Large Language Models Are Neurosymbolic Reasoners

Meng Fang[*1,2], **Shilong Deng**[*1], **Yudi Zhang**[*2],
Zijing Shi[3], **Ling Chen**[3], **Mykola Pechenizkiy**[2], **Jun Wang**[4]

[1]University of Liverpool, United Kingdom
[2]Eindhoven University of Technology, Netherlands
[3]University of Technology Sydney, Australia
[4]University College London, United Kingdom
{Meng.Fang, shilong.deng}@liverpool.ac.uk, {y.zhang5, m.pechenizkiy}@tue.nl,
zijing.shi@student.uts.edu.au, ling.chen@uts.edu.au, j.wang@cs.ucl.ac.uk

Abstract

A wide range of real-world applications is characterized by their symbolic nature, necessitating a strong capability for symbolic reasoning. This paper investigates the potential application of Large Language Models (LLMs) as symbolic reasoners. We focus on text-based games, significant benchmarks for agents with natural language capabilities, particularly in symbolic tasks like math, map reading, sorting, and applying common sense in text-based worlds. To facilitate these agents, we propose an LLM agent designed to tackle symbolic challenges and achieve in-game objectives. We begin by initializing the LLM agent and informing it of its role. The agent then receives observations and a set of valid actions from the text-based games, along with a specific symbolic module. With these inputs, the LLM agent chooses an action and interacts with the game environments. Our experimental results demonstrate that our method significantly enhances the capability of LLMs as automated agents for symbolic reasoning, and our LLM agent is effective in text-based games involving symbolic tasks, achieving an average performance of 88% across all tasks.

Introduction

The ability to perform reasoning is crucial for AI due to its significant impact on various real-world tasks. The widespread adoption of large language models (LLMs), such as ChatGPT and GPT-4 (OpenAI 2023), has led to a series of remarkable successes in reasoning tasks, ranging from question & answering to solving math problems. Among these challenges, text-based games serve as important benchmarks for agents with natural language capabilities and have garnered significant attention in the realm of language-centric machine learning research (Narasimhan, Kulkarni, and Barzilay 2015; Côté et al. 2019; Xu et al. 2020; Ryu et al. 2022; Shi et al. 2022). In these games, an agent uses language to interpret various scenarios and make decisions. The complexity of such games arises from the need for language comprehension, common sense, managing action spaces with combinatorial complexity, and the crucial importance of long-term memory and planning (Côté et al. 2019; Wang et al. 2022a). The challenges escalate in text-based games that involve symbolic tasks (Wang et al. 2022b). For instance, contemporary agents might be tasked with a scenario where they are required to solve a mathematical problem and simultaneously gather a specified amount of fruits, with the quantity needed being the solution to the math problem.

Using symbolic modules or external tools for arithmetic, navigation, sorting, and knowledge-base lookup is crucial for language agents, especially in complex text-based games (Lample and Charton 2020; Poesia, Dong, and Goodman 2021; Wang et al. 2022b; Qian et al. 2023). However, effectively integrating these aspects into language agents remains a relatively unaddressed challenge. Solving such text-based games requires interactive multi-step reasoning, and agents have most commonly been modeled using reinforcement learning (Xu et al. 2020; Yao et al. 2020; Xu et al. 2021). These methods, however, face challenges such as delayed rewards and difficulty in exploring large action spaces. Recently, there has been an exploration of imitation learning approaches, which utilize human play data (Wang et al. 2022b). While Behavior Cloning (BC) shows potential in effectively addressing these challenges, it often necessitates substantial effort and resources. This is primarily due to the need for acquiring expert data.

Recently, large language models (LLMs) have demonstrated notable in-context generalization capabilities, suggesting the potential to elicit reasoning abilities by prompting these models (Brown et al. 2020; Min et al. 2022). However, the application of LLMs in performing symbolic reasoning remains an under-explored area. Models like GPT-3.5 and GPT-4 have shown the ability to encode extensive information (OpenAI 2023). A significant example of this is their acquisition of substantial knowledge during training, enabling them to approach human-level performance across a wide range of tasks (OpenAI 2023). This indicates the feasibility of utilizing LLMs as neurosymbolic reasoners without relying on labeled gold training data. However, there is currently limited research on utilizing these models for reasoning tasks that involve logic, graphs, or symbolic formulas. The exploration and development of methods that leverage LLMs for symbolic reasoning is highly intriguing and holds significant potential impact.

[*]These authors contributed equally.

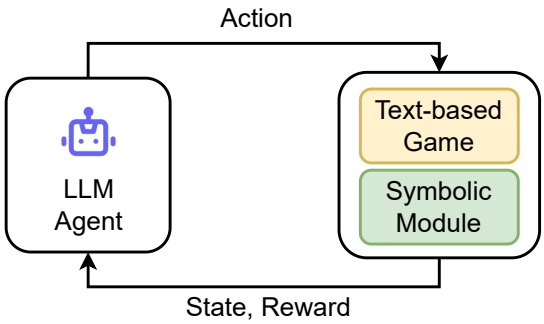

Figure 1: The LLM agent is capable of interacting with the game environment, leveraging its reasoning abilities to determine the most suitable actions. These actions alter the environment's state and contribute to achieving the given objective. The environment, along with its corresponding symbolic modules, offers a valid set of actions to the LLM agent. The agent's responsibility is to select an action from this set. The chosen action will then dictate how the agent interacts with either the game environment or the symbolic module.

In this paper, our aim is to investigate the role of Large Language Models (LLMs) in symbolic reasoning within the context of text-based games. When engaging in games that involve symbolic tasks, our LLM agent generates the most rational actions based on the observed game state in a zero-shot manner, assisted by external symbolic modules such as calculators or navigators, as illustrated in Figure 1. The LLM agent employs both the text-based game environment and symbolic modules to generate a list of valid actions. These valid actions, along with the current observation, are integrated into the prompt to direct the LLM agent in selecting an appropriate action. Subsequently, the LLM agent executes this action, interacting with both the game environment and symbolic modules to complete the task.

In summary, our contributions include:

- We introduce the use of LLMs for symbolic reasoning and provide a framework for employing the LLM agent as a neurosymbolic reasoner. This achievement underscores the potential of LLMs, with the support of external modules, to function as neurosymbolic reasoners, capable of successfully completing complex tasks.
- We have developed the LLM agent with tailored prompts, enabling it to effectively utilize symbolic modules and enhance its performance in text-based games that involve symbolic tasks.
- Our experiments demonstrate that our agent outperforms strong baselines, including the Deep Reinforcement Relevance Network with symbolic modules and the Behavior Cloned Transformer trained with extensive expert data, achieving an average performance of 88% across all tasks.[1]

[1]Code at: https://github.com/hyintell/LLMSymbolic.

Related Work

Large Language Models for Decision Making. LLMs have demonstrated notable capabilities, enabling their application in tasks that extend beyond language generation (OpenAI 2023). Furthermore, they are increasingly being grounded as policy models for decision-making in interactive contexts (Yang et al. 2023). Current studies focus on enhancing the decision-making capacity of LLMs through techniques such as prompting and in-context learning. For instance, Wei et al. (2022) introduce the Chain-of-Thought (CoT) approach, showing that a sequence of intermediate reasoning steps can enhance decision-making capabilities. Yao et al. (2022) present ReAct, a method for interleaved reasoning and action generation to improve performance in interactive decision-making tasks. Other studies (Singh et al. 2023; Huang et al. 2022a,b; Liang et al. 2023; Vemprala et al. 2023) have explored innovative strategies involving prompt engineering and the utilization of high-level function libraries to enhance the capabilities of LLMs. Additionally, some approaches incorporate mechanisms of self-critique and self-reflection into LLMs, enabling them to refine their generation. For example, Shinn, Labash, and Gopinath (2023) introduce Reflexion, a technique that employs external feedback to detect ineffective actions and engage in self-reflection. Madaan et al. (2023) enable an LLM to offer feedback on its previously generated text and refine it adaptively. Recent attempts have also explored different aspects of LLMs for decision-making. Kwon et al. (2023) utilize LLMs as proxy reward functions by prompting them with desired behaviors, while Brooks et al. (2022) consider LLMs as world models, where the agent learns policy by interacting with the LLM-based world model. In our work, we focus on developing suitable prompting strategies to enhance the decision-making performance of LLMs in solving symbolic tasks.

Text-based Game. Text-based games can be formally characterized as partially observable Markov decision processes (POMDPs) (Côté et al. 2019). In recent years, there has been a notable increase in the design of reinforcement learning (RL) agents to solve these games (Liu et al. 2021; Hendrycks et al. 2021; Osborne, Nõmm, and Freitas 2022). Current research primarily addresses challenges such as long-term dependencies, partial state observations, sparse rewards, and complex action combinations in text-based games (Yin and May 2019; Ammanabrolu and Hausknecht 2020; Kimura et al. 2021b; Xu et al. 2022). For instance, Adhikari et al. (2020) address the challenge of partial observability by exploring the acquisition of graph-structured state representations through data-driven methods. Yao et al. (2020) and Shi et al. (2022) employ a language model to generate a compact set of action candidates for RL agents, tackling the issue of the combinatorial action space. More recently, with the advancement of LLMs, research has shifted towards using prompts to enable LLMs to solve text-based games (Yao et al. 2022; Shinn, Labash, and Gopinath 2023). However, these efforts have primarily focused on the LLMs' capability for in-context learning, while the exploration of their potential in symbolic reasoning has been rel-

atively overlooked.

Neurosymbolic Reasoning. The field of neurosymbolic reasoning combines the capabilities of deep neural networks with symbolic reasoning, significantly reducing the search space associated with symbolic techniques. This approach has been used to tackle various complex multi-step inference challenges, including tasks like multi-hop question answering (Weber et al. 2019), language contextualization (Zellers et al. 2021), and semantic analysis (Cambria et al. 2022). Text-based games that involve symbolic tasks serve as a valuable test-bed for addressing such challenges. Previous approaches have employed traditional optimization techniques or reinforcement learning agents. For example, Kimura et al. (2021a) decompose text-based games into collections of logical rules, which are then integrated with deep reinforcement learning. Basu et al. (2022) use Integer Linear Programming (ILP) to substantially improve agent performance, providing an interpretable framework for understanding agents' selection of specific actions.

Preliminaries

Text-based Games as POMDPs. Text-based games can be formally defined as partially observable Markov decision processes (POMDPs), considering that the agent only observes partial information about the environment at each turn (Sutton and Barto 2018). In games with symbolic modules, at each discrete time step t, the agent is provided with an observation denoted as o_t and is given a task description denoted as d. The symbolic module then produces a collection of valid actions, denoted as $A_{t,SyM}$, while the text game environment concurrently establishes its own set of proper actions, denoted as $A_{t,Env}$. Consequently, the set of acceptable actions at time step t is the union of these two sets, denoted as $A_t = A_{t,Env} \cup A_{t,SyM}$. The agent's goal is to select an action a_t from the set of valid actions A_t, given the observation o_t and the task description d. If a_t belongs to the set $A_{t,SyM}$, the symbolic module generates the next observation o_{t+1}. Conversely, if a_t is not part of $A_{t,SyM}$, the text-based game environment processes a_t and produces both the subsequent observation o_{t+1} and the reward r_t.

Symbolic Tasks. There are four distinct tasks within text-based games, namely Arithmetic, MapReader, Sorting and Text World Common Sense (TWC) (Wang et al. 2022b). Each task is equipped with its own symbolic modules designed to assist agents in successfully accomplishing the task.

Methodology

We introduce an LLM agent, namely a language agent, for employing LLMs[2] to engage in text-based games by leveraging symbolic modules in a zero-shot manner. We begin with an overview of playing games using symbolic modules, followed by a detailed description of the key design features of our language agent, including its prompting mechanism.

[2]We utilize LLMs from OpenAI: https://chat.openai.com/.

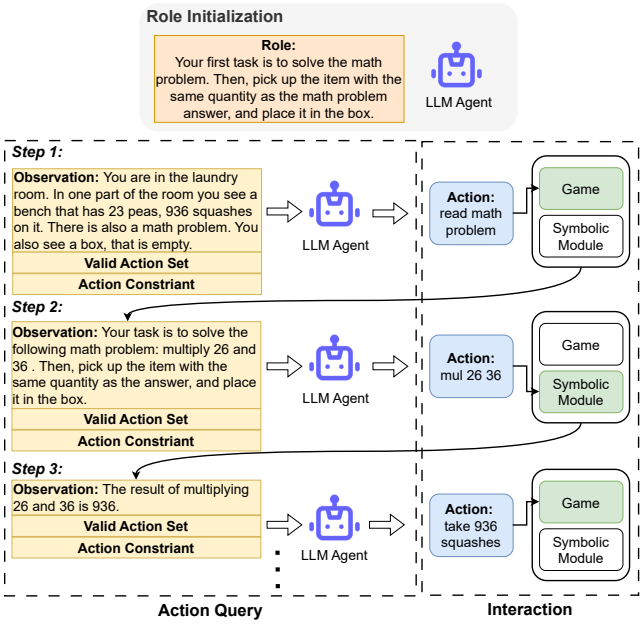

Figure 2: An overview of how an LLM agent plays text-based games with external symbolic modules. The following procedural steps are involved in utilizing the LLM agent for engaging in a text-based game. Initially, the LLM agent is provided with a role initialization prompt. The first observation received by the LLM agent comes from the text game environment. As depicted in the diagram, the selection of actions, determined by the LLM's reasoning, activates the symbolic module. Subsequently, the symbolic module provides output, including observations related to the module. Then the next action chosen by the LLM agent is influenced by the outcome from the symbolic module. This process is executed repeatedly until the end of the game.

Playing Games with Symbolic Tasks

We describe the process of playing games that involve symbolic tasks, using the LLM agent in conjunction with external symbolic modules.

Symbolic Modules. Symbolic modules play a crucial role in maximizing the reasoning capabilities of LLMs. For example, as shown in Figure 2, consider a scenario where a mathematical problem is presented, and a calculator is available. In such cases, the LLM's reasoning can effectively use the calculator to complete the task in a zero-shot manner. Furthermore, symbolic modules are adept at their functions, as employing an external tool like a calculator is considered an action in itself.

The scenarios include four distinct symbolic modules: the Calculation Module, Sorting Module, Knowledge Base Module, and Navigation Module. Table 1 shows examples of how these symbolic modules are utilized. The observation produced by a symbolic module indicates the current state of the game, while the action selected by the LLM agent serves as the input. Additionally, the Navigation Module requires the previous observation as input to accurately determine

Task (Symbolic Module)	Description	Symbolic Module
Arithmetic (Calculation Module)	Your first task is to solve the math problem. Then, pick up the item with the same quantity as the math problem answer, and place it in the box.	INPUT: mul 8 7 RESPONSE: Multiplying 8 and 7 results in 56.
MapReader (Navigation Module)	Your task is to take the coin located in the pantry, and put it into the box found in the chamber. A map is provided, that you may find helpful.	INPUT: next step to pantry RESPONSE: The next location to go to is canteen. If you want to go to pantry from chamber, you need go through canteen, pantry.
Sorting (Sorting Module)	Your task is to sort objects by quantity. First, place the object with the smallest quantity in the box. Then, place the objects with the next smallest quantity in the box, and repeat until all objects have been placed in the box.	INPUT: sort ascending RESPONSE: The observed items, sorted in order of increasing quantity, are: 25 g of oak, 47 g of brick, 15 kg of cedar, 21 kg of marble.
TWC (Knowledge Base Module)	Your task is to pick up objects, then place them in their usual locations in the environment.	INPUT: query clean brown shirt RESPONSE: Clean brown shirt is expected to be located at wardrobe.

Table 1: Text-based games with symbolic tasks and their corresponding symbolic modules. INPUT refers to the current action that is sent to the symbolic modules. RESPONSE denotes the responses generated by the symbolic modules at the present time.

the player's current position. For instance, in a mathematical task, the LLM agent may select a computational action such as "multiply 8 by 7" (mul 8 7). This action triggers the symbolic module to calculate the product, and the resulting observation, "Multiplying 8 and 7 results in 56," is then returned.

The process of engaging in text-based games with LLMs involves multiple stages. The specifics of these steps are detailed in Figure 2. As mentioned earlier, the comprehensive environment, comprising both the symbolic modules and the text-based game environment, presents the LLM agent with a list of allowable actions. Upon receiving an observation, the LLM agent uses its symbolic reasoning to select an action from this list. If the chosen action involves the symbolic module, the module provides the next observation; otherwise, the text-based game environment supplies the subsequent observation.

LLM as the Neurosymbolic Reasoner

We investigate whether the accumulated world knowledge of LLMs can aid in making accurate decisions for downstream symbolic tasks. To ground LLMs in text-based games, we employ a prompting approach, which eliminates the need for costly additional training. Therefore, we construct prompts in a way that incorporates external context, enabling the LLM agent to generate reasonable actions.

We describe the role of the agent, incorporating the observation, valid actions, and the constraints of executing the action in the prompt, as it is not easy for the LLM agent to understand the underlying rules of the environment through interacting with game environments. The key components of our approach include:

- **Role initialization:** We initialize the agent by providing them with task descriptions and action constraints.
- **Action Query:** This step is repeated at each timestep. We prompt the LLM agent with the current observation, inventory state, valid action set, and a question.

Role Initialization	You are a robot. {TASK DESC}\n You are required to choose action from the valid action set to complete the task step by step.\n To take action, respond with an action in the valid action set. \n
Action Query	{OBS}\n {INV STATE}\n Your current score is: {SCORE}\n The valid action set contains: {VALID ACT SET}.\n Please choose one action from the valid action set to finish the task step by step.\n Do NOT respond with any other text, and you cannot decline to take an action.

Table 2: The prompting format for role initialization and action query for each time step. {TASK DESC} is the task description. {OBS} is the current observation. {INV STATE} describes the items in your inventory. {SCORE} is the obtained reward. {VALID ACT SET} is a set of valid actions at the current time step.

- **Answer by the LLM agent:** The LLM agent chooses an action from the valid action set to complete the task.

Role Initialization. We initialize the role and provide instructions for a functional agent assigned to a task. This process informs the agent about its role, the task description, and the actions it can take, along with their explanations and constraints. These actions are necessary for interacting with text-based games or calling the symbolic module. The agent is instructed to choose from a valid set of actions, such as reading the map, getting paths to specific locations, and recalling the task. Additionally, the agent is advised to utilize the external symbolic module and to avoid unnecessary actions during the task.

Action Query. At each timestep, we inform the LLM agent of the current game state, as outlined in Table 2. This information includes the player's observation, the state of

Task	Constrained Prompts
Arithmetic	There are some rules for choosing action: \n 1) If you do not see the items that meet your requirements, please choose 'look around'.\n 2) If you want to put something in the box, please first take it and then put it in box.\n 3) For example, if you want to put 20 apples in the box, you should first choose 'take 20 apples' and then choose 'put 20 apples in box'.\n 4) The next action of 'take math problem' is 'read math problem'.\n 5) However, please never choose 'put math problem in box' as action.\n
MapReader	1) At the beginning choose 'read map' to get the unknown surrounding layout.\n 2) After that, if you do not know how to get to SOMEPLACE, you can choose 'next step to SOMEPLACE' to get the path to SOMEPLACE.\n 3) To choose the action, 'task', you can recall your task.\n 4) Do NOT go to anywhere that is unnecessary for finishing the task.\n
Sorting	To sort the items one by one, please follow the instruction:\n 1) choose 'sort ascending' or 'sort descending' to know which one should be sort next.\n 2) take the items.\n 3) put the items in box.\n
TWC	1) When you take the item, you will get positive score.\n 2) When you put the item in the right place, you will get higher positive score. Otherwise you get 0.\n 3) You are supposed to get as much score as possible.\n

Table 3: The prompting format for adding constraints on the actions of an agent.

the inventory, the reward, and the valid action set. The inventory state refers to the current possessions of the agent. For instance, in mathematical tasks, the inventory state may consist of a mathematical problem, while in the MapReader task, it could include a map. Additionally, the inventory state can encompass tangible objects, such as toothpaste or a quantity of 18 avocados, acquired by the agent within the environment. The LLM agent is then tasked with selecting one action from the valid action set to continue with the task. It is important to note that the LLM agent is not allowed to decline or provide any text beyond the prescribed response. We also limit the number of valid actions provided by the symbolic module.

In addition, it is essential to develop appropriate prompts that effectively restrict the agent's actions according to the information provided in Table 3. It is not feasible for the agent to acquire knowledge and infer the rules within trajectories solely through its interaction with the environment. In all tasks, there is typically a specific order of events, where the object is first taken and then placed in a designated location. This strategy is adopted to prevent scenarios where the object is placed before it is acquired, which would be considered unacceptable in the given context.

Experiments

We demonstrate the potential of LLMs in serving as neurosymbolic reasoners for text-based games. In particular, we present experimental results on four text-based games that involve different symbolic tasks. In these tasks, we observe that LLMs can effectively function as symbolic reasoners.

Setup

We follow the evaluation framework and game environments in Wang et al. (2022b). These games are developed using the TextWorldExpress game engine (Jansen and Côté 2022). For our LLM agent, we use GPT-3.5-turbo. The LLM agent can interact with game environments and symbolic modules. The task descriptions and examples of how the symbolic modules are called are provided in Table 1. The evaluation includes four text-based games involving symbolic tasks. Each task is divided into "Train", "Dev", and "Test" sets. All evaluations are conducted on the "Test" set.

The evaluation metric is based on two factors: the average score achieved at the end of each game, and the average number of steps taken within a single episode.

Environments

We use four text-based game benchmark environments (Wang et al. 2022b):

Arithmetic. The task at hand involves a mathematical component, wherein an agent is required to read and solve a mathematical problem. This process determines the specific object from a given set of objects that they should select and place. The arithmetic game includes a calculator module equipped with the capability to perform basic mathematical operations, including addition, subtraction, multiplication, and division.

MapReader. A pick-and-place game with a navigation theme, similar to the Coin Collector game (Yuan et al. 2018). The agent is equipped with a map that may be exploited to optimize route planning. The map provides information on the connections between rooms, such as the lounge connecting to the cookery and supermarket. The navigation symbolic module has the capability to extract location information from the observation space. This includes specific information relating to the present location and geographical features leading to the intended destination. For instance, the instructions sent to the agent might indicate that in order to get from the cooking area to the recreation zone, one must pass through the bar, steam room, library, and finally reach the recreation zone.

Sorting. This game involves an agent initially situated in a room containing a variable number of objects, ranging from three to five. The agent's task is to sequentially place these objects into a designated box, adhering to a specific sorting criterion based on increasing quantity. In this game, units related to volume, mass, or length are used, as exemplified

	DRRN				Behavior Cloned Transformer				LLM Agent	
	Baseline		+symbolic module		Baseline		+symbolic module			
Benchmark	Score	Steps	Score	Steps	Score	Steps	Score	Steps	Score	Steps
Arithmetic	0.17	10	0.14	7	0.56	5	1.00	5	1.00	4
MapReader	0.02	50	0.02	50	0.71	27	1.00	10	0.86	15
Sorting	0.03	21	0.03	18	0.72	7	0.98	8	0.71	7
TWC	0.57	27	0.37	34	0.90	6	0.97	3	0.94	4
Average	0.20	27	0.14	27	0.72	11	0.99	7	0.88	7

Table 4: The average performance of the model across a set of 100 games in the unseen test set. "+symbolic module" indicates the utilization of symbolic modules within the action space of the models.

by items such as 25g of oak, 12ml of marble, and 6cm of cedar. The sorting game includes a module capable of extracting information from the observation space. This module is specifically designed to identify items that include quantities and can arrange these objects in either ascending or descending order, following the user's instructions.

Text World Common Sense (TWC). The challenges provided in this game serve as a baseline for evaluating common sense reasoning abilities (Murugesan et al. 2021). In this game, agents are required to gather objects from their surroundings, such as a clean brown shirt, and subsequently place these objects in their appropriate and commonly recognized locations, like a wardrobe. The incorporation of a symbolic module within this game enables agents to engage in knowledge-based queries. For instance, it allows them to deduce that a clean brown shirt is typically found in a wardrobe.

Baselines

We also compare our LLM agent with two baselines, namely the Deep Reinforcement Relevance Network (DRRN) (He et al. 2016) and the T5-based Behavior Cloned Transformer (Raffel et al. 2020; Wang et al. 2022b), as follows:

- **DRRN:** The primary concept of the DRRN is based on Q-learning. The candidate action with the highest anticipated Q-value is chosen as the next action, based on the current observation. The DRRN employs a Deep Q-Network (Mnih et al. 2013) to estimate the Q-value for each observation-action pair. Xu et al. (2020) note that the DRRN is a fast and robust reinforcement learning baseline, frequently used to produce near state-of-the-art performance in a variety of text-based games.
- **Behavior Cloned Transformer:** This method adopts an imitation learning approach, conceptualizing reinforcement learning as a sequence-to-sequence problem, similar to the Decision Transformer (Chen et al. 2021). It predicts the subsequent action based on a sequence of previous observations. This baseline aligns with the approach described in Ammanabrolu et al. (2020), where the model input at timestep t includes the task description, current state observation, previous action, and previous state observation. Symbolic modules are utilized in the demonstrations, specifically employing gold trajectories.

Following Wang et al. (2022b), both baseline models include two variants: one with symbolic modules and one without. When using symbolic modules, we inject actions from these modules into the action space of each game for the baseline models.

Results

Based on the results presented in Table 4, it is evident that the use of the symbolic module in conjunction with the LLM agent yields a favorable average performance compared to other baseline approaches. When comparing the outcomes of the Behavior Cloned Transformer with a symbolic module to those of the LLM agent, the performance of the LLM agent is observed to be slightly lower. However, the LLM agent demonstrates a similar level of competency in interacting with the game environment. Furthermore, unlike the Behavior Cloned Transformer models, the LLM agent does not require extensive training with a large volume of expert data. As a result, this approach saves significant training resources.

Table 5 demonstrates that the LLM agent possesses a robust capacity for reasoning, enabling effective handling of tasks involving symbolic tasks. It shows exceptional performance, particularly in mathematics. In the MapReader benchmark, the agent achieves commendable scores, though it requires a considerable number of steps to complete the task. This inefficiency is mainly due to the agent's tendency to forget the route obtained from the symbolic module, leading to the risk of reaching incorrect locations and necessitating repeated route queries. The complexity of map logic, which involves determining one's current location and desired destination, adds to the probabilistic nature of this task. In contrast, the Sorting task reveals suboptimal performance, as the LLM agent's understanding of sorting logic is not fully developed. This issue is largely attributed to the agent's limited memory capacity, hindering its ability to remember the ascending order of all objects.

In Table 6, it compares the performance of the model with constrained prompts to that of the model without constrained prompts. The results indicate that when the LLM agent is provided with the prompts outlined in Table 3, there is an improvement in performance across all tasks. Additionally, a reduction in the average number of steps required to interact with the game environment is observed. This demonstrates the effectiveness of our constrained prompts in these tasks. Furthermore, experimental results using GPT-4, as shown in

Task	Train		Dev		Test	
	Score	Steps	Score	Steps	Score	Steps
Arithmetic	1.00	3	0.95	4	1.00	4
MapReader	0.84	15	0.84	14	0.86	15
Sorting	0.70	7	0.63	6	0.71	7
TWC	0.93	4	0.835	5	0.94	4
Average	0.87	7	0.81	7	0.88	7

Table 5: The performance of the LLM agent on different sets of the game, including "Train", "Dev", and "Test". The scores are subjected to normalization, resulting in values ranging from 0 to 1, with higher values indicating greater performance. On the other hand, the steps quantify the number of actions taken by an agent inside the environment, with lower values indicating more efficient behavior.

Task	With Constraints		Without Constrains	
	Score	Steps	Score	Steps
Arithmetic	1.00	4	0.96	3
MapReader	0.86	15	0.64	12
Sorting	0.71	7	0.35	10
TWC	0.94	4	0.73	7
Average	0.88	7	0.67	8

Table 6: The performance of the LLM agent with and without constrained prompts on the "Test" set. The constrained prompts are shown in Table 3.

Table 7, reveal that it significantly outperforms the GPT-3.5 agent in the MapReader and Sorting tasks, while showing weaker performance in the TWC task.

Discussion. Our results demonstrate that the incorporation of external symbolic modules by the LLM agent leads to enhanced average accuracy compared to other baselines. This capability is achieved by leveraging the underlying patterns present in the training data. Instead of relying on symbolic thinking or explicit rules, this approach acquires knowledge by recognizing patterns and associations from the extensive corpus of text to which it has been exposed during its training phase, as exemplified by GPT-3.5 and GPT-4 (OpenAI 2023). Although the LLM agent has the capability to connect with a symbolic module for specific tasks, it still exhibits uncertainty and is prone to making mistakes.

Conclusion

This paper has demonstrated the effective application of Large Language Models (LLMs) in complex text-based games involving symbolic tasks. Utilizing a prompting approach, we have guided the LLM agent to efficiently engage with symbolic modules within these games. The efficacy of our method, leveraging LLMs, has shown superior performance compared to alternative benchmarks, highlighting the potential of LLMs to enhance training procedures in text-based games. Consequently, it can be posited that Large Language Models can be considered as Neurosymbolic Reasoners, possessing significant potential for performing symbolic tasks in real-world applications.

Task	With GPT-3.5		With GPT-4	
	Score	Steps	Score	Steps
Arithmetic	1.00	4	1.00	4
MapReader	0.86	15	0.99	7
Sorting	0.71	7	0.93	8
TWC	0.94	4	0.71	16
Average	0.88	7	0.91	8

Table 7: The performance of the LLM agent using GPT-3.5 and GPT-4 on the "Test" set.

Limitations

The addition of more detailed prompts could offer greater control over the actions of the LLM agent. This would be particularly beneficial in tasks like Sorting, where providing essential information beforehand is advantageous. Acknowledging and addressing these limitations could significantly enhance the system's performance. For future progress, it is crucial to extend the model's application to more complex domains, going beyond the scope of straightforward text-based games. Integrating more sophisticated symbolic modules would be necessary to tackle the complexities of diverse scenarios, thereby facilitating a more efficient problem-solving approach.

References

Adhikari, A.; Yuan, X.; Côté, M.-A.; Zelinka, M.; Rondeau, M.-A.; Laroche, R.; Poupart, P.; Tang, J.; Trischler, A.; and Hamilton, W. 2020. Learning dynamic belief graphs to generalize on text-based games. *Advances in Neural Information Processing Systems*, 33: 3045–3057.

Ammanabrolu, P.; and Hausknecht, M. 2020. Graph constrained reinforcement learning for natural language action spaces. *arXiv preprint arXiv:2001.08837*.

Ammanabrolu, P.; Urbanek, J.; Li, M.; Szlam, A.; Rocktäschel, T.; and Weston, J. 2020. How to motivate your dragon: Teaching goal-driven agents to speak and act in fantasy worlds. *arXiv preprint arXiv:2010.00685*.

Basu, K.; Murugesan, K.; Atzeni, M.; Kapanipathi, P.; Talamadupula, K.; Klinger, T.; Campbell, M.; Sachan, M.; and Gupta, G. 2022. A hybrid neuro-symbolic approach for text-based games using inductive logic programming. In *Combining Learning and Reasoning: Programming Languages, Formalisms, and Representations*.

Brooks, E.; Walls, L.; Lewis, R. L.; and Singh, S. 2022. In-context policy iteration. *arXiv preprint arXiv:2210.03821*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Cambria, E.; Liu, Q.; Decherchi, S.; Xing, F.; and Kwok, K. 2022. SenticNet 7: A commonsense-based neurosymbolic AI framework for explainable sentiment analysis. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 3829–3839.

Chen, L.; Lu, K.; Rajeswaran, A.; Lee, K.; Grover, A.; Laskin, M.; Abbeel, P.; Srinivas, A.; and Mordatch, I. 2021. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34: 15084–15097.

Côté, M.-A.; Kádár, A.; Yuan, X.; Kybartas, B.; Barnes, T.; Fine, E.; Moore, J.; Hausknecht, M.; El Asri, L.; Adada, M.; et al. 2019. Textworld: A learning environment for text-based games. In *Computer Games: 7th Workshop, CGW 2018, Held in Conjunction with the 27th International Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13, 2018, Revised Selected Papers 7*, 41–75. Springer.

He, J.; Ostendorf, M.; He, X.; Chen, J.; Gao, J.; Li, L.; and Deng, L. 2016. Deep reinforcement learning with a combinatorial action space for predicting popular reddit threads. *arXiv preprint arXiv:1606.03667*.

Hendrycks, D.; Mazeika, M.; Zou, A.; Patel, S.; Zhu, C.; Navarro, J.; Song, D.; Li, B.; and Steinhardt, J. 2021. -[What would jiminy cricket do? Towards agents that behave morally](https://arxiv. org/abs/2110.13136),* Advances in Neural Information Processing Systems (Datasets and Benchmarks Track),* 2021. *Advances in neural information processing systems*.

Huang, W.; Abbeel, P.; Pathak, D.; and Mordatch, I. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, 9118–9147. PMLR.

Huang, W.; Xia, F.; Xiao, T.; Chan, H.; Liang, J.; Florence, P.; Zeng, A.; Tompson, J.; Mordatch, I.; Chebotar, Y.; et al. 2022b. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Jansen, P. A.; and Côté, M.-A. 2022. TextWorldExpress: Simulating Text Games at One Million Steps Per Second. In *Conference of the European Chapter of the Association for Computational Linguistics*.

Kimura, D.; Chaudhury, S.; Ono, M.; Tatsubori, M.; Agravante, D. J.; Munawar, A.; Wachi, A.; Kohita, R.; and Gray, A. 2021a. LOA: Logical optimal actions for text-based interaction games. *arXiv preprint arXiv:2110.10973*.

Kimura, D.; Ono, M.; Chaudhury, S.; Kohita, R.; Wachi, A.; Agravante, D. J.; Tatsubori, M.; Munawar, A.; and Gray, A. 2021b. Neuro-symbolic reinforcement learning with first-order logic. *arXiv preprint arXiv:2110.10963*.

Kwon, M.; Xie, S. M.; Bullard, K.; and Sadigh, D. 2023. Reward design with language models. *arXiv preprint arXiv:2303.00001*.

Lample, G.; and Charton, F. 2020. Deep Learning For Symbolic Mathematics. In *International Conference on Learning Representations*.

Liang, J.; Huang, W.; Xia, F.; Xu, P.; Hausman, K.; Ichter, B.; Florence, P.; and Zeng, A. 2023. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 9493–9500. IEEE.

Liu, G.; Adhikari, A.; Farahmand, A.-m.; and Poupart, P. 2021. Learning object-oriented dynamics for planning from text. In *International Conference on Learning Representations*.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

Min, S.; Lewis, M.; Zettlemoyer, L.; and Hajishirzi, H. 2022. MetaICL: Learning to Learn In Context. In Carpuat, M.; de Marneffe, M.-C.; and Meza Ruiz, I. V., eds., *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2791–2809. Seattle, United States: Association for Computational Linguistics.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Murugesan, K.; Atzeni, M.; Kapanipathi, P.; Shukla, P.; Kumaravel, S.; Tesauro, G.; Talamadupula, K.; Sachan, M.; and Campbell, M. 2021. Text-based rl agents with commonsense knowledge: New challenges, environments and baselines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 9018–9027.

Narasimhan, K.; Kulkarni, T.; and Barzilay, R. 2015. Language Understanding for Text-based Games using Deep Reinforcement Learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1–11.

OpenAI. 2023. GPT-4 Technical Report. *ArXiv*, abs/2303.08774.

Osborne, P.; Nõmm, H.; and Freitas, A. 2022. A survey of text games for reinforcement learning informed by natural language. *Transactions of the Association for Computational Linguistics*, 10: 873–887.

Poesia, G.; Dong, W.; and Goodman, N. 2021. Contrastive reinforcement learning of symbolic reasoning domains. *Advances in neural information processing systems*, 34: 15946–15956.

Qian, J.; Wang, H.; Li, Z.; Li, S.; and Yan, X. 2023. Limitations of Language Models in Arithmetic and Symbolic Induction. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 9285–9298. Toronto, Canada: Association for Computational Linguistics.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1): 5485–5551.

Ryu, D.; Shareghi, E.; Fang, M.; Xu, Y.; Pan, S.; and Haf, R. 2022. Fire Burns, Sword Cuts: Commonsense Inductive Bias for Exploration in Text-based Games. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *Proceedings of the 60th Annual Meeting of the Association for Computational*

Linguistics (Volume 2: Short Papers), 515–522. Dublin, Ireland: Association for Computational Linguistics.

Shi, Z.; Fang, M.; Xu, Y.; Chen, L.; and Du, Y. 2022. Stay moral and explore: Learn to behave morally in text-based games. In *The Eleventh International Conference on Learning Representations*.

Shinn, N.; Labash, B.; and Gopinath, A. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*.

Singh, I.; Blukis, V.; Mousavian, A.; Goyal, A.; Xu, D.; Tremblay, J.; Fox, D.; Thomason, J.; and Garg, A. 2023. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 11523–11530. IEEE.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Vemprala, S.; Bonatti, R.; Bucker, A.; and Kapoor, A. 2023. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2: 20.

Wang, R.; Jansen, P.; Côté, M.-A.; and Ammanabrolu, P. 2022a. ScienceWorld: Is your Agent Smarter than a 5th Grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 11279–11298.

Wang, R.; Jansen, P. A.; Côté, M.-A.; and Ammanabrolu, P. 2022b. Behavior Cloned Transformers are Neurosymbolic Reasoners. In *Conference of the European Chapter of the Association for Computational Linguistics*.

Weber, L.; Minervini, P.; Münchmeyer, J.; Leser, U.; and Rocktäschel, T. 2019. Nlprolog: Reasoning with weak unification for question answering in natural language. *arXiv preprint arXiv:1906.06187*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E. H.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*.

Xu, Y.; Fang, M.; Chen, L.; Du, Y.; and Zhang, C. 2021. Generalization in Text-based Games via Hierarchical Reinforcement Learning. In Moens, M.-F.; Huang, X.; Specia, L.; and Yih, S. W.-t., eds., *Findings of the Association for Computational Linguistics: EMNLP 2021*, 1343–1353. Punta Cana, Dominican Republic: Association for Computational Linguistics.

Xu, Y.; Fang, M.; Chen, L.; Du, Y.; Zhou, J.; and Zhang, C. 2022. Perceiving the World: Question-guided Reinforcement Learning for Text-based Games. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 538–560.

Xu, Y.; Fang, M.; Chen, L.; Du, Y.; Zhou, J. T.; and Zhang, C. 2020. Deep reinforcement learning with stacked hierarchical attention for text-based games. *Advances in Neural Information Processing Systems*, 33: 16495–16507.

Yang, S.; Nachum, O.; Du, Y.; Wei, J.; Abbeel, P.; and Schuurmans, D. 2023. Foundation models for decision making: Problems, methods, and opportunities. *arXiv preprint arXiv:2303.04129*.

Yao, S.; Rao, R.; Hausknecht, M.; and Narasimhan, K. 2020. Keep CALM and Explore: Language Models for Action Generation in Text-based Games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 8736–8754. Online: Association for Computational Linguistics.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Yin, X.; and May, J. 2019. Comprehensible context-driven text game playing. In *2019 IEEE Conference on Games (CoG)*, 1–8. IEEE.

Yuan, X.; Côté, M.-A.; Sordoni, A.; Laroche, R.; Combes, R. T. d.; Hausknecht, M.; and Trischler, A. 2018. Counting to explore and generalize in text-based games. *arXiv preprint arXiv:1806.11525*.

Zellers, R.; Holtzman, A.; Peters, M.; Mottaghi, R.; Kembhavi, A.; Farhadi, A.; and Choi, Y. 2021. PIGLeT: Language grounding through neuro-symbolic interaction in a 3D world. *arXiv preprint arXiv:2106.00188*.