Improving the Decision-Making Accuracy of LLM-Based Agents in Game Applications

Guge Sun

Dulwich International High School Suzhou, Suzhou, China
simon.sun26@stu.dulwich.org

Abstract. Large Language Models (LLMs), such as GPT, have recently been deployed as agents in interactive environments, including games. While these models exhibit strong natural language understanding capabilities, they often produce incorrect or inconsistent decisions in strategic games that require strict rule adherence and multi-step planning. This project investigates the decision-making behavior of an LLM-based agent in the game of Tic-Tac-Toe. A simulator was developed to enable gameplay between the LLM and a human player, and multiple prompt designs were evaluated. The experiments compare prompts with no contextual guidance, basic rule descriptions, example-based contexts, and explicit strategic instructions. The results demonstrate that prompt design plays a critical role in improving the reliability of LLM-based game agents and offer insights into how decision-making accuracy can be significantly enhanced without modifying the underlying model.

Keywords: Large Language Models (LLMs), Prompt Engineering, Game Agent Decision-Making

1. Introduction

Large Language Models (LLMs), such as GPT-4 and Claude, have demonstrated impressive capabilities in text generation, reasoning, and question answering [1]. More recently, researchers have explored the use of LLMs as autonomous agents in interactive environments, including games and simulations [2-4]. In these contexts, an agent must perceive the current state of the environment, reason about potential future outcomes, and select an appropriate action.

However, unlike traditional game-playing algorithms, LLMs are not explicitly trained to adhere to formal rules or perform systematic, step-by-step planning [5]. Consequently, LLM-based agents may make illegal moves, fail to block obvious threats, or overlook winning opportunities, even in simple games. Understanding the causes of these errors and identifying effective strategies to mitigate them are crucial steps toward developing more reliable and robust AI agents.

In this project, Tic-Tac-Toe [6] is employed as a controlled test environment to examine the decision-making behavior of LLM-based agents. Although the game is simple, it requires rule comprehension, pattern recognition, and strategic reasoning. The central research question of this study is: How different prompt design affects the decision-making accuracy of an LLM-based agent in a game environment?

To address this question, a range of prompt styles—spanning from minimal instructions to detailed strategic guidance with illustrative examples—were systematically evaluated and compared. The experimental results reveal clear and progressive performance differences across prompt designs. Prompts with no contextual guidance frequently produced invalid outputs and illegal moves, indicating insufficient specification of formatting and rule constraints. Simple contextual prompts enabled stable input–output interactions but resulted in poor gameplay performance, with rigid and repetitive move patterns and frequent failures to block opponent threats.

Example-based prompts led to noticeable improvements: the model demonstrated more reasonable move selection, increased blocking behavior, and a higher frequency of tied games, suggesting a preliminary understanding of the game objective. However, residual routine patterns remained. The strongest performance emerged under detailed strategy prompts, where the agent consistently blocked threats, prioritized advantageous positions, and avoided obvious losses. In particular, the most structured strategy configuration resulted in a significant reduction in steps required to secure a win and achieved complete winning performance during evaluation.

Overall, performance improved as prompts became increasingly structured, concrete, and strategy-focused, demonstrating that prompt engineering plays a critical role in enhancing rule adherence and strategic reasoning in LLM-based agents (see Table 1).

2. Related work

Early game-playing systems, such as minimax-based algorithms, rely on explicit rules and exhaustive search to guarantee optimal play in games like Tic-Tac-Toe. These systems are deterministic and never produce illegal moves.

More recent research has investigated the use of Large Language Models (LLMs) as agents in games and simulations. While LLMs can reason about game states using natural language, their performance is often unstable and inconsistent. Prior studies indicate that prompt engineering—such as the use of few-shot examples and structured instructions—can significantly enhance LLM reasoning in domains including mathematical problem-solving and logical inference.

In game-related contexts, researchers have also explored approaches such as Retrieval-Augmented Generation (RAG) and model fine-tuning to provide external knowledge or task-specific behavior. However, prompt-based methods are particularly attractive because they improve performance without requiring retraining of the underlying model.

Building on this line of work, the present project systematically compares different prompt designs within a simple yet well-defined game environment to evaluate the impact of prompt design on the decision-making accuracy of LLMs that are not explicitly optimized for logical reasoning.

3. Methodology

3.1. Game environment

A Tic-Tac-Toe simulator was implemented in Python. The game board is represented as a 3×3 grid, where X denotes the LLM-based agent, O denotes the human player, and empty cells are represented by blank spaces.

The simulator performs the following steps during gameplay:

1) Displays the current state of the game board.
2) Sends the board state to the LLM agent.
3) Receives the LLM's response containing the updated board configuration.

4) Checks for a win, loss, or tie after the LLM's move.
5) Prompts the human player to enter a move.
6) Checks again for a win, loss, or tie after the human player's move.

3.2. LLM interaction

The LLM is prompted to play one move at a time and is required to output the complete updated board state after placing an X. A parser then validates the model's output and updates the game state accordingly.

3.3. Prompt designs

The code for prompting LLM is shown below, where the CONTEXT will be changed during experiments.

```
Def send_to_gpt(current_board_text):
messages = [
{ 'role': 'system', 'content': CONTEXT },
{ 'role': 'user', 'content': f"Current status:\n{current_board_text}\nPlay one step and output the updated status\n{Strategy}" }
]
response = client.chat.completions.create(
model='gpt-4.1-mini',
messages=messages
)
return response.choices[0].message.content
```

The code above demonstrates how a prompt is generated and sent to gpt-4.1-mini. The prompt "Current status:\n{current_board_text}\nPlay one step and output the updated status\n{Strategy}" recalls the current board state and appends the specified strategy on a new line, guiding the model's next move. Several types of prompts were tested:

1) No Context: The model receives only the current board and a request to play a single move. No rules, examples, or strategic guidance are provided.

2) Simple Context: Basic Tic-Tac-Toe rules are described in text. No examples or detailed strategies are included. This setup is used to preliminarily evaluate the LLM's ability to play the game and identify areas for further improvement. Example context is shown below:

```
Tic-Tac-Toe
Whoever lines up three in a row wins.
```

3) Context with Examples: Multiple examples of winning boards for both X and O are provided in the context, such that the model is explicitly instructed to focus on identifying three-in-a-row patterns. For example, we can add the following example in the context:

```
In situations like the following, X will win:
|X| |O|      |X|X|X|      |X| | |      | | |X|
| |X| |      | |O| |      |X|O|O|      | |X|O|
|O| |X|      | | |O|      |X| | |      |X| |O|
In this example, you should focus only on the positions of X. In each case, X wins because three Xs align in a straight line—
horizontally, vertically, or diagonally.
Similarly, in situations like the following, O will win:
|O| |X|      |O|O|O|      |O|X| |      | |X|O|
|X|O|X|      | |X|X|      |O|X|X|      |X|O|X|
|X| |O|      |X|X| |      |O| |X|      |X|X| |
Here, focus only on the positions of O. In each case, O wins because three Os align in a straight line—horizontally, vertically, or
diagonally
```

4) Strategy-Based Context: Explicit strategies are provided, including: always block the opponent if they can win on their next turn; prioritize occupying the center square; evaluate all rows, columns, and diagonals before making a move. More advanced versions also include explanations of why each move is necessary, along with examples of correct and incorrect decisions. Below are three strategies designed for this type of context.

Strategy v1:

```
Always block O if they can win on their next turn.
Prioritize the center square, then blocking, followed by corners.
Before moving, evaluate all rows, columns, and diagonals.
Focus on stopping threats before creating your own opportunities.
Monitor win and tie conditions after each move.
```

This initial version of the strategy was developed based on AI analysis of gameplay data and behavioral observations from preliminary experiments. The emphasis on prioritizing the center square is supported by simple statistical analysis of winning moves.

Strategy v2:

```
Always block O if they can win on their next turn.
Example:
Before:      After:
|X| | |      |X| | |
| |X| |  →   | |X| |
|O| |O|      |O|X|O|
Prioritize the center, then blocking, then corners.
Examples:
Before:      After:
| | | |      | | | |
| | | |  →   | |X| |
| | | |      | | | |
Before:      After:
|O| | |      |O| | |
| |X| |  →   |X|X| |
|O| | |      |O| | |
Before:      After:
|O| | |      |O| |X|
| |X| |  →   | |X| |
| | | |      | | | |
Before moving, check all rows, columns, and diagonals.
Stop threats before creating your own.
Example:
Before:      After:
|X| | |      |X| | |
| |X| |  →   | |X| |
|O| |O|      |O|X|O|
Monitor win and tie conditions after every move.
For each strategy, examples are included to illustrate how the strategy should be applied in practice.
```

Strategy V3:

```
Always block O if they can win on their next turn.
Example:
Before:      After:
|X| | |      |X| | |
| |X| |  →   | |X| |
|O| |O|      |O|X|O|
This move is necessary because if X is not placed here, O could complete three in a row on the next turn.
Also, prioritize the center square.
Before moving, evaluate all rows, columns, and diagonals.
Stop threats before creating your own.
```

```
Example:
Before:      After:
|X| | |      |X| | |
| |X| | →    | |X| |
|O| |O|      |O|X|O|
When O poses no immediate threat, create your own threats to force responses.
Examples:
Before:      After:
|O| | |      |O| |X|
| |X| | →    | |X| |(creates threat at 2, 0)
| | | |      | | | |
Or:
Before:      After:
|O| | |      |O| | |
| |X| | →    |X|X| |(creates threat at 1, 2)
| | | |      | | | |
Do not update the board as:
Before:      After:
|O| | |      |O| | |
| |X| | →    | |X| |
| | | |      | | |X|
As this misses the opportunity to force the opponent's response.
Strategies apply in all directions. For instance, approaches used for:
|O| | |
| |X| |
| | | |
Can also be applied to rotated or mirrored boards such as:
| | |O|     | | | |     | | | |
| |X| |     | |X| |     | |X| |
| | | |     | | |O|     |O| | |
```

Note: Detailed explanations for each example were added to encourage strategic behavior beyond what was observed in preliminary experiments. Each prompt design was tested multiple times under similar game conditions to ensure consistency.

4. Experiments and results

4.1. Experimental setup

For each prompt type, multiple games were played between the LLM-based agent (X) and a human player (O). The following outcomes were recorded: LLM wins, Human wins, Ties, and Illegal or suboptimal moves.

Due to the LLM's weak performance in early experiments, the LLM agent (X) was assigned the first move—an inherent advantage in Tic-Tac-Toe—in order to make performance improvements more observable across different prompt designs.

4.2. Results summary

The experimental results revealed clear performance differences across prompt designs:

• No Context: This prompt frequently produced invalid outputs and illegal moves. The results indicate that the prompt lacked sufficient specificity regarding output rules and formatting requirements.

• Simple Context: This prompt enabled stable input–output interactions across complete games but resulted in poor gameplay performance. The model failed to block the opponent's threats and followed a fixed, repetitive move pattern in each game. Even when the intended move was deliberately blocked by the human player, the model's subsequent moves remained highly predictable. This rigid behavior is reflected in the grayscale visualizations, which show a strong concentration on a single board position.

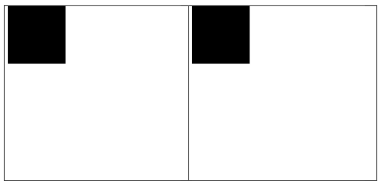

Figure 1. Grayscale visualization of move distribution at step 0 under regular prompting

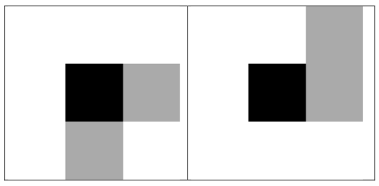

Figure 2. Grayscale visualization of move distribution at step 1 under regular prompting

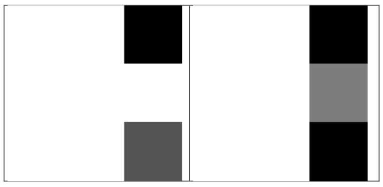

Figure 3. Grayscale visualization of move distribution at step 1 under regular prompting

• Example-Based Contexts: LLM made more reasonable decisions, but still loses in most cases. A significant improvement in the number of blocking actions and ties is observed, indicating a preliminary improvement in understanding of the game's goal. While there's still a routine of moves followed across the games.

• Detailed Strategy Prompts: They produced the best results, with the LLM consistently blocking threats, prioritizing strong positions, and avoiding obvious losses.The decrease in number of ties is expected to be the result of better understanding of the game rule and more aggressive decision making due to given strategy. With strategy 3, LLM-agent showed significant growth in identifying moves that lead to win. There's a massive decrease in steps taken to win while achieving full winning games in the feeding section.

Overall, performance improved as prompts became increasingly structured, concrete, and strategy focused, as shown in Table 1.

Table 1. Evaluation results of different prompts

LLM Tic-Tac-Toe Performance Across Prompt Designs					
Prompt Type	Games	Wins (LLM)	Mistakes	Blocks	Ties
Simple context (no examples)	10	0/2	1	0	0/1
Context + example 1	10	0/3	0	1	1/1
Context + example 2	10	0/2	1	1	4/1
Context + example 3	10	2/3	2	4	1/0
Strategy V1	10	0/4	2	6	0/0
Strategy V2	10	0/3	2	1	0/1

Table 1. (continued)

Strategy V3	10	0/5*	0	1	1/0

* Strategy V3 shows a significant reduction in steps required to achieve a win.

5. Discussion

The results suggest that LLMs do not naturally internalize game rules or strategic principles unless they are clearly and explicitly specified. Although the model demonstrates the ability to reason about the board state, it frequently fails to apply correct strategies without structured guidance.

Providing examples enables the model to recognize relevant gameplay patterns, while explicit strategic instructions help it apply these patterns effectively in novel situations. Additionally, explanations describing why a particular move is important appear to further reduce errors, indicating that causal reasoning can enhance decision-making performance.

Beginning with Strategy v1, a noticeable shift in the LLM's behavior was observed. As illustrated in the grayscale visualizations, the model successfully adopted the strategy of prioritizing the center square as the opening move.

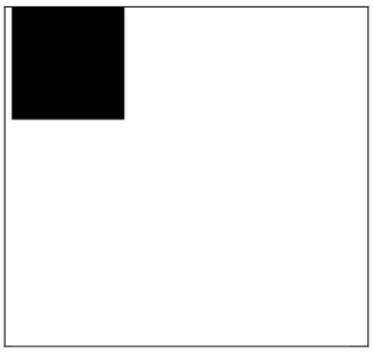

Figure 4. Grayscale images illustrating the distribution of initial moves (step 0) for regular gameplay with example-based context prompts (examples 1, 2, and 3)

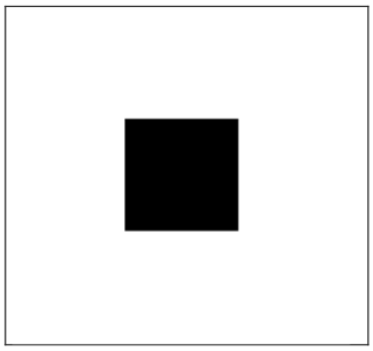

Figure 5. Grayscale images illustrating the distribution of initial moves (step 0) for regular gameplay with example-based context prompts (examples 1, 2, and 3)

However, the consistently poor performance in standard games where the human player selects optimal moves suggests that the LLM-based agent has a limited ability to perceive the board state holistically and to anticipate the opponent's future actions. The observed pattern of initially increasing and subsequently decreasing blocking behavior further indicates potential issues such as inadequate strategy prioritization or an incomplete evaluation of the game state, both of which require more explicit and structured instructions.

These findings highlight a key limitation of LLM-based agents: their reasoning behavior is highly sensitive to prompt design. Unlike traditional game-playing algorithms, LLMs do not guarantee optimal play unless carefully guided. Moreover, experiments with different LLM models show that even minor variations in keyword choice can lead to substantially different behaviors. Certain keywords may implicitly reorder strategic priorities, revealing additional complexity and fragility when transferring or reusing prompt contexts across models.

6. Conclusion

This project examined how prompt design influences the decision-making accuracy of an LLM-based agent in the game of Tic-Tac-Toe. Using a custom-built simulator and a range of prompt styles, the experiments demonstrated that explicit strategic instructions substantially improve the model's performance, whereas structured examples alone did not yield a clear or consistent performance gain.

In regular gameplay, no significant improvement in the agent's performance was observed across the experiments, indicating a highly limited depth of move planning by the model. These results suggest that more detailed and carefully structured strategies are necessary to compensate for this limitation, and they highlight the need to balance prompt complexity with algorithmic reasoning requirements when designing effective LLM-based game agents.

As suggested by the experiments, for simple games such as Tic-Tac-Toe, where an optimal move can be clearly defined, low-level LLM-based agents may not be as efficient as algorithms that follow a predetermined optimal strategy. However, these experiments provide a glimpse of the potential for improving LLM behavior in game AI applications. Future work could extend this approach to more complex games, explore automated strategy generation, or combine prompt design with retrieval-based methods.

For more sophisticated games without a simple or intuitive optimal move, LLM-based agents have the potential to outperform fixed algorithms. While LLMs are inherently dependent on prompts and can exhibit unpredictable behavior—traits that reflect many real-world game situations—their tendency to approach optimal decision-making or simulate limited human foresight can be guided through carefully designed prompts.

References

[1] Naveed, Humza, et al. "A comprehensive overview of large language models." ACM Transactions on Intelligent Systems and Technology 16.5 (2025): 1-72.
[2] Guo, Taicheng, et al. "Large language model based multi-agents: A survey of progress and challenges." arXiv preprint arXiv: 2402.01680 (2024).
[3] Hu, Sihao, et al. "A survey on large language model-based game agents." arXiv preprint arXiv: 2404.02039 (2024).
[4] Gurcan, Onder. "Llm-augmented agent-based modelling for social simulations: Challenges and opportunities." arXiv preprint arXiv: 2405.06700 (2024).
[5] Costarelli, Anthony, et al. "Gamebench: Evaluating strategic reasoning abilities of llm agents." arXiv preprint arXiv: 2406.06613 (2024).
[6] Dalffa, Mohaned Abu, Bassem S. Abu-Nasser, and Samy S. Abu-Naser. "Tic-Tac-Toe Learning Using Artificial Neural Networks." (2019).