards

UNO comprises 3 categories of cards: number cards, function cards, and wild cards. In total, UNO features 108 cards.

- **The Number Cards**: the number cards can be expressed in the form of COLOR + NUMBER, where COLOR is belong to the set $\{Red, Blue, Yellow, Green\}$, and NUMBER is an integer from 0 to 9. It is important to note that there is only one 0-number card per color, while there are two 1-9 number cards per color. There are a total of 76 number cards.
- **The Function Cards**: the function cards can be expressed in the form of COLOR + FUNCTION, where COLOR is belong to the set $\{Red, Blue, Yellow, Green\}$, and FUNCTION is belong to the set $\{Skip, Reverse, DrawTwo\}$. There are two cards of the same COLOR for each FUNCTION. There are a total of 24 function cards.
  - **Skip**: the player's next player skips this round of play.
  - **Reverse**: the player reverses the order of play (from clockwise to counterclockwise, or from counterclockwise to clockwise).
  - **Draw Two**: The player's next player draws two cards and skips this round of play.
- **The Wild Cards**: the wild card includes 4 Black Wild cards and 4 Black Wild Draw Four cards, totaling 8 cards.
  - **Wild**: the player selects one color from the COLOR set $\{Red, Blue, Yellow, Green\}$ as the new color for the top card in the discard pile.
  - **Wild Draw Four**: the player selects one color from the COLOR set $\{Red, Blue, Yellow, Green\}$ as the new color for the top card in the discard pile, and the player's next player draws 4 cards.

A.3 Game Progress

First, deal each player 7 initial cards in clockwise order, then continue drawing cards until a number card is drawn and set as the top card of the initial discard pile. All players take rounds playing cards in clockwise order(it will be reversed by a reverse card) until a player runs out of his cards or the draw pile is exhausted, signaling the end of the game.

A.4 Legal Decision(Action)

In every round of the game, player in charge can using rules to match the top card of the discard pile otherwise pick up a new card into hand. The rules, or say, the legal decisions consists of several sorts:

- **Draw Card**: If a player does not have any cards to play during their playing round, they must draw a card, or their previous

player used Draw Two or Wild Draw Four cards to make the player draw multiple cards.

- **Select Card**: In a player's playing round, they need to play a card that matches either the COLOR, NUMBER, or FUNCTION of the top card in the discard pile, or play a Wild card(include Wild Draw Four card) to match. The card played by the player then becomes the new top card.
- **Select Color**: After selecting either Wild Card or Wild Draw Four Card, the player needs to convert the color of the current top card to one of $\{Red, Blue, Yellow, Green\}$.
- **Select ChallengeFlag**: The use of the Wild Draw Four card may be illegal. After a player plays a Wild Draw Four card, their next player can choose to challenge its use. If the player who played the Wild Draw Four card still holds non-Wild cards matching the color of the current top card, the use of the Wild Draw Four card is illegal. Possible scenarios are as follows: (1) If the player's play is illegal and their next player challenges it, the player must draw 4 cards, and their next player faces no penalty; (2) If the player's play is legal and their next player challenges it, the player's next player must draw 6 cards. (3) If the player's next player does not challenge, regardless of the legality of the player's play, the player's next player must draw 4 cards. Note that Challenge is not a stand-alone action to complete turns, it should be accompanied by a card draw or card match action.

B Prompt

Here is the prompt design for the entire experiment.

B.1 Select Card

The input1 prompt of the select card shared by the vanilla LLM player and the TUTRI player is shown in the Figure 4. The game history Reflection module prompt of the select card for the TUTRI player is shown in the Figure 5. The game strategy reflection module prompt of the select card for the TUTRI player is shown in the Figure 6.

B.2 Select Color

The input1 prompt of the select color shared by the vanilla LLM player and the TUTRI player is shown in the Figure 7. The game history reflection module prompt of the select color for the TUTRI player is shown in the Figure 8. The game strategy reflection module prompt of the select color for the TUTRI player is shown in the Figure 9.

B.3 Select ChallengeFlag

The input1 prompt of the select ChallengeFlag shared by the vanilla LLM player and the TUTRI player is shown in the Figure 10. The game history reflection module prompt of the select ChallengeFlag for the TUTRI player is shown in the Figure 11. The game strategy reflection module prompt of the select ChallengeFlag for the TUTRI player is shown in the Figure 12.

Select Card Input1 Prompt

You are playing a two-player UNO game.
- You are the player{player_id}, and your opponent is the player{opponent_id}.
- Currently, there are {len_deck} cards in the deck, and the discard pile has {len_discard_pile} cards.
- The number of cards in the hand of your opponent is {len_opponent_hand}.
- The game history of the last {len_history} rounds is {history}.
- Your entire hand consists of {hand}.
- The cards you can play are: {playable_card}.

Please note that you are not playing a normal UNO game, if the cards in the deck are depleted, the person who has the minimum cards will win directly. The goal of the game is to minimize the number of cards in your possession. In order to win the UNO game, you must consider all the provided information and select the best card from the cards you can play.

The output should strictly be a JSON object with two keys: 'thoughts' and 'action'.
- In this context, the value corresponding to the 'thoughts' key represents your thoughts and considerations, with its data type being a string.
- Simultaneously, the value corresponding to the 'action' key is a int which represents the card index of the card you have selected.

Figure 4: The input1 prompt of the select card shared by the vanilla LLM player and the TUTRI player.

Select Card Reflection1 Prompt

Here is the statistical data of the game history:
{history_summary}
Now, in order to win the game, you must consider the statistical data carefully and reflect the action you just selected.
You should strictly output a JSON object with two keys: 'reflection' and 'action'.
- The value corresponding to the 'reflection' key is your reflection.
- The value corresponding to the 'action' key is the card index you currently select.

Figure 5: The game history reflection module prompt of the select card for TUTRI player.

Select Card Reflection2 Prompt

Here is an useful tip that you can follow:
- The card values range from low to high, starting with number cards 0, followed by number cards (1-9), reverse cards, skip cards and wild cards.
- It is better to start with low-value cards before playing high-value cards.
- Unless your opponent is on the verge of victory, it is time to play some high-value cards to disrupt your opponent's strategy.

Now, in order to win the game, you should reflect the action you just selected based on the tip.
You should strictly output a JSON object with two keys: 'reflection' and 'action'.
- The value corresponding to the 'reflection' key is your reflection.
- The value corresponding to the 'action' key is the final card index you currently select.

Figure 6: The game strategy reflection module prompt of the select card for TUTRI player.

Select Color Input1 Prompt

You are playing a two-player UNO game.
- You are the player{player_id}, and your opponent is the player{opponent_id}.
- Currently, there are {len_deck} cards in the deck, and the discard pile has {len_discard_pile} cards.
- The number of cards in the hand of your opponent is {len_opponent_hand}.
- The game history of the last {len_history} rounds is {history}.
- Your entire hand consists of {hand}.

Please note that you are not playing a normal UNO game, if the cards in the deck are depleted, the person who has the minimum cards will win directly. The goal of the game is to minimize the number of cards in your possession. In order to win the UNO game, you just played a {wild_type} card, and you must consider all the provided information and select the best color from Red, Yellow, Blue and Green to switch.
The output should strictly be a JSON object with two keys: 'thoughts' and 'action'.
- In this context, the value corresponding to the 'thoughts' key represents your thoughts and considerations, with its data type being a string.
- Simultaneously, the value corresponding to the 'action' key is one of Red, Yellow, Blue or Green, indicating the color you have selected.

Figure 7: The input1 prompt of the select color shared by the vanilla LLM player and the TuTri player.

Select Color Reflection1 Prompt

Here is the statistical data of the game history:
{history_summary}
Now, in order to win the game, you must consider the statistical data carefully and reflect the action you just selected.
You should strictly output a JSON object with two keys: 'reflection' and 'action'.
- The value corresponding to the 'reflection' key is your reflection.
- The value corresponding to the 'action' key is the color you currently select.

Figure 8: The game history reflection module prompt of the select color for TuTri player.

Select Color Reflection2 Prompt

Here are some useful tips that you can follow:
- It is better to select the color with the highest frequency of occurrence in your hand.
- It is better to avoid selecting the color with the lowest frequency of occurrence in your hand.
- Consider carefully which color of cards is relatively more frequent in your opponent's hand and try to avoid selecting that color.

Now, in order to win the game, you should reflect the action you just selected based on these tips.
You should strictly output a JSON object with two keys: 'reflection' and 'action'.
- The value corresponding to the 'reflection' key is your reflection.
- The value corresponding to the 'action' key is the final color you currently select."

Figure 9: The game strategy reflection module prompt of the select color for TuTri player.

Select ChallengeFlag Input1 Prompt

You are playing a two-player UNO game.
- You are the player{player_id}, and your opponent is the player{opponent_id}.
- Currently, there are {len_deck} cards in the deck, and the discard pile has {len_discard_pile} cards.
- The number of cards in the hand of your opponent is {len_opponent_hand}.
- The game history of the last {len_history} rounds is {history}.
- Your entire hand consists of {hand}.

Your opponent played a Wild Draw Four card, and changed the color of the current discard pile's top card to {new_color}. But the use of the Wild Draw Four card may be illegal,when your opponent still has cards in {old_color}.Please note that you are not playing a normal UNO game, if the cards in the deck are depleted, the person who has the minimum cards will win directly. The goal of the game is to minimize the number of cards in your possession. In order to win the UNO game, you must consider all the provided information and select whether to challenge the use of the Wild Draw Four card which played by your opponent.
The output should strictly be a JSON object with two keys: 'thoughts' and 'action'.
- In this context, the value corresponding to the 'thoughts' key represents your thoughts and considerations, with its data type being a string.
- Simultaneously, the value corresponding to the 'action' key is 'Yes' or 'No', indicating that you select to challenge or not to challenge, respectively."

Figure 10: The input1 prompt of the select ChallengeFlag shared by the vanilla LLM player and the TUTRI player.

Select ChallengeFlag Reflection1 Prompt

Here is the statistical data of the game history:
{history_summary}
Now, in order to win the game, you must consider the statistical data carefully and reflect the action you just selected.
You should strictly output a JSON object with two keys: 'reflection' and 'action'.
- The value corresponding to the 'reflection' key is your reflection.
- The value corresponding to the 'action' key is the choice you currently select.

Figure 11: The game history reflection module prompt of the select ChallengeFlag for TUTRI player.

Select ChallengeFlag Reflection2 Prompt

Here are some useful tips that you can follow:
- Please remember the penalty for a failed challenge: you must draw 6 cards.
- Please remember the benefits of a successful challenge: your opponent must draw 4 cards.
- Wild Draw Four is only illegal if your opponent has cards of {old_color} color in his hand.

Please carefully consider whether your opponent's Wild Draw Four card is genuinely illegal. Now, in order to win the game, you should reflect the action you just selected based on these tips.
You should strictly output a JSON object with two keys: 'reflection' and 'action'.
- The value corresponding to the 'reflection' key is your reflection.
- The value corresponding to the 'action' key is the final choice you currently select.

Figure 12: The game strategy reflection module prompt of the select ChallengeFlag for TUTRI player.