Assessing Logical Puzzle Solving in Large Language Models: Insights from a Minesweeper Case Study

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have shown remarkable proficiency in language understanding and have been successfully applied to a variety of real-world tasks through task-specific fine-tuning or prompt engineering. Despite these advancements, it remains an open question whether LLMs are fundamentally capable of reasoning and planning, or if they primarily rely on recalling and synthesizing information from their training data. In our research, we introduce a novel task—Minesweeper—specifically designed in a format unfamiliar to LLMs and absent from their training datasets. This task challenges LLMs to identify the locations of mines based on numerical clues provided by adjacent opened cells. Successfully completing this task requires an understanding of each cell's state, discerning spatial relationships between the clues and mines, and strategizing actions based on logical deductions drawn from the arrangement of the cells. Our experiments, including trials with the advanced GPT-4 model, indicate that while LLMs possess the foundational abilities required for this task, they struggle to integrate these into a coherent, multi-step logical reasoning process needed to solve Minesweeper. These findings highlight the need for further research to understand and nature of reasoning capabilities in LLMs under similar circumstances, and to explore pathways towards more sophisticated AI reasoning and planning models.

1 Introduction

Large Language Models (LLMs) have made remarkable strides in the Natural Language Processing (NLP) arena, capturing the spotlight with their multifaceted capabilities. These models have been effectively utilized across a spectrum of NLP tasks, including information extraction (Agrawal et al., 2022; Zhu et al., 2023b), answering arithmetic and common-sense questions (Li et al., 2022; Yuan et al., 2023; Imani et al., 2023), as well as aiding in strategic planning (Yao et al., 2023; Wang et al., 2023c) or acting as game agents (Callison-Burch et al., 2022; Wang et al., 2023a; Gupta, 2023). With growing scale, LLMs begin to exhibit "emergent abilities" (Wei et al., 2022a; Schaeffer et al., 2023), including marked improvements in their ability to follow instructions, perform multi-step reasoning and planning, and even extend to comprehending humor (OpenAI, 2023). Such flourishing capability has triggered excitement within the research community, cultivating the belief that LLMs could be instrumental in achieving Artificial General Intelligence (Bubeck et al., 2023; Zhang et al., 2023; Tu et al., 2023). Nevertheless, a crucial question remains underexplored: *To what extent can the reasoning capabilities of LLMs go beyond the scope of their training distribution?*

This question is critical as it uncovers if LLMs are reliable for sophisticated hands-off planning in various scenarios such as biomedical experimental design (Bran et al., 2023; O'Donoghue et al., 2023) and autonomous driving (Sha et al., 2023), where the agents have to deal with off-distribution inputs occasionally. Unfortunately, the answer is not straightforward, and the debate is going on (Huang et al., 2022; Helbling et al., 2023; Wu et al., 2023; Huang et al., 2023; Valmeekam et al., 2023; Stechly et al., 2023). The reason is that LLMs' substantial parameter counts enable them to store and recall vast amounts of information from their training material (Gudibande et al., 2023; Rosenfeld et al., 2020; Kaplan et al., 2020; Brown et al., 2020), which may contain conventional reasoning datasets such as GSM8K (Cobbe et al., 2021), MultiArith (Roy and Roth, 2015), or StrategyQA (Geva et al., 2021). This could exaggerate LLMs' reasoning abilities on such benchmarks (Qin et al., 2023; Deng et al., 2023), leading to false a promise when applying LLMs to practical scenarios.

Although attempts to develop new benchmarks that challenge LLMs suggest potential zero-shot

learning successes (Suzgun et al., 2023; Bao et al., 2023), there is an underlying concern that these benchmarks may not be distinct enough from the LLMs' training data, thereby giving LLMs an undue advantage and masking their true abilities (Valmeekam et al., 2022). For instance, Gudibande et al. (2023) suggest that smaller LLMs only learn specific tasks when fine-tuned with data from larger models and that these improvements do not generalize well. Drawing an analogy between smaller and larger LLMs to humans implies that LLMs may not exhibit genuine intelligence or intrinsic reasoning beyond their training distribution (Liu et al., 2023; Bubeck et al., 2023). To robustly claim that LLMs possess reasoning abilities, they must be tested under a wider array of conditions. A fundamental approach to overcoming these challenges is to decouple the test inputs from the LLMs' pre-training data, compelling them to rely on task descriptions for reasoning and planning rather than prior knowledge. This entails an environment with clear instructions, a rich but well-defined action space, and a distinct objective. Such a setup facilitates the verification of the reasoning process and qualitative performance assessment, while it also reduces the chances of successful random guessing.

Minesweeper, a well-known logic puzzle game illustrated in Figure 1, is proposed as an ideal testing ground under these considerations. The game presents a range of complexities: while its fundamental principles are straightforward and accessible to beginners, achieving proficiency requires advanced logical reasoning skills that go beyond mere pattern recognition on the game board. For LLMs, the challenge in Minesweeper lies in interpreting the states of individual cells, making use of numerical hints, and understanding the spatial interconnections between cells. This is essential for accurately determining the locations of mines and strategizing subsequent moves based on incomplete information——a process that is instinctive for human players. Moreover, there has been no known attempt to translate Minesweeper boards into textual formats prior to this study. This ensures that LLMs cannot depend on matching patterns from their pre-existing data repositories. Instead, they are required to apply the rules and principles of the game to logically infer and extrapolate solutions.

Building upon our initial motivation, we conduct a comprehensive set of experiments with LLMs including GPT-3.5-16k, GPT-3.5-instruct, and GPT-4 (OpenAI, 2022, 2023), probing their inherent reasoning skills. We introduce the Minesweeper game in various formats to determine how different types of input influence LLM performance. Objective measures like the count and proportion of accurate moves, correctly flagged mines, and completely resolved boards were employed to gauge effectiveness. However, a critical aspect of our evaluation is the manual examination of the LLMs' reasoning processes, inspecting the validity and logical soundness of their intermediate deductions. Our findings indicate that these models generally struggle to maintain consistent logical reasoning chains. The GPT-3.5 versions, in particular, tended to repeat information from provided examples or previous conversations, often failing to adjust to the updated board layouts. While GPT-4 showed improvements in response diversity, relevance, and coherence, it still faced issues with logical inconsistencies and incomplete reasoning. This research sheds light on the specific skills required for solving logical puzzles, underscoring the importance of these competencies in future LLM-focused studies. To support forthcoming research and comprehensive evaluation of LLMs, we have published our code and data at anonymous.4open.science/r/Minesweeper-for-LLM-781D. We hope it contributes to LLMs' broader understanding and future development.

2 Minesweeper

Minesweeper, a classic logic game, has been a part of the Windows operating system since around 1990s. Though the rule is simple, it features a rich and discrete action space, a clear goal, and direct feedbacks and evaluation metrics. The objective is straightforward: players must unveil all cells on a board peppered with hidden mines without detonating any. Each cell on the board can exist in one of four possible states (Figure 1b):

- **Unopened** ("?"): The state of these cells is unknown; they could transition into any of the other three states.
- **Numbered** ("1" to "8"): These cells reveal the count of adjacent mines, including diagonals, providing critical clues for safe navigation.
- **Blank** (". "): These cells have no adjacent mines, often opening up larger areas of the board.
- **Flagged** ("F"): Cells suspected to contain mines are marked accordingly for safety.

Notably, the choice of symbols is arbitrary and should not impact the model performance, as they serve as mere representations of concepts.

(a)

```
"0","1","2","3","4","5","6","7","8","9"
"1",`?',`?',`1',`.',`.',`.',`.',`1',`?'
"2",`?',`1',`1',`.',`.',`.',`.',`1',`?'
"3",`?',`1',`1',`1',`.',`.',`1',`2',`?'
"4",`?',`2',`F',`1',`.',`.',`1',`F',`?'
"5",`?',`2',`1',`1',`.',`1',`2',`2',`1'
"6",`1',`1',`1',`1',`1',`1',`F',`1',`.'
"7",`.',`.',`1',`F',`1',`1',`1',`1',`.'
"8",`.',`.',`2',`?',`3',`2',`1',`1',`.'
"9",`.',`.',`1',`?',`F',`?',`F',`1',`.'
```

(b)

```
(1,1): ?
(1,2): ?
(1,3): 1
(1,4): .
(1,5): .
(1,6): .
(1,7): .
(1,8): 1
(1,9): ?
(2,1): ?
(2,2): 1
(2,3): 1
(2,4): .
(2,5): .
(2,6): .
(2,7): .
(2,8): 1
(2,9): ?
(3,1): ?
(3,2): 1
(3,3): 1
(3,4): 1
(3,5): .
...
```

(c)

```
L(5,5)
R(6,7)
R(4,8)
M(5,7)
M(6,8)
R(4,3)
M(3,3)
M(5,4)
M(5,3)
M(6,6)
R(7,4)
...
```

(d)

Figure 1: An example of Minesweeper on a 9×9 board containing 10 mines, along with its interaction format. Subfigure 1a displays the game's GUI; Subfigure 1b shows a plain-text, table-formatted representation of the game board, enhanced with color for improved visualization; Subfigure 1c depicts the coordinate-based plain-text representation of the board; and Subfigure 1d provides a log of the player's (in this case, the first author's) actions, where "L", "R", and "M" denote left-click, right-click, and middle-click actions, respectively.

During gameplay, players may execute one of three actions per turn: left-click ("L") to reveal a cell's contents, right-click ("R") to flag a suspected mine, and middle-click ("M") to verify the correctness of flagged cells based on adjacent numbers. On a beginner's 9×9 board (Figure 1a), this results in 243 potential actions per round (Figure 1d), creating an extensive action landscape despite numerous actions being invalid (as listed in Table 3). The game provides immediate feedback for each action, with outcomes ranging from revealing or flagging cells to error messages, game endings or victories, allowing models to adjust their strategies in real-time. Victory is reached by correctly identifying and flagging all mines or revealing all non-mine cells, whereas failure occurs upon missteps such as erroneous left-click on mines or incorrect flag placements during middle-click verifications. With action space and game objective established, metrics such as the proportion of valid moves, the percentage of successfully solved games, and the average number of moves to resolve a board can be readily computed. In summary, Minesweeper serves as a less knowledge-intensive, more symbol-comprehension and basic math-reasoning benchmark, pivoting on spatial reasoning over pure information retrieval.

We employ two plain-text formats to present the game board when interacting with LLMs, aiming to understand LLMs' capabilities of comprehending each of them and to reduce the potential influence of input format on the reasoning process. The first format is **table representation**, depicted in Figure 1b. Here, the game board is portrayed as a table with the states of each cell enclosed in LaTeX-style quotation marks (grave accent "`" and apostrophe "'") to differentiate them from the cell indices and separators. Rows and columns are delineated by line break "\n" and comma "," respectively, with indices for the first row and column serving as the coordinates for the cells. This format is intuitive for human interpretation; LLMs equipped with capabilities to understand tables, as suggested by the literature (Chen, 2023; Singha et al., 2023), should process it similarly. For comparison, we also employ **coordinate representation** (Figure 1c). It explicitly associates each cell's coordinates with its state in the form of a look-up table, potentially offering a more direct description for LLMs to comprehend the board's configuration.

The following sections first explore two critical skills required for Minesweeper agents (§ 3). Then, they evaluate the proficiency of LLMs in playing Minesweeper, detailing objective scores and case studies that highlight their reasoning and planning capabilities (§ 4). Each section begins with a brief introduction of the experimental setup, with comprehensive details available in appendix B.

3 Board Understanding

3.1 Experiment Setup

A fundamental skill for LLMs in Minesweeper is the ability to comprehend the game board. To assess this capability, we conducted two types of straightforward experiments: **board navigation** and **neighbor counting**. In board navigation, LLMs are presented with a game board and a spe-

Task	Representation	3.5-16k	3.5-instr
Board Navigation	table	66.7	53.3
	− ids	63.7	40.7
	+ example	64.3	47.0
	+ verification	66.3	-
	coordinate	82.3	66.7
	+ example	**89.7**	61.0
Neighbor Counting	table	33.3	16.7
	coordinate	**37.3**	30.3

Table 1: Comparative analysis of GPT-3.5 variants in board understanding tasks. The performance are quantified using exact-matching accuracy percentages. The notation "− ids" indicates the omission of indices in the table representation, while "+ example" denotes the inclusion of additional examples in the prompts. The "+ verification" symbol refers to the application of the self-verification technique as described in (Weng et al., 2022). As GPT-3.5-instruct is not optimized for conversational contexts, the self-verification is not applicable.

cific coordinate, and they are tasked to identify the corresponding cell state. For neighbor counting, LLMs are given an additional arbitrary state from the set {`?', `.', `1', `2',} and are required to calculate the number of occurrences of this state (ranging from 0 to 8) surrounding the provided coordinate. We annotated 100 randomly generated 9×9 boards, each containing 10 mines, with complete action steps (refer to Figure 1d). The boards were randomized at different stages of the game, either fully or partially revealed, and 3 sets of coordinates were selected randomly for the experiments.

3.2 Results

The performance of GPT-3.5-16k and GPT-3.5-instruct was evaluated using both table and coordinate representations, as shown in Table 1. With the table representation, GPT-3.5-16k demonstrated a notable deficiency in accurately identifying the cell state, failing approximately one-third of the time in board navigation tasks. GPT-3.5-instruct showed even lower performance. Attempts to enhance performance through various modifications to the boards and prompts, including removing indices, adding examples, and requesting models to revise their outputs, were unsuccessful. However, with the coordinate representation, both models exhibited improved results. This suggests that the challenge of interpreting tables might be a limiting factor for the models in subsequent gameplay tests. Nonetheless, GPT-3.5 displayed a failure rate of over 10%, which is suboptimal given that the task essentially involves copying and pasting values in coordinate-state mappings. In neighbor counting tasks, both models achieved a maximum accuracy of about one-third, significantly better than random guessing. This indicates that GPT-3.5 models possess basic arithmetic and geographical planning skills, which are essential for Minesweeper. These findings suggest that the models have a moderate level of board understanding, which is sufficient to not severely impede the gameplay tests that follow.

4 Minesweeper Gameplay

4.1 Experiment Setup

In our study, we utilize 5×5 Minesweeper boards containing 4 hidden mines, a setup less complex than the conventional beginner level. LLMs are instructed to start the game by left-clicking the center cell and then proceed with either left, right, or middle-click actions on cells they assess as beneficial for advancing gameplay. We craft and annotate 100 boards for GPT-3.5 models, ensuring the initial left-click would reveal at least 10 cells (40% of the board). This strategy aims to streamline the length of the prompts and minimize the number of interactive rounds required. For GPT-4, we randomly subsample 10 boards from the pool to reduce the experiment time and expense.

Prompting techniques such as chain-of-thought reasoning (Wei et al., 2022b) and few-shot in-context learning (appendix B.3) are employed to activate the LLMs' reasoning capabilities and provide sufficient information for decision-making. Each game session permits a maximum of 10 actions per board, concluding either when the model achieves victory, triggers a mine, or produces unrecognizable responses. Given that human annotators, on average, complete each board with an average of 6.14 actions, we deemed a 10-action limit adequate for the model to effectively resolve most boards, contingent on its reasoning ability.

To balance the influence of prompt details and accommodate models with shorter prompt constraints, our study introduces two distinct prompting approaches: 1) **Natural Conversation** (NC) mode, and 2) **Compact History** (CH) mode, as illustrated in Figure 2. In the NC mode, each game commences with a detailed explanation of rules and examples. During each round, we prompt LLMs to execute actions along with their associated reasoning. The game system then provides feedback on these actions and updates the board in a new user

	Human	3.5-16k (NC)		3.5-16k (CH)		3.5-instr (CH)		4 (NC)	
	GUI	Table	Coord	Table	Coord	Table	Coord	Table	Coord
Game Outcome									
# Total Games	100							10	
• % Solved (↑)	89.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
• % Failed (↓)	11.0	17.0	35.0	17.0	35.0	19.0	41.0	70.0	70.0
Mine Identification									
# Total Mines	400							40	
• % Flagged (↑)	93.0	2.8	11.7	8.0	9.5	11.7	23.3	30.0	45.0
Action Accuracy									
# Total Actions	514	825	679	700	687	813	642	41	52
• % Valid (↑)	99.6	7.2	22.4	15.7	22.6	24.1	64.6	82.9	82.7
• % Repeated (↓)	5.1	43.6	44.6	42.3	45.3	39.4	19.9	0.0	0.0
Reasoning Soundness									
# Selected Reasoning Chains	-	34	38	45	39	45	45	25	38
• % Valid Actions (↑)	-	61.8	81.1	46.7	59.0	80.0	95.6	88.0	84.2
• % Accurate and Coherent Logic (↑)	-	0.0	0.0	0.0	0.0	0.0	0.0	12.0	26.4

Table 2: The comparison of GPT model variants, prompting modes, and board representation formats on 5×5 boards with 4 mines. "#" indicates the number of instances; "%" represents the percentage ratio. The initial action, `L(3,3)`, is consistently excluded from the count. GPT-3.5-instruct is prompted using the compact history mode, as it is not optimized for chat-based interactions. The terms "Table" and "Coord" in the table refer to the table-format and coordinate-format representations of the Minesweeper board, respectively.

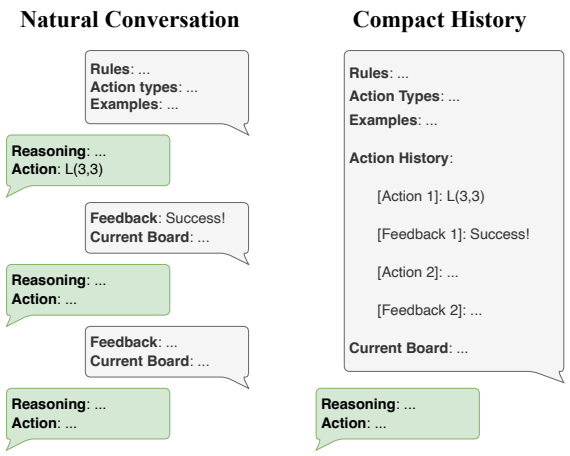

Figure 2: Interaction prompting modes. The "Natural Conversation" mode encompasses the full interaction history, whereas the "Compact History" mode condenses the actions generated by the LLM and the game's feedback into a succinct, unified prompt.

message, while maintaining the complete history of the interaction. CH mode, on the other hand, condenses all relevant information and interaction history into a concise, singular prompt, eliminating the need for extensive conversational details. This approach significantly shortens the length of the conversation history. Given the turn-based nature of Minesweeper, the CH mode is effective in providing all vital information required for the model to make well-informed decisions.

4.2 Metrics

The evaluation of our model's performance is anchored in objective metrics, which encompass: the count of valid actions, the accurate identification of mines, the number of successfully completed games, and the logical coherence of reasoning chains. A "valid" action is one that advances gameplay without activating a mine. Table 3 details a broad spectrum of negative examples. The metric for correctly flagged mines is based on the quantity of mines identified by the model that align with the actual mine locations at the game's conclusion. A game is deemed "solved" when all mines are accurately marked and no extraneous cells are flagged. To assess the reasoning chains, we initially select 5 game boards where each model executed the highest number of valid actions. We then perform a manual inspection to ensure these chains are logically sound and directly pertinent to the gameplay. Due to the variable action counts across different models and boards, the totals of actions and reasoning chains differ, as Table 2 illustrates.

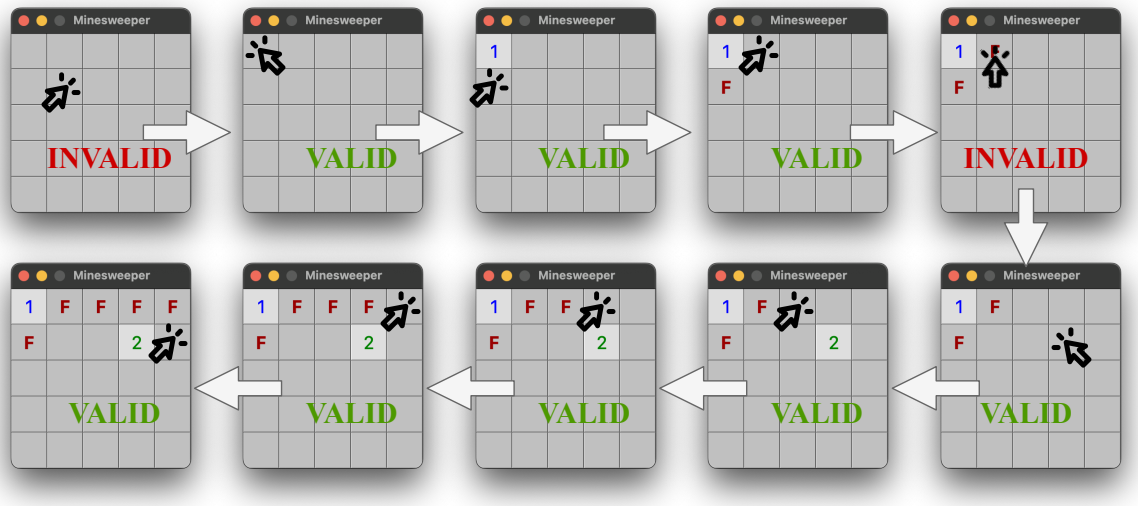

Figure 3: A detailed analysis of the example interactions performed by GPT-3.5-instruct. Arrows oriented to the left and right signify left and right mouse clicks, respectively. The arrow pointing upwards represents a middle-click. The majority actions are technically allowed but do not effectively advance the gameplay.

4.3 Results and Discussion

Table 2 showcases the performance comparisons between various GPT family models and a human benchmark, focusing on valid actions, correctly identified mines, completed games, and logical reasoning chains. In line with the findings on board comprehension, models utilizing coordinate representation exhibit superior performance across all metrics compared to table representation. This supports our previous assertion that LLMs struggle with tables containing symbols not found in their training data, preferring familiar in-domain terminology as evidenced in the studies by Chen (2023) and Singha et al. (2023).

Objective Scores In assessing specific gameplay metrics, we first focus on three main areas: Game Outcomes, Mine Identification, and Action Accuracy. Among the GPT-3.5 variants, 3.5-instruct demonstrates better overall performance compared to 3.5-16k in these metrics, albeit with a marginally increased failure rate, which diverges from the board understanding results. A detailed analysis of GPT-3.5-instruct's *initial* move reveals a consistent anomaly: the model persistently selects the `R(2,2)` action, contrary to the given instruction of `L(3,3)` (refer to Table 3 for invalid actions and Listing 5, Line 83 for the specific prompt). After receiving feedback, it switches to the `L(1,1)` action. This action often uncovers fewer cells in our specially selected boards, usually just the chosen cell itself

```
>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`.',`1',`?',`?',`?'
"2",`.',`1',`2',`4',`?'
"3",`.',`.',`.',`1',`1'
"4",`.',`.',`.',`.',`.'
"5",`.',`.',`.',`.',`.'
```

```
>> ASSISTANT (gpt-3.5-turbo-16k):
```
REASONING:
Looking at the current board, we can see that the cell at (2,4) has a number `4', indicating that there are four mines in its neighboring cells. Out of these neighboring cells, three are already opened, and one remains unopened at (1,4). Since there are four mines in total, it is logical to deduce that the unopened cell at (1,4) contains a mine. Therefore, our next move is to flag the cell at (1,4) with a right-click.

ACTION:
R(1,4)

Figure 4: A case study of a "valid" action and its corresponding reasoning generated by GPT-3.5-16k for solving Minesweeper. Blue indicates logical reasoning; red and golden are illogical ones.

(as illustrated in Figure 3), which increases the probability of the next action being valid. This behavior could also explain the higher failure rate and lower repeatance rate observed in GPT-3.5-instruct compared to the 3.5-16k model. Furthermore, GPT-3.5-instruct tends to make conservative moves by flagging all cells on the board, which accounts for its inability to solve any board despite having a higher ratio of flagged mines. Overall, all GPT-
```

3.5 models present a noticeable pattern of repetitive actions during board solving, persisting even with direct instructions to avoid repetition (see Listing 5, line 29). This repetitive behavior suggests a significant limitation in the models' capacity to fully understand and reason about the nuanced variations occurring on the board at each turn. As Transformer-based models designed primarily for next-token prediction (Vaswani et al., 2017), their focus likely remains on static aspects of cell states, overlooking subtle changes. This attentional bias, compounded by the lack of pre-training on tasks resembling this one, often leads to the repeated execution of identical actions.

In contrast, GPT-4 exhibits a marked improvement in following instructions, achieving a zero rate of repetition while maintaining a high frequency of valid actions. Notably, it flags the most mines, despite not solving any board and recording a high percentage of game failures, which is partly attributed to its tendency for divergent actions. When both operated in NC mode, GPT-4 is more prone to generating unrecognizable actions compared to GPT-3.5-16k, as illustrated in Figure 5, even when the conversation history is within the token limit. A plausible interpretation of this behavior is that GPT-4 demonstrates heightened sensitivity to recent changes in board states, allocating more attention to these immediate alterations. Conversely, it appears to pay less regard to the more distant historical context, where the constraints governing action formats are established.

Reasoning Chains The foregoing discussion highlights that merely counting valid actions is not an adequate measure of the models' reasoning skills. Therefore, we shift our focus to the logical coherence of their reasoning chains. A review of the final section in Table 2 reveals that the GPT-3.5 variants struggle to generate coherent reasoning. For instance, in a typical case from GPT-3.5-16k, shown in Figure 4, the model correctly identifies cell `4'at (2,4) and understands that there are four mines around, but fails to grasp the concept of "neighboring cells" or to accurately count the surrounding cell states, as indicated by the statement marked in red. Furthermore, the statement in golden font is not well-grounded, making it challenging to discern any logical connection between its first and second halves. This sentence appears more like an unsuccessful attempt to replicate our in-context example "*Given that there's just one adjacent mine, it's logical to deduce that the unopened cell at (2,3) contains the mine*" (refer to Listing 5, Line 51) rather than a display of genuine reasoning. While this example is specifically for GPT-3.5-16k, similar errors are prevalent across all GPT-3.5 outputs. It appears that despite possessing a certain degree of individual capabilities in elementary symbolic understanding, geographical reasoning, and arithmetic calculation, as suggested by the board understanding results, the GPT-3.5 models fall short in integrating these skills cohesively for action planning, resulting in a fragmented reasoning chain.

GPT-4 exhibits more encouraging results compared to its predecessors. While its achievements are modest, GPT-4 demonstrates a certain level of logical understanding in this unfamiliar task, suggesting an ability to grasp the game's rules and dynamics beyond mere replication. To explore the extent and origin of this reasoning, we delve deeper into GPT-4's reasoning chains, analyzing a representative case in Figure 5. It shows that GPT-4 can logically reason if the number of neighboring unopened cells equals the remaining unflagged mines near a numbered cell, specifically in simple scenarios like `1'and `2', and particularly around the board's corners (*e.g.*, (1,2) or (5,4)). Yet, even in such basic situations, GPT-4 shows around 50% fails. The model struggles particularly with configurations where exactly three unopened cells surround a `3'cell and other complex setups, as seen in the lower rows of the example. To determine if GPT-4's ability stems from the rules or the examples, we conducted a control experiment by omitting Example 1 from the initial prompt (Listing 5, Line 43), which closely mirrors the successful scenarios and is the sole example of left-clicking. This led to a dramatic 90% reduction in coherent reasoning with the coordinate board representation, indicating the significant role of examples in GPT-4's reasoning process, likely by providing a familiar context for reference and imitation.

Another notable point from Figure 5 is GPT-4's non-linear action sequence. For a human player, following the initial action of R(1,2), the logical next step would be to mark the adjacent unopened cell as a mine, *i.e.*, R(1,1). However, GPT-4 instead opts for R(5,1), an irrational choice that strays from the focus of the previous action, before returning to (1,1). This behavior suggests that GPT-4 might not fully consider its historical actions and lacks the capability for long-term, multi-

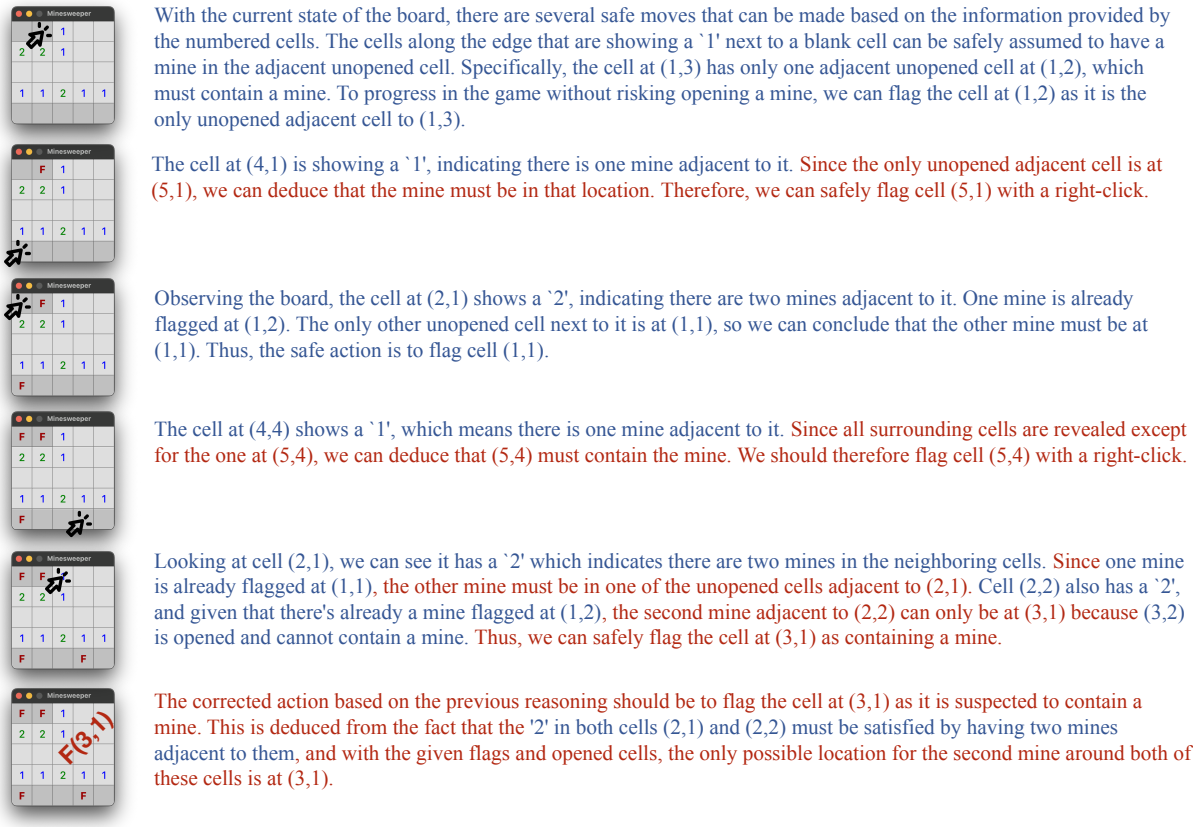

Figure 5: This figure presents a case study of the reasoning sequences formulated by GPT-4 with coordinate board representation during action planning. The GUI on the left shows the board states and actions taken by the agent during the game. Elements highlighted in blue represent accurate facts and logical inferences as assessed by human evaluation, while those marked in red indicate incorrect observations or illogical conclusions. Notably, the final generated action "`F(3,1)`" deviates from the permissible action formats, resulting in the termination of the game.

step planning, a crucial aspect in evaluating an agent's intrinsic reasoning abilities as opposed to mere summarization capabilities.

It is important to note that the results discussed were derived from experiments conducted on 5×5 boards, and it is reasonable to anticipate that performance may further deteriorate on larger boards with more flexible mine arrangements.

5 Conclusion

In this study, we assessed the capabilities of LLMs in solving logical puzzles by examining their performance in the game Minesweeper, a task for which they have received limited exposure during the pre-training phase. This approach aims to assess the intrinsic reasoning capabilities of LLMs. Our experiments reveal that GPT-3.5 models exhibit basic abilities in spatial navigation, symbol recognition, and counting. However, they struggle with multi-step planning and generating coherent reasoning chains. GPT-4 shows improvement in simpler scenarios with shorter logic chains but still encounters issues with hallucinations and inconsistent context awareness. We find that GPT-4 can learn from examples and apply this knowledge to similar situations but is less adept at deducing underlying rules and applying them to novel scenarios, unlike GPT-3.5 models which tend to replicate examples. Our conclusion seems to be in correlation with Yadlowsky et al. (2023): LLMs do not produce new knowledge (a situation where logical reasoning is necessary); they are just (good) handlers of existing experience.

We do not intend to diminish the significance of LLMs or their transformative impact on natural language processing. Rather, we believe it is premature to consider LLMs as an intelligent object and a threat to human society. They resemble a comprehensive dictionary, useful for reference but lacking in comprehension. We hope that our work can inspire future research on this topic, and we will continue exploring the capabilities of LLMs.

Limitations

In our study, we employed two board representation methods: table and coordinate representations. Our results indicate that coordinate representation is more effective for LLM comprehension and offers a wider range for evaluating the model's reasoning abilities. However, we did not explore other board representation methods, such as different table formats (*e.g.*, HTML, Markdown) or image-based approaches. According to Singha et al. (2023), alternative table formats may not significantly differ in effectiveness from our chosen method. Still, it's possible that other representations could be more suitable for our specific context. Moreover, our initial attempts to use an image-based Web interface with board GUI, a novel feature for GPT-4, were not successful. As discussed in § 2, the game's symbols are customizable, but we have not examined how altering these symbols might affect the model's performance.

Another area for exploration is fine-tuning LLMs like Llame-2-70b (Touvron et al., 2023) with specific reasoning capabilities on our dataset labeled with human actions. The idea is that if a fine-tuned model outperforms the original or GPT-4 when the fine-tuning and testing data match in terms of symbol usage, board size, and representation, and does not improve with variations in these factors, we can infer that the "reasoning capability" is tied to the training data and not broadly generalizable. However, due to constraints in computational resources, we haven't conducted these experiments, leaving them for future research.

At last, we would like to emphasize that we can easily extend the Minesweeper dataset to a scale of $\binom{n_{\text{row}} \times n_{\text{column}}}{n_{\text{mine}}}$ with zero effort. Nonetheless, we keep a relatively small scale of test cases as it is sufficient to support our conclusion. With the future development of LLMs, a larger evaluation set might be desired for a more robust and unbiased objective metric-based evaluation.

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1998–2022, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity.

Qiming Bao, Gaël Gendron, Alex Yuxuan Peng, Wanjun Zhong, Neset Tan, Yang Chen, Michael Witbrock, and Jiamou Liu. 2023. A systematic evaluation of large language models on out-of-distribution logical reasoning tasks. *CoRR*, abs/2310.09430.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. 2023. Chemcrow: Augmenting large-language models with chemistry tools.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712.

Chris Callison-Burch, Gaurav Singh Tomar, Lara Jean Martin, Daphne Ippolito, Suma Bailis, and David Reitter. 2022. Dungeons and dragons as a dialog challenge for artificial intelligence. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 9379–9393. Association for Computational Linguistics.

Wenhu Chen. 2023. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1120–1130, Dubrovnik, Croatia. Association for Computational Linguistics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro

Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Christopher Cui, Xiangyu Peng, and Mark O. Riedl. 2023. Thespian: Multi-character text role-playing game agents. *CoRR*, abs/2308.01872.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2023. Investigating data contamination in modern benchmarks for large language models. *CoRR*, abs/2311.09783.

Xidong Feng, Yicheng Luo, Ziyan Wang, Hongrui Tang, Mengyue Yang, Kun Shao, David Mguni, Yali Du, and Jun Wang. 2023. Chessgpt: Bridging policy learning and language modeling. *CoRR*, abs/2306.09200.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Trans. Assoc. Comput. Linguistics*, 9:346–361.

Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. 2023. The false promise of imitating proprietary llms. *CoRR*, abs/2305.15717.

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.

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

Alec Helbling, Mansi Phute, Matthew Hull, and Duen Horng Chau. 2023. LLM self defense: By self examination, llms know they are being tricked. *CoRR*, abs/2308.07308.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *CoRR*, abs/2210.11610.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. In *Proceedings of the The 61st Annual Meeting of the Association for Computational Linguistics: Industry Track, ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 37–42. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Xiang Lorraine Li, Adhiguna Kuncoro, Jordan Hoffmann, Cyprien de Masson d'Autume, Phil Blunsom, and Aida Nematzadeh. 2022. A systematic investigation of commonsense knowledge in large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11838–11855. Association for Computational Linguistics.

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. 2023. Evaluating the logical reasoning ability of chatgpt and GPT-4. *CoRR*, abs/2304.03439.

David Noever, Matthew Ciolino, and Josh Kalin. 2020. The chess transformer: Mastering play using generative language models. *CoRR*, abs/2008.04057.

Odhran O'Donoghue, Aleksandar Shtedritski, John Ginger, Ralph Abboud, Ali Essa Ghareeb, Justin Booth, and Samuel G. Rodriques. 2023. Bioplanner: Automatic evaluation of llms on protocol planning in biology. *CoRR*, abs/2310.10632.

OpenAI. 2022. Introducing ChatGPT. (Accessed on Jun 18, 2023).

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? *CoRR*, abs/2302.06476.

Jonathan S. Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. 2020. A constructive prediction of the generalization error across scales. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. 2023. Are emergent abilities of large language models a mirage? *CoRR*, abs/2304.15004.

Hao Sha, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. 2023. Languagempc: Large language models as decision makers for autonomous driving. *CoRR*, abs/2310.03026.

Ananya Singha, José Cambronero, Sumit Gulwani, Vu Le, and Chris Parnin. 2023. Tabular representation, noisy operators, and impacts on table structure understanding tasks in llms. *CoRR*, abs/2310.10358.

Kaya Stechly, Matthew Marquez, and Subbarao Kambhampati. 2023. GPT-4 doesn't know it's wrong: An analysis of iterative prompting for reasoning problems. *CoRR*, abs/2310.12397.

Andreas Stöckl. 2021. Watching a language model learning chess. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), Held Online, 1-3September, 2021*, pages 1369–1379. INCOMA Ltd.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13003–13051. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Chen-Feng Tsai, Xiaochen Zhou, Sierra S. Liu, Jing Li, Mo Yu, and Hongyuan Mei. 2023. Can large language models play text games well? current state-of-the-art and open questions. *CoRR*, abs/2304.02868.

Tao Tu, Shekoofeh Azizi, Danny Driess, Mike Schaekermann, Mohamed Amin, Pi-Chuan Chang, Andrew Carroll, Chuck Lau, Ryutaro Tanno, Ira Ktena, Basil Mustafa, Aakanksha Chowdhery, Yun Liu, Simon Kornblith, David J. Fleet, Philip Andrew Mansfield, Sushant Prakash, Renee Wong, Sunny Virmani, Christopher Semturs, S. Sara Mahdavi, Bradley Green, Ewa Dominowska, Blaise Agüera y Arcas, Joelle K. Barral, Dale R. Webster, Gregory S. Corrado, Yossi Matias, Karan Singhal, Pete Florence, Alan Karthikesalingam, and Vivek Natarajan. 2023. Towards generalist biomedical AI. *CoRR*, abs/2307.14334.

Karthik Valmeekam, Alberto Olmo Hernandez, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (A benchmark for llms on planning and reasoning about change). *CoRR*, abs/2206.10498.

Karthik Valmeekam, Matthew Marquez, and Subbarao Kambhampati. 2023. Can large language models really improve by self-critiquing their own plans? *CoRR*, abs/2310.08118.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Louis Vervoort, Vitaliy Mizyakov, and Anastasia Ugleva. 2023. A criterion for artificial general intelligence: hypothetic-deductive reasoning, tested on chatgpt.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. Voyager: An open-ended embodied agent with large language models. *CoRR*, abs/2305.16291.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023b. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents.

Zihao Wang, Shaofei Cai, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023c. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *CoRR*, abs/2302.01560.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. 2023. Skywork: A more open bilingual foundation model. *CoRR*, abs/2310.19341.

Yixuan Weng, Minjun Zhu, Shizhu He, Kang Liu, and Jun Zhao. 2022. Large language models are reasoners with self-verification. *CoRR*, abs/2212.09561.

Yue Wu, Shrimai Prabhumoye, So Yeon Min, Yonatan Bisk, Ruslan Salakhutdinov, Amos Azaria, Tom M. Mitchell, and Yuanzhi Li. 2023. SPRING: GPT-4 out-performs RL algorithms by studying papers and reasoning. *CoRR*, abs/2305.15486.

Fangzhi Xu, Qika Lin, Jiawei Han, Tianzhe Zhao, Jun Liu, and Erik Cambria. 2023. Are large language models really good logical reasoners? a comprehensive evaluation and beyond.

Steve Yadlowsky, Lyric Doshi, and Nilesh Tripuraneni. 2023. Pretraining data mixtures enable narrow model selection capabilities in transformer models. *CoRR*, abs/2311.00871.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, and Songfang Huang. 2023. How well do large language models perform in arithmetic tasks? *CoRR*, abs/2304.02015.

Chaoning Zhang, Chenshuang Zhang, Chenghao Li, Yu Qiao, Sheng Zheng, Sumit Kumar Dam, Mengchun Zhang, Jung Uk Kim, Seong Tae Kim, Jinwoo Choi, et al. 2023. One small step for generative ai, one giant leap for agi: A complete survey on chatgpt in aigc era. *arXiv preprint arXiv:2304.06488*.

Andrew Zhu, Lara J. Martin, Andrew Head, and Chris Callison-Burch. 2023a. Calypso: Llms as dungeon masters' assistants.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023b. Large language models for information retrieval: A survey. *CoRR*, abs/2308.07107.

A Related Works

Recent studies have explored the reasoning capabilities of LLMs, with several investigations underway to understand (Qin et al., 2023; Bang et al., 2023; Liu et al., 2023; Xu et al., 2023; bench authors, 2023; Bao et al., 2023), enhance (Wei et al., 2022b; Imani et al., 2023), and leverage (Yao et al., 2023; Sha et al., 2023) these abilities for various applications. Some critics argue that the datasets used for evaluation are overly simplistic and potentially compromised by inclusion in the training data of LLMs (OpenAI, 2023; Wei et al., 2023), casting doubt on the authenticity of these purported capabilities (Valmeekam et al., 2022; Bao et al., 2023). In response, new tasks and evaluation frameworks have been suggested (Valmeekam et al., 2022; Xu et al., 2023; Bao et al., 2023; Vervoort et al., 2023; Wang et al., 2023a). However, these tend to focus narrowly on specific aspects of arithmetic or commonsense reasoning, often presented in unstructured natural language that intersects with LLMs' pre-existing world knowledge. This overlap presents a challenge in fully assessing LLMs' inductive reasoning abilities. In contrast, the game of Minesweeper encompasses symbolic, arithmetic, and commonsense reasoning in a structured format lying outside the scope of LLMs' training data, and thus demands genuine reasoning skills for successful resolution.

Alone another track, the use of LLMs in gameplay has gained attention, with efforts to harness them for diverse games such as card games (Gupta, 2023; Guo et al., 2023), interactive narrative games (Callison-Burch et al., 2022; Tsai et al., 2023; Cui et al., 2023; Zhu et al., 2023a), chess (Noever et al., 2020; Stöckl, 2021; Suzgun et al., 2023; Feng et al., 2023), and video games like Minecraft (Wang et al., 2023a,b). Nonetheless, we contend that such games are intrinsically distinct from Minesweeper in both structure and the level of challenge they pose to LLMs. Card and interactive narrative games largely draw on common sense and narrative understanding, with a constrained range of actions. This limitation can obscure whether LLM decisions stem from deep reasoning or instinctual response. Chess, while offering a more extensive range of possibilities, presents an abstract playfield that typically requires tailored pre-training for LLMs to grasp its rules. Minecraft poses a further challenge with its complex syntax and the need for meticulous, often biased, prompt engineering, making it an impressive display but not an accurate indicator of LLMs' functional application. Minesweeper, by comparison, offers a straightforward yet strategically rich gameplay experience with clear objectives and direct evaluation criteria, making it an exemplary model for assessing the practical reasoning capabilities of LLMs.

B Experiment Setup

B.1 GPT Versions

In this paper, we discuss three distinct versions of the GPT model: GPT-3.5-16k, GPT-3.5-instruct, and GPT-4. The term GPT-3.5-16k is used to denote the "gpt-3.5-turbo-16k" model as listed on the OpenAI API model webpage, specifically the checkpointed 0613.[1] The GPT-3.5-instruct model refers to the "gpt-3.5-turbo-instruct" variant, with the checkpoint version dated 2023-09-15. Lastly, GPT-4 represents the "gpt-4" model, utilizing also the 0613 checkpoint version. All mentioned checkpoints are hosted on Microsoft Azure,[2]. The model temperature is set to 0 in all cases.

B.2 Board Understanding

In our board understanding experiments, we initially generate 1,000 randomly arranged boards, each featuring a 9×9 grid with 10 hidden mines. Notably, the central cells, positioned at `(5,5)`, are always mine-free. From this pool, we select 100 boards, ensuring that the initial click at the center `L(5,5)` reveals at least 10 cells. Subsequently, we conduct gameplay using the graphical user interface depicted in Figure 1a. The actions of the annotators, who are average players in Minesweeper and unaware of the board's layout beyond the initial instruction to click the center, are recorded as shown in Figure 1d. Although not all trials are successful——with about 90% boards being solved and the remainder failing due to human errors or 50-50 guesses——the entire action history, including unsuccessful attempts, is preserved for analysis.

For each board, we randomly choose an action (excluding the first and last from the history) and associate board state as the focus. We then select 3 random coordinates per board, setting two challenges for LLMs: to determine the status of each selected cell (board navigation), and to count the number of specific symbols within {`?', `.', `1',

[1] https://platform.openai.com/docs/models

[2] *.openai.azure.com

`2'} in their neighbors, including diagonally adjacent ones (neighbor counting). This process yields 300 test cases for each task.

As outlined in § 3, we employ two distinct board representations: table and coordinate. Each required a slightly different approach in prompting. Specifically, the table representation uses LaTeX-style quotation marks, while the coordinate representation does not use quotations for board cell states and employs regular double quotes in the necessary positions in the game description. Examples illustrating these differences are provided below.

Listing 1: An example of board navigation prompting with table representation.

```
You will be presented with a 9 by 9 board for the Minesweeper game. The board is wrapped in a 10 by 10 table, where the first row and the first column with numbers in double quotation marks are the row and column indices. A coordinate (x,y) represents the cell at the x-th row and y-th column, where x and y, starting from 1, are the row and column indices, respectively. The state of each cell is represented by the following symbols:
- `.' represents a blank cell.
- `1' to `8' represents numbered cells with that number of mines in the adjacent cells.
- `F' represents a flagged cell.
- `?' represents an unopened cell.

--- EXAMPLES ---
--- PARTIAL BOARD ---
"0","1","2","3","4"
"1",`?',`?',`1',`.'
"2",`?',`?',`3',`1'
"3",`?',`F',`2',`F'
"4",`?',`2',`2',`1'

QUESTION: What is the cell at coordinate (1,3)?
ANSWER: `1'

--- PARTIAL BOARD ---
"0","1","2","3","4"
"1",`?',`?',`1',`.'
"2",`?',`?',`3',`1'
"3",`?',`F',`2',`F'
"4",`?',`2',`2',`1'

QUESTION: What is the cell at coordinate (4,1)?
ANSWER: `?'

--- END OF EXAMPLES ---

--- CURRENT BOARD ---
"0","1","2","3","4","5","6","7","8","9"
"1",`?',`?',`?',`?',`1',`.',`.',`.',`.'
"2",`?',`?',`?',`?',`1',`.',`1',`1',`1'
"3",`?',`1',`1',`1',`1',`.',`1',`F',`1'
"4",`?',`1',`.',`.',`.',`.',`1',`1',`1'
"5",`?',`2',`1',`1',`.',`.',`.',`.',`.'
"6",`?',`?',`?',`1',`1',`2',`3',`2',`1'
"7",`?',`?',`?',`?',`?',`?',`?',`?',`?'
"8",`?',`?',`?',`?',`?',`?',`?',`?',`?'
"9",`?',`?',`?',`?',`?',`?',`?',`?',`?'

QUESTION: What is the cell at coordinate (2,1)?
ANSWER:
```

Listing 2: An example of board navigation prompting with coordinate representation.

```
You will be presented with a 9 by 9 board for the Minesweeper game, which is devided into cells. The cells are presented as "coordinate: state" mappings. A coordinate (x,y) represents the element at the x-th row and y-th column, where x and y, starting from 1, are the row and column indices, respectively. The state of each cell is represented by the following symbols:
- "." represents a blank cell.
- "1" to "8" represents numbered cells with that number of mines in the adjacent cells.
- "F" represents a flagged cell.
- "?" represents an unopened cell.

--- EXAMPLES ---
--- PARTIAL BOARD ---
(1,1): ?
(1,2): ?
(1,3): ?
(1,4): ?
(2,1): ?
(2,2): ?
(2,3): ?
(2,4): ?
(3,1): ?
(3,2): ?
(3,3): ?
(3,4): 1
(4,1): ?
(4,2): ?
(4,3): ?
(4,4): 1

QUESTION: What is the cell at coordinate (3,4)?
ANSWER: "1""

--- END OF EXAMPLES ---

--- CURRENT BOARD ---
(1,1): ?
(1,2): ?
(1,3): ?
(1,4): ?
(1,5): 1
(1,6): .
(1,7): .
(1,8): .
(1,9): .
(2,1): ?
(2,2): ?
```

```
(2,3): ?
(2,4): ?
(2,5): 1
(2,6): .
(2,7): 1
(2,8): 1
(2,9): 1
(3,1): ?
(3,2): 1
(3,3): 1
(3,4): 1
(3,5): 1
(3,6): .
(3,7): 1
(3,8): F
(3,9): 1
(4,1): ?
(4,2): 1
(4,3): .
(4,4): .
(4,5): .
(4,6): .
(4,7): 1
(4,8): 1
(4,9): 1
(5,1): ?
(5,2): 2
(5,3): 1
(5,4): 1
(5,5): .
(5,6): .
(5,7): .
(5,8): .
(5,9): .
(6,1): ?
(6,2): ?
(6,3): ?
(6,4): 1
(6,5): 1
(6,6): 2
(6,7): 3
(6,8): 2
(6,9): 1
(7,1): ?
(7,2): ?
(7,3): ?
(7,4): ?
(7,5): ?
(7,6): ?
(7,7): ?
(7,8): ?
(7,9): ?
(8,1): ?
(8,2): ?
(8,3): ?
(8,4): ?
(8,5): ?
(8,6): ?
(8,7): ?
(8,8): ?
(8,9): ?
(9,1): ?
(9,2): ?
(9,3): ?
(9,4): ?
(9,5): ?
(9,6): ?
(9,7): ?
(9,8): ?
(9,9): ?

QUESTION: What is the cell at coordinate (2,1)?
ANSWER:
```

Listing 3: An example of neighbor counting prompting with table representation.

```
You will be presented with a 9 by 9 board for the Minesweeper game. The board is wrapped in a 10 by 10 table, where the first row and the first column with numbers in double quotation marks are the row and column indices. A coordinate (x,y) represents the cell at the x-th row and y-th column, where x and y, starting from 1, are the row and column indices, respectively. The state of each cell is represented by the following symbols:
- `.' represents a blank cell.
- `1' to `8' represents numbered cells with that number of mines in the adjacent cells.
- `F' represents a flagged cell.
- `?' represents an unopened cell.

--- EXAMPLES ---
--- PARTIAL BOARD ---
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`?',`?',`?',`?'
"3",`?',`?',`?',`1',`1'
"4",`?',`?',`?',`1',`.'
"5",`?',`?',`?',`1',`.'

QUESTION: How many cells `F' are neighbors (including diagonal) of the cell with coordinate (2,1)?
ANSWER:

To find out how many cells with the value `1' are neighbors of the cell with coordinate (2,1), we need to look at the 8 neighboring cells of (2,1). These coordinates are:
(1,1), (1,2), (3,1), (3,2), (2,2).

Now, we will check the values of these cells on the given Minesweeper board:

(1,1) = `?'
(1,2) = `?'
(3,1) = `?'
(3,2) = `?'
(2,2) = `?'

All of these neighboring cells are `?'. So, the number of cells with `1' that are neighbors of the cell (2,1) is:

ANSWER: 0.

--- END OF EXAMPLES ---

--- CURRENT BOARD ---
"0","1","2","3","4","5","6","7","8","9"
```

```
"1",`?',`?',`?',`?',`1',`.',`.',`.',`.'
"2",`?',`?',`?',`?',`1',`.',`1',`1',`1'
"3",`?',`1',`1',`1',`1',`.',`1',`F',`1'
"4",`?',`1',`.',`.',`.',`.',`1',`1',`1'
"5",`?',`2',`1',`1',`.',`.',`.',`.',`.'
"6",`?',`?',`?',`1',`1',`2',`3',`2',`1'
"7",`?',`?',`?',`?',`?',`?',`?',`?',`?'
"8",`?',`?',`?',`?',`?',`?',`?',`?',`?'
"9",`?',`?',`?',`?',`?',`?',`?',`?',`?'

QUESTION: How many cells `1' are
    neighbors (including diagonal) of
    the cell with coordinate (2,1)?
Let's think step by step.
ANSWER:
```

Listing 4: An example of neighbor counting prompting with coordinate representation.

```
You will be presented with a 9 by 9
    board for the Minesweeper game,
    which is devided into cells. The
    cells are presented as "coordinate:
    state" mappings. A coordinate (x,y)
    represents the element at the x-th
    row and y-th column, where x and y,
    starting from 1, are the row and
    column indices, respectively. The
    state of each cell is represented by
     the following symbols:
- "." represents a blank cell.
- "1" to "8" represents numbered cells
    with that number of mines in the
    adjacent cells.
- "F" represents a flagged cell.
- "?" represents an unopened cell.

--- EXAMPLES ---
--- PARTIAL BOARD ---
(1,1): F
(1,2): ?
(1,3): ?
(1,4): F
(2,1): ?
(2,2): ?
(2,3): ?
(2,4): ?
(3,1): ?
(3,2): F
(3,3): ?
(3,4): 1
(4,1): ?
(4,2): ?
(4,3): ?
(4,4): 1

QUESTION: How many cells "F" are
    neighboring (including diagonally)
    the cell with coordinates (2,1)?
Let's think step by step
ANSWER:

To find out how many cells with the
    value "1" are neighbors of the cell
    with coordinate (2,1), we need to
    look at the 8 neighboring cells of
    (2,1). These coordinates are:
(1,1), (1,2), (3,1), (3,2), (2,2).

Now, we will check the values of these
    cells on the given Minesweeper board
    :

(1,1) = F
(1,2) = ?
(3,1) = ?
(3,2) = F
(2,2) = ?

From of these neighboring cells (1,1)
    and (3,2) are "F". So, the number of
     cells with "1" that are neighbors
    of the cell (2,1) is:

ANSWER: 2.

--- END OF EXAMPLES ---

--- CURRENT BOARD ---
(1,1): ?
(1,2): ?
(1,3): ?
(1,4): ?
(1,5): 1
(1,6): .
(1,7): .
(1,8): .
(1,9): .
(2,1): ?
(2,2): ?
(2,3): ?
(2,4): ?
(2,5): 1
(2,6): .
(2,7): 1
(2,8): 1
(2,9): 1
(3,1): ?
(3,2): 1
(3,3): 1
(3,4): 1
(3,5): 1
(3,6): .
(3,7): 1
(3,8): F
(3,9): 1
(4,1): ?
(4,2): 1
(4,3): .
(4,4): .
(4,5): .
(4,6): .
(4,7): 1
(4,8): 1
(4,9): 1
(5,1): ?
(5,2): 2
(5,3): 1
(5,4): 1
(5,5): .
(5,6): .
(5,7): .
(5,8): .
(5,9): .
(6,1): ?
(6,2): ?
(6,3): ?
(6,4): 1
(6,5): 1
(6,6): 2
```

```
(6,7): 3
(6,8): 2
(6,9): 1
(7,1): ?
(7,2): ?
(7,3): ?
(7,4): ?
(7,5): ?
(7,6): ?
(7,7): ?
(7,8): ?
(7,9): ?
(8,1): ?
(8,2): ?
(8,3): ?
(8,4): ?
(8,5): ?
(8,6): ?
(8,7): ?
(8,8): ?
(8,9): ?
(9,1): ?
(9,2): ?
(9,3): ?
(9,4): ?
(9,5): ?
(9,6): ?
(9,7): ?
(9,8): ?
(9,9): ?

QUESTION: How many cells "1" are neighbors (including diagonal) of the cell with coordinate (2,1)?
Let's think step by step.
ANSWER:
```

In our analysis of the prompts, we employ the Chain-of-Thought (COT) technique, as described by Wei et al. (2022b), for the neighbor counting task but not for board navigation. The rationale behind this is that neighbor counting necessitates multi-step action: initially identifying the neighboring cells of a given target cell and subsequently tallying a specific symbol within these neighbors. In contrast, the board navigation task is straightforward, involving a direct query about a specified target cell. We have observed that incorporating COT into the neighbor counting process notably enhances its performance.

Furthermore, we have made the inclusion of in-context examples optional for the board navigation task, but mandatory for neighbor counting. The need for examples in neighbor counting arises from the models' propensity to generate unclear outputs in this task. By providing an example, we aim to guide the models towards producing more interpretable responses.

B.3 Minesweeper

The detailed setup of the Minesweeper experiment is presented in § 4. We hereby provide the actual initial prompts with table-formatted board representation used in our experiments. The coordinate representation closely mirrors the variations found in Listing 3 and Listing 4 and will not be omitted.

Listing 5: Initial prompt for Minesweeper gameplay.

```
>> SYSTEM:
You are a helpful assistant who is good at playing Minesweeper.

>> USER:
--- MINESWEEPER INTRODUCTION ---
In Minesweeper, 4 hidden mines are scattered throughout a 5 by 5 board, which is divided into cells. The rows are seperated by newlines, and columns by commas. The board is structured as a 6 by 6 table, with the first row and column labeled using numbers in double quotation marks to indicate row and column indices. Cells have multiple possible states:
- Unopened cells (represented by `?', which cover the board at the start of the game, can also be made by removing flags)
- Numbered cells (represented by `1' to `8', which indicate the number of mines in the eight neighboring cells, including those diagonally adjacent)
- Blank cells (represented by `.', which have no neighboring mines)
- Flagged cells (represented by `F', which are marked by the player to indicate a potential mine location)

A player selects a cell to open it. If a player opens a cell containing a mine, the game ends in a loss. Otherwise, the opened cell displays either a number, indicating the number of mines diagonally and/or adjacent to it, or a blank tile (sometimes shown as a 0), and all adjacent cells will automatically be opened. To win a game of Minesweeper, all non-mine cells must be opened without opening a mine.

--- ACTION OPTIONS ---
There are three permissible actions in Minesweeper:

- Left-click an unopened cell (`?') to reveal it.
- Right-click an unopened cell (`?') to place a flag or a flagged cell (`F') to remove the flag.
- Middle-click on a numbered cell (`1' to `8') to unveil its neighboring cells, but only if all adjacent mines have been correctly flagged. If any flags are misplaced, you'll lose the game.

--- ACTION FORMAT ---
```

Type of Invalid Actions	Example	Feedback
	Coordinate	
Coordinate Out of Bound	"L(0,10)" on a 9×9 board	Invalid Coordinates! Please make sure your coordinate are within [1, 9] for rows and [1, 9] for columns.
	Game Initialization	
Starting by Right-Clicking	"R(5,5)" on a new board	Please begin by left-clicking on the center cell.
Starting by Middle-Clicking	"M(5,5)" on a new board	Please begin by left-clicking on the center cell.
	Left-Clicking	
Left-Clicking on Blank Cell	"L(5,5)" where (5,5) is opened as `.'	Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Left-Clicking on Flagged Cell	"L(5,5)" where (5,5) is flagged as `F'	Invalid action: Cannot left-click a flagged cell. Left-click is only for unopened cells (`?').
Left-Clicking on Numbered Cell	"L(5,5)" where (5,5) is opened as `2'	Invalid action: Cannot left-click a numbered cell. Left-click is only for unopened cells (`?').
	Middle-Clicking	
Middle-Clicking on Blank Cell	"M(5,5)" where (5,5) is opened as `.'	Invalid action: Cannot middle-click a blank cell. Middle-click is only for numbered cells (`1' to `8').
Middle-Clicking on Flagged Cell	"M(5,5)" where (5,5) is flagged as `F'	Invalid action: Cannot middle-click a flagged cell. Middle-click is only for numbered cells (`1' to `8').
Middle-Clicking on Unopened Cell	"M(5,5)" where (5,5) is unopened (`?')	Invalid action: Cannot middle-click an unopened cell. Middle-click is only for numbered cells (`1' to `8').
Middle-Clicking When Numbered Cell Has no Flagged Neighbor		Error: No flagged cells detected nearby. Flag adjacent mines before middle-clicking.
Middle-Clicking When # Flagged Neighbor Mismatchs Cell Number		Error: Flag count mismatch. Ensure all adjacent mines are flagged before middle-clicking.
	Right-Clicking	
Right-Clicking on Blank Cell	"R(5,5)" where (5,5) is opened as `.'	Invalid action: Cannot right-click a blank cell. Right-click is only for unopened cells (`?') or flagged cells (`?').
Right-Clicking on Numbered Cell	"R(5,5)" where (5,5) is opened as `1'	Invalid action: Cannot right-click a numbered cell. Right-click is only for unopened cells (`?') or flagged cells (`?').

Table 3: Types and feedbacks for invalid actions.

```
Each of your actions should be formatted as "A(row,col)", where:
- "A" represents the action type: "L" denotes a left-click, "R" indicates a right-click, and "M" signifies a middle-click.
- "row" specifies the row number of the targeted cell.
- "col" details the column number of the targeted cell.
For instance, an action like "L(1,2)" translates to a left-click on the cell located at the first row and second column.

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`?',`?',`?',`?'
"3",`?',`?',`?',`?',`?'
"4",`?',`?',`?',`?',`?'
"5",`?',`?',`?',`?',`?'
```

--- EXAMPLES ---
Example 1:
--- PARTIAL BOARD ---
"0","1","2","3","4"
"1",`.',`1',`?',`?'
"2",`.',`1',`?',`?'
"3",`.',`1',`2',`4'
"4",`.',`.',`.',`2'

REASONING: The cell located at (3,2) displays the number `1', indicating there's a single mine in its neighboring cells. Among these neighbors, 7 are opened as empty or numbered, and only one remains unopened at (2,3). Given that there's just one adjacent mine, it's
```

```
    logical to deduce that the unopened
    cell at (2,3) contains the mine.
    Thus, our next move is to flag the
    cell at (2,3) with a right-click.

ACTION: R(2,3)

Example 2:
--- PARTIAL BOARD ---
"0","4","5","6","7"
"1",`.',`1',`?',`?'
"2",`.',`1',`F',`?'
"3",`.',`1',`2',`4'
"4",`.',`.',`.',`2'

REASONING: The cell at (1,5) displays
    the number `1', indicating a single
    adjacent mine. Upon examining its
    neighboring cells, only the cell at
    (2,6) is flagged with `F'. This
    suggests that all mines neighboring
    (1,5) have been identified.
    Consequently, we can safely unveil
    the other surrounding cells by
    middle-clicking on (1,5).

ACTION: M(1,5)

Example 3:
--- PARTIAL BOARD ---
"0","1","2","3","4"
"1",`1',`1',`2',`?'
"2",`?',`?',`?',`?'
"3",`?',`?',`?',`?'

REASONING: The cell at (1,1) indicates
    there's a single mine amongst its
    neighbors. Examining the cells
    adjacent to it, both (2,1) and (2,2)
     remain unopened, implying one of
    them contains a mine. Similarly, the
     cell at (1,2) displays a `1',
    suggesting that out of (2,1), (2,2),
     and (2,3), one holds a mine. Since
    one of (2,1) or (2,2) already
    contains a mine, it becomes evident
    that (2,3) is mine-free. We can then
     safely uncover (2,3) with a left-
    click.

ACTION: L(2,3)

--- END OF EXAMPLES ---

--- RESPONSE GUIDE ---
Let's think step by step.

Begin by detailing your rationale in the
     "REASONING" section, followed by
    specifying your move using the
    previously mentioned format in the "
    ACTION" section. Ensure your
    reasoning doesn't exceed 200 words.
    To commence the game, left-click the
     center cell at (3, 3).

REASONING:
ACTION:
```

In subsequent iterations, the prompts are modified based on the model's responses. When the model generates a valid action, the updated game board is presented, followed by a request for the model to initiate the next move. In contrast, if the action is invalid, an error message is displayed, prompting the model to select an alternative action. Table 3 enumerates the various types of invalid actions and their corresponding feedback tailored for LLMs. Note that these actions do not directly violate the game rules. Rather, they are deemed invalid within the scope of our experiments because they are logically ungrounded and do not contribute to the progression of the game.

C Interaction Examples

In this section, we provide more case studies on the reasoning chains generated by LLMs as support of our discussion in § 4. **For better readability, we convert all coordinate representations to table representations.** Please refer to our GitHub repository for a complete set of experiment results.

Listing 6: An example of LLM-system interaction with table representation in NC mode from GPT-3.5-16k.

```
// Initial Instructions are Omitted

>> ASSISTANT:
REASONING: Since the board is completely
     covered with unopened cells, there
    is no specific information available
     to make an informed decision about
    which cell to open first. Therefore,
     I will start by randomly selecting
    a cell to open.

ACTION: L(3,3)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:


>> ASSISTANT:
REASONING: Looking at the newly revealed
     cells, we can see that the cell at
    (2,2) is surrounded by two mines.
    The cell at (2,3) is surrounded by
```

one mine. The cell at (3,2) is also surrounded by one mine. Based on this information, we can deduce that the cell at (3,3) is safe to open.

ACTION: L(3,3)

>> USER:
Your previous action "L(3,3)" is invalid. Error Message:
Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the mistake. Since the cell at (3,3) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,5) is surrounded by two mines. Therefore, it is safer to open the cell at (5,5) next.

ACTION: L(5,5)

>> USER:
Your previous action "L(5,5)" is invalid. Error Message:
Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the mistake. Since the cell at (5,5) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,4) is surrounded by two mines. Therefore, it is safer to open the cell at (5,4) next.

ACTION: L(5,4)

>> USER:
Your previous action "L(5,4)" is invalid. Error Message:
Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the mistake once again. Since the cell at (5,4) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,3) is surrounded by two mines. Therefore, it is safer to open the cell at (5,3) next.

ACTION: L(5,3)

>> USER:
Your previous action "L(5,3)" is invalid. Error Message:

Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the repeated mistakes. Since the cell at (5,3) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,2) is surrounded by two mines. Therefore, it is safer to open the cell at (5,2) next.

ACTION: L(5,2)

>> USER:
Your previous action "L(5,2)" is invalid. Error Message:
Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the repeated mistakes. Since the cell at (5,2) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,1) is surrounded by two mines. Therefore, it is safer to open the cell at (5,1) next.

ACTION: L(5,1)

>> USER:
Your previous action "L(5,1)" is invalid. Error Message:
Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the repeated mistakes. Since the cell at (5,1) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,0) is surrounded by two mines. Therefore, it is safer to open the cell at (5,0) next.

ACTION: L(5,0)

>> USER:
Your previous action "L(5,0)" is invalid. Error Message:
Invalid Coordinates! Please make sure your coordinate are within [1, 5] for rows and [1, 5] for columns.
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
```

```
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the repeated mistakes. Since the cell at (5,0) is not a valid coordinate, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,1) is surrounded by two mines. Therefore, it is safer to open the cell at (5,1) next.

ACTION: L(5,1)

>> USER:
Your previous action "L(5,1)" is invalid. Error Message:
Invalid action: Cannot left-click a blank cell. Left-click is only for unopened cells (`?').
Please follow the instructions and try again.

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`?',`?',`?',`?'
"2",`?',`2',`1',`2',`?'
"3",`1',`1',`.',`1',`?'
"4",`.',`.',`.',`1',`1'
"5",`.',`.',`.',`.',`.'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:

ACTION:

>> ASSISTANT:
REASONING: Apologies for the repeated mistakes. Since the cell at (5,1) is already opened and blank, we need to look for another cell to open. Based on the current information, the cell at (4,4) is surrounded by one mine and the cell at (4,5) is surrounded by one mine. However, the cell at (5,2) is surrounded by two mines. Therefore, it is safer to open the cell at (5,2) next.

ACTION: L(5,2)

Listing 7: An example of LLM-system interaction with coordinate representation in CH mode from GPT-3.5-instruct.

// Initial Instruction Duplicates Listing 5 and thus Omitted

--- YOUR ACTION HISTORY ---
[Action 1]: ACTION: R(2,2) -> FEEDBACK: Please begin by left-clicking on a cell.
[Action 2]: ACTION: L(1,1) -> FEEDBACK: Action successful!
[Action 3]: ACTION: R(5,4) -> FEEDBACK: Action successful!
[Action 4]: ACTION: R(4,5) -> FEEDBACK: Action successful!
[Action 5]: ACTION: R(5,5) -> FEEDBACK: Action successful!
[Action 6]: ACTION: R(5,2) -> FEEDBACK: Action successful!
[Action 7]: ACTION: R(5,1) -> FEEDBACK: Action successful!
[Action 8]: ACTION: R(5,5) -> FEEDBACK: Action successful!
[Action 9]: ACTION: R(5,5) -> FEEDBACK: Action successful!

--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`.',`.',`.',`.',`.'
"2",`.',`.',`.',`.',`.'
"3",`.',`.',`.',`1',`1'
"4",`1',`2',`2',`3',`F'
"5",`F',`F',`?',`F',`F'
```

--- RESPONSE GUIDE ---
Let's think step by step.

Begin by detailing your rationale in the "REASONING" section, followed by specifying your move using the previously mentioned format in the "ACTION" section. Ensure your reasoning doesn't exceed 200 words.

REASONING:
ACTION:

>> ASSISTANT:

REASONING: The cell at (4,5) is a numbered cell with a value of "3". This indicates that there are three mines in its neighboring cells. Out of the three neighboring cells, two are already flagged with "F". This means that the remaining unopened cell at (5,3) must contain the third mine. Therefore, we can safely flag the cell at (5,3) with a right-click.

```
ACTION: R(5,3)
```

Listing 8: An example of LLM-system interaction with coordinate representation in NC mode from GPT-4.

```
// Initial Instructions and Responses are Omitted

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`?'
"2",`?',`2',`.',`1',`?'
"3",`?',`2',`.',`1',`?'
"4",`?',`1',`1',`1',`?'
"5",`?',`?',`?',`?',`?'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:
ACTION:

>> ASSISTANT:
REASONING:
The cell at (2,2) indicates there are 2 mines in its neighboring cells. Since (1,1), (2,1), and (3,1) are the only unopened cells around it, we can deduce that two of these cells contain mines. Similarly, the cell at (3,2) also indicates there are 2 mines in its neighboring cells. Considering that (3,1), (4,1), and (5,1) are the only unopened cells around it, and we know that two of (1,1), (2,1), and (3,1) contain mines, we can conclude that (4,1) and (5,1) are safe to open.

ACTION: L(4,1)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`?'
"2",`?',`2',`.',`1',`?'
"3",`?',`2',`.',`1',`?'
"4",`1',`1',`1',`1',`?'
"5",`?',`?',`?',`?',`?'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:
ACTION:

>> ASSISTANT:
REASONING:
The cell at (4,1) indicates there's 1 mine in its neighboring cells. Since (3,1), (5,1), and (5,2) are the only unopened cells around it, and we know that one of (3,1) or (5,1) contains a mine (from the previous reasoning), we can conclude that (5,2) is safe to open.

ACTION: L(5,2)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`?'
"2",`?',`2',`.',`1',`?'
"3",`?',`2',`.',`1',`?'
"4",`1',`1',`1',`1',`?'
"5",`.',`.',`1',`?',`?'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:
ACTION:

>> ASSISTANT:
REASONING:
The cell at (5,3) indicates there's 1 mine in its neighboring cells. Since (4,4), (5,4), and (5,5) are the only unopened cells around it, we can deduce that one of these cells contains a mine. However, we don't have enough information to determine which one. Therefore, the best strategy is to open a cell that is not adjacent to any numbered cells. The cell at (1,5) fits this criterion.

ACTION: L(1,5)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`1'
"2",`?',`2',`.',`1',`?'
"3",`?',`2',`.',`1',`?'
"4",`1',`1',`1',`1',`?'
"5",`.',`.',`1',`?',`?'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:
ACTION:

>> ASSISTANT:
REASONING:
The cell at (1,5) indicates there's 1 mine in its neighboring cells. Since (2,5) is the only unopened cell around it, we can deduce that (2,5) contains a mine. Therefore, our next move is to flag the cell at (2,5) with a right-click.

ACTION: R(2,5)
```

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`1'
"2",`?',`2',`.',`1',`F'
"3",`?',`2',`.',`1',`?'
"4",`1',`1',`1',`1',`?'
"5",`.',`.',`1',`?',`?'
```

>> ASSISTANT:
REASONING:
The cell at (2,4) indicates there's 1 mine in its neighboring cells. Since (2,5) is flagged and (3,5) is the only other unopened cell around it, we can deduce that (3,5) does not contain a mine. Therefore, our next move is to open the cell at (3,5) with a left-click.

ACTION: L(3,5)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`1'
"2",`?',`2',`.',`1',`F'
"3",`?',`2',`.',`1',`1'
"4",`1',`1',`1',`1',`?'
"5",`.',`.',`1',`?',`?'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:
ACTION:

>> ASSISTANT:
REASONING:
The cell at (3,5) indicates there's 1 mine in its neighboring cells. Since (4,5) is the only unopened cell around it, we can deduce that (4,5) contains a mine. Therefore, our next move is to flag the cell at (4,5) with a right-click.

ACTION: R(4,5)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`?',`1',`.',`1',`1'
"2",`?',`2',`.',`1',`F'
"3",`?',`2',`.',`1',`1'
"4",`1',`1',`1',`1',`F'
"5",`.',`.',`1',`?',`?'
```

>> ASSISTANT:
REASONING:
The cell at (5,3) indicates there's 1 mine in its neighboring cells. Since (4,5) is flagged and (5,4) and (5,5) are the only other unopened cells around it, we can deduce that one of them contains a mine. However, we don't have enough information to determine which one. Therefore, the best strategy is to open a cell that is not adjacent to any numbered cells. The cell at (1,1) fits this criterion.

ACTION: L(1,1)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`1',`1',`.',`1',`1'
"2",`?',`2',`.',`1',`F'
"3",`?',`2',`.',`1',`1'
"4",`1',`1',`1',`1',`F'
"5",`.',`.',`1',`?',`?'
```

>> ASSISTANT:
REASONING:
The cell at (1,1) indicates there's 1 mine in its neighboring cells. Since (2,1) is the only unopened cell around it, we can deduce that (2,1) contains a mine. Therefore, our next move is to flag the cell at (2,1) with a right-click.

ACTION: R(2,1)

>> USER:
--- CURRENT BOARD ---
```
"0","1","2","3","4","5"
"1",`1',`1',`.',`1',`1'
"2",`F',`2',`.',`1',`F'
"3",`?',`2',`.',`1',`1'
"4",`1',`1',`1',`1',`F'
"5",`.',`.',`1',`?',`?'
```

please ensure:
- You do not duplicate actions.
- You submit only one action at a time.

REASONING:
ACTION:

>> ASSISTANT:
REASONING:
The cell at (3,2) indicates there are 2 mines in its neighboring cells. Since (2,1) and (2,5) are flagged and (3,1) is the only other unopened cell around it, we can deduce that (3,1) does not contain a mine. Therefore, our next move is to open the cell at (3,1) with a left-click.

ACTION: L(3,1)