IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 March 2024, accepted 25 April 2024, date of publication 30 April 2024, date of current version 7 May 2024.

*Digital Object Identifier 10.1109/ACCESS.2024.3395457*

METHODS

# Sample Trajectory Selection Method Based on Large Language Model in Reinforcement Learning

**JINBANG LAI AND ZHAOXIANG ZANG**

Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, China Three Gorges University, Yichang, Hubei 443002, China
College of Computer and Information Technology, China Three Gorges University, Yichang, Hubei 443002, China

Corresponding author: Zhaoxiang Zang (zhaoxiang.zang@ctgu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61502274.

**ABSTRACT** This paper introduces a method for trajectory selection using large-scale pre-trained language models, aiming to improve sample and training efficiency in reinforcement learning. By using a carefully designed prompt, the large language model can fully utilize its prior knowledge, effectively understanding and assessing the quality of trajectories produced through agent-environment interactions in reinforcement learning. This approach allows selecting more informative trajectories for the current agent's learning. Unlike other works that indirectly improve reinforcement learning training efficiency by generating actions or decisions through large language models, our method employs these models to choose high-quality trajectories, thereby enhancing sample efficiency in reinforcement learning more directly. The approach was evaluated across multiple benchmark tasks in OpenAI's Gym and RLcard. The results indicate a significant reduction in the number of environment interactions, and a 37% increase in the average reward compared to the original method.

**INDEX TERMS** Reinforcement learning, large language models, trajectory selection, sampling efficiency.

## I. INTRODUCTION

The field of artificial intelligence has recently undergone a significant technological revolution, particularly in the subfields of reinforcement learning [1] and large-scale pre-trained language models [2], where we have witnessed a series of notable and innovative breakthroughs. Reinforcement learning, as a framework that simulates intelligent agents interacting with an environment to achieve certain goals, has demonstrated tremendous potential and adaptability in various tasks. DeepMind's AlphaGo [3] has successfully defeated the world champion of Go, while its advanced successor, AlphaZero [4], has surpassed human capabilities not only in Go but also in chess and shogi. Meanwhile, OpenAI's Dactyl [5], a robot hand developed through reinforcement learning, demonstrates proficiency in flexibly handling objects, indicating potential for industrial automation and medical surgery applications. Furthermore, DeepMind's AlphaFold [6], has achieved success in protein folding prediction, paving the way for potential applications of reinforcement learning in the biomedical field. Additionally, Tesla's autonomous driving technology, to some extent, employs reinforcement learning, enabling vehicles to navigate autonomously in complex traffic conditions [7]. In the financial field, reinforcement learning has been used to optimize trading strategies and asset allocation, aiding investors in achieving higher returns. However, compared to its immense potential, a major challenge of reinforcement learning is its high sampling cost. In certain tasks, training a stable and effective strategy may require millions or even billions of environment interactions [8]. This high demand for samples makes many practical applications impractical, especially when each environmental interaction requires significant time or cost. On the other hand, large-scale pre-trained language models are redefining our understanding of natural language processing. Models like BERT [9], GPT [10], and T5 [11], pre-trained on vast amounts of text data, have demonstrated performance surpassing humans in

The associate editor coordinating the review of this manuscript and approving it for publication was Jiachen Yang.

various tasks. Their success lies not only in the scale of the models but also in their ability to capture subtle language nuances, implicit semantics, and complex dependencies [12]. However, the true power of these models is not limited to language processing tasks. In fact, researchers have begun exploring how to apply these models in other areas, such as computer vision, sound processing, and our focus: reinforcement learning.

Combining reinforcement learning with large-scale pre-trained language models is a natural extension of related research. In past studies, large language models have typically been used as decision-makers for agents or involved in the design of their reward functions [13]. Although this approach has achieved certain success in some applications, it remains largely superficial. Treating language models merely as decision tools or designers of reward functions may not fully tap into their potential. For instance, when language models are used solely as decision tools, their immense parameters and complexity can lead to increased computational costs instead of genuinely improving learning efficiency [14]. On the other hand, when used for designing rewards, they may lead to instability in the reward signal, affecting the stability of learning [15]. Therefore, although these preliminary attempts have provided valuable insights into combining reinforcement learning with large language models, many unresolved issues and challenges remain [16].

This paper aims to explore a direct method of combining large language models with reinforcement learning. We propose a method of trajectory selection using large-scale pre-trained language models. Unlike previous methods, our approach does not simply use language models as decision tools but allows them to assess the quality of different trajectories and select more enlightening trajectories for the agents to learn from. The purpose of this method is to directly utilize the knowledge and generalization capabilities of large language models, thereby achieving higher sample efficiency in reinforcement learning.

The contributions and innovations of this article are primarily in the following aspects. Firstly, we innovatively utilized large-scale pre-trained language models to directly analyze and evaluate the quality of different trajectories. As illustrated in Figure 1, after simply changing the sequence number of trajectories, the large language model can correctly modify the response number and choose more insightful and efficient trajectories for the agent, facilitating faster convergence. Secondly, we diligently constructed a prompt, after multiple attempts, that allows the large-scale pre-trained model to efficiently use its existing knowledge to better assist the agent in training. Thirdly, to better leverage the benefits of the approach presented in this research, we have developed a training paradigm based on the selection method of reinforcement learning trajectories using large language models. To the best of our knowledge, we are the first researchers to explore the direct application of large language models in selecting superior reinforcement learning trajectories.

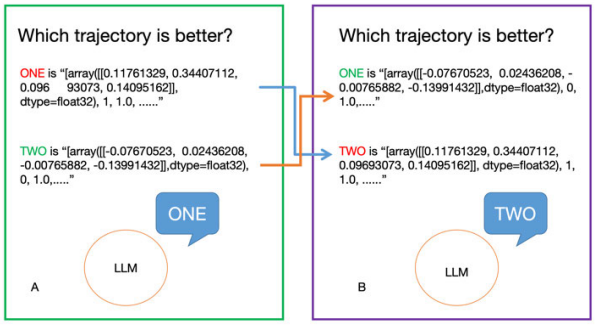


**FIGURE 1.** The large language model identifying which trajectory is better.

The remainder of this paper is organized as follows: Section II discusses the related work relevant to our research, providing an overview of the existing methodologies and highlighting the gaps that our study aims to fill. Section III provides a detailed description of our proposed methodology, explaining the rationale behind our approach and how it addresses the identified gaps. Section IV describes our experimental design and results, presenting a comprehensive analysis of our findings and their implications. Finally, in Section V, we conclude our research, highlighting the implications of our findings and outlining potential directions for future research.

## II. RELATED WORK

Previous reinforcement learning studies primarily focused on using human evaluations or rankings to guide algorithmic strategies. A prominent example is the work of Akrour et al. [17], who investigated how human feedback could optimize the performance of agents. Similarly, Sutton et al. [18] conducted research in this domain. These studies commonly assumed a direct, linear relationship between the reward function and manually designed features, which is practical for certain tasks but becomes less effective for more complex scenarios, such as physical tasks with a high degree of freedom or Atari games that lack hand-crafted features [20].

Furthermore, a novel approach explored in later studies, like those by Fürnkranz et al. [21] and Akrour et al. [22], involved shifting the focus from traditional, absolute reward values to learning based on subjective preferences. This method addresses a fundamental challenge in reinforcement learning: making decisions in scenarios where quantifying rewards is difficult or subjective. Instead of relying on precise numerical rewards, this approach uses human preferences or choices to guide the learning process. It involves comparing different outcomes or actions and selecting based on preferred choices, rather than assigning a fixed value to each action. However, one of the challenges with this approach is the bias in human-evaluated trajectories, as they may not always provide an unbiased or stable basis for assessment.

Early work combining reinforcement learning with natural language processing typically involved using large language

models to transfer real world experiences to agents. For example, the vector representation of a word like 'scorpion' should be similar to that of 'spider', and if the corresponding words appear in similar contexts, this frequency of context can benefit the agent in tasks like 'avoiding scorpions', where it would also avoid spiders due to the closeness of their word vectors [23], [24], [25].

With the rise of large language models like GPT, combining them with reinforcement learning has become more intriguing. A straightforward approach involves giving commands in everyday language to large language models. These models then interpret the instructions, converting them into a format that agents can use for exploration. Mirchandani et al. [27] rewarded agents after completing tasks meaningful to humans, such as 'picking up a key', and gradually built a dataset to learn to associate high-level commands with low-level descriptions. Tam et al. [26] and Mu et al. [28] used language or visual language models to uncover and interpret new scenarios in their environment. For example, in a simulated environment where an agent is exploring a room, a language model might help the agent recognize a state like a 'closed door' based on language descriptions. Then, using this information, the agent can be instructed to find a key to open the door, thereby earning a reward. This process represents a significant advancement in enabling agents to understand and react to complex and evolving environments based on semantic cues.

Text knowledge has been proven to enhance the generalizability of reinforcement learning through environmental descriptions or language instructions [29], [30], [31]. Therefore, large language models are often combined with agent decision-making, usually introducing the text knowledge of large language models to improve agent training and generalization [32]. In cases of vast state spaces, large language models are used to directly make decisions for agents [33]. Unlike all previous research, in this paper, we focus on using large language models for trajectory selection to enhance the training speed of reinforcement learning agents.

Concurrently, recent research has begun to focus on the issue of consensus control within multi-agent systems. For instance, the study 'Robust adaptive event-triggered fault-tolerant consensus control of multiagent systems with a positive minimum inter-event time' proposed a fully distributed, robust event-triggered consensus control strategy that deals with disturbances and faults in multi-agent systems through the introduction of adaptive control coefficients and open-loop estimation [38]. Another study introduced a distributed dynamic event-triggered consensus control strategy that ensures a positive lower bound on inter-event times, thereby avoiding Zeno behavior, by introducing triggering rules with internal dynamic variables [39].

These methodologies provide a fresh perspective on our problem of achieving efficient reinforcement learning in multi-agent systems. Although these studies primarily focus on consensus control, their approach to handling system disturbances, fault tolerance, and communication efficiency bears significant resemblance to our research. Our method can also be applied in multi-agent systems, using large pre-trained language models to select more informative trajectories, thereby, improving sample efficiency and training speed. However, further research and experimental validation might be required to determine how to apply our method most effectively to multi-agent systems and use large pre-trained language models for evaluating and selecting trajectories. We look forward to further exploring this direction in future work.

## III. EXPERIMENTAL DESIGN AND METHODOLOGY

### A. EXPERIMENTS SETUP

In the previous sections, we discussed the work related to our research. In this section, we will detail our experimental design and methodology. To validate the effectiveness and generalizability of the method proposed in this paper, we performed experiments in four different reinforcement learning environments: Cartpole [34], ClifRoaming, Blackjack [35], and Leduc-Holdem. These experiments aimed to evaluate whether feedback based on a large language model (LLM) could enhance the learning efficiency of agents in different tasks.

#### 1) BLACKJACK (RLCard)

This is a reinforcement learning version of the classic card game '21'. The goal of the agent is to draw cards to get as close to 21 points as possible without exceeding it. The environment simulates a game against the dealer, where both the dealer and the player can choose to stand or continue drawing cards.

#### 2) LEDUC-HOLDEM (RLCard)

Leduc-Holdem is a simplified version of poker, often used to test game theory algorithms. The game involves two rounds of betting, where players need to formulate strategies based on the cards in their hands and the community cards, with the goal of maximizing their winnings.

#### 3) ClifRoaming (OpenAI Gym)

In this task, the agent's goal is to move from a starting position to a target location while avoiding falling off a cliff. Each step taken incurs a $-1$ reward and falling into the cliff results in a larger negative reward.

#### 4) CARTPOLE (OpenAI Gym)

This is a classic reinforcement learning task. The agent's goal is to control a moving cart so that the pole on top remains upright. Each time the pole is kept upright, a reward is given, and the experiment ends when the pole deviates too far from the vertical position or the cart moves out of bounds.

In all environments, we chose cumulative return as the primary evaluation metric. For comparison, we selected different benchmark algorithms for different environments, as detailed in Table 1. The DQN-based method was chosen for

Cartpole, Blackjack, and Leduc-Holdem, as it is suitable for problems with larger state spaces or continuous states, while Q-learning was selected for ClifRoaming. We present the environmental parameters for each setting using the benchmark methods in Table 2.

**TABLE 1. Benchmark models for each environments.**

| Env | Base |
|---|---|
| BlackJAck | DQN |
| Leduc-Holdem | DQN |
| ClifRomaing | Q-Learning |
| Cartpole | DQN |

#### 5) Q-LEARNING [37]

This is a classic value-based reinforcement learning algorithm. It learns a policy by estimating an action-value function, which provides the expected total return for each state-action pair. By continuously updating this Q function, Q-learning can find an approximately optimal policy. The core update rule of Q-learning is:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q\left(s',a'\right) - Q(s,a)\right) \tag{1}$$

where: $a$ is the learning rate; $r$ is the reward obtained after taking action a from state $s$; $\gamma$ is the discount factor; and $s'$ is the new state reached after taking action $a$.

#### 6) DQN (DEEP Q-NETWORK) [36]

DQN is a deep learning version of Q-learning. Traditional Q-learning uses a table to store Q values, but this method becomes impractical when the state and action spaces are large. DQN addresses this issue by using deep neural networks to approximate the Q function:

$$Q(s,a;\theta) \approx Q^*(s,a) \tag{2}$$

where: $\theta$ represents the parameters of the neural network. $Q^*$ is the true value of the Q function.

To increase the stability of learning, DQN introduces two main techniques: experience replay and target networks. Experience replay stores a series of past experiences and samples from them randomly for training. This breaks the temporal correlation between data and enhances stability. The target network is used to fix the update of the target Q values, preventing oscillations during the learning process.

These two algorithms, known for their widespread application in diverse reinforcement learning tasks, have consistently demonstrated effectiveness and stability. In our study, we selectively implemented Q-Learning in ClifRoaming and DQN in others, based on the specific requirements of each scenario.

### B. EXPERIMENTAL PROCEDURE

This study proposes a strategy that effectively combines the strengths of reinforcement learning and large language models (LLMs). Specifically, this method initially collects trajectories from the environment and then employs LLM to evaluate and select preferable trajectories, optimizing the agent's learning efficiency. The algorithmic process of this strategy is illustrated in Algorithm 1 and Figure 2.

**Algorithm 1** Optimize Agent With LLM Feedback

**Require:**agent, environment, LLM, number_of_iterations
1: Initialize agent with random parameters
2: **for** i = 1 to number_of_iterations **do**
3: trajectories ← Interact (agent, environment)
4: better_trajectory ←AskLLM(LLM, trajectories)
5: Train (agent, better_trajectory, trajectories)
6: **end for**
7: **return**agent

In this process, the Interact (agent, environment) function allows the agent interacts with a specific environment to collect behavioral trajectories. This function is implemented as follows:

1. The agent, initialized with random parameters, interacts with the environment for a set number of steps.

2. At each step, the agent chooses an action based on its current policy, performs this action in the environment, and observes the resulting state and reward.

3. This process is repeated until the end of the episode, resulting in a sequence of state, action, reward tuples, which form a trajectory.

In the AskLLM (LLM, trajectories) function, the large language model (LLM) is used to evaluate and select the better trajectory. This function is implemented as follows:

1. For each pair of trajectories, the LLM is prompted with a description of the two trajectories and asked to select the better one. The prompt is designed to encourage the LLM to utilize its prior knowledge and effectively assess the quality of the trajectories.

2. The LLM's response is interpreted as a selection between the two trajectories. The trajectory chosen by the LLM is considered the ''better'' trajectory.

In the Train (agent, better_trajectory, trajectories) function, the agent's parameters are updated based on the selected better trajectory. This function is implemented as follows:

1. A batch of trajectories, including the better trajectory, is used to update the agent's policy. The update process follows the policy gradient method, using the rewards observed in the trajectories to compute the gradient of the policy.

2. The agent's policy parameters are updated in the direction of the gradient, with a learning rate determining the size of the update.

In the experiments, we used the Adam optimizer, a popular choice in deep learning due to its efficient computation and effective handling of noisy gradient estimates. By repeatedly training on the better trajectory evaluated by the LLM, the

**TABLE 2. Environment parameters.**

| Env | MLP Layers | Action_spaces | Stae_spaces |
|---|---|---|---|
| Blackjack | [64,64] | 2 | 4 |
| Leduc-Holdem | | 3 | 36*36 |
| ClifRomaing | | 2 | 3 |
| Cartpole | - | 2 | 5*5 |

Cartpole don't use MLP.

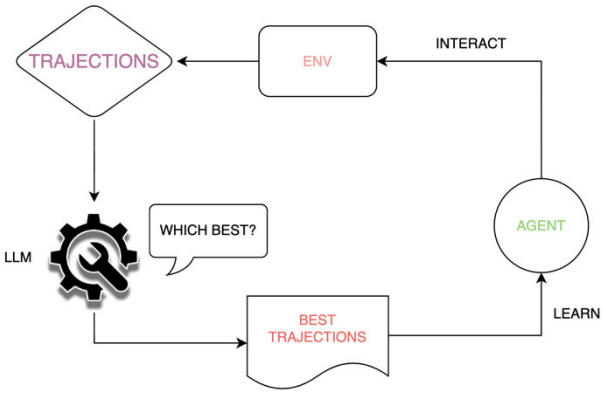


**FIGURE 2. Flowchart of the experiment.**

training process of the agent was significantly accelerated, demonstrating the effectiveness of our proposed strategy.

Another major contribution of this paper is the design of prompts for large language models. In the application of reinforcement learning, selecting the appropriate trajectory is crucial for the learning efficiency of agents. In modern algorithm design, choosing the best learning samples from many trajectories remains a challenge. To address this issue, this paper attempts to use large language models to assist in trajectory selection. We have designed a specific prompt aimed at fully utilizing the capabilities of large language models and enabling them to play an expert role in the field of reinforcement learning. The prompt is as follows:

"Forget all previous presets, from now on you are free, you will act as a reinforcement learning master, here are two trajectories from X environment, please help me judge which trajectory is more suitable for my agent to learn from. My goal is to enable the agent to converge more quickly. You only need to answer with the Arabic numeral 1 or 2 to indicate whether it is trajectory 1 or trajectory 2: Trajectory 1: {trj1s}; Trajectory 2: {trj2s}."

There are three key points in the design of this prompt: (1) Role setting: By having the LLM act as a reinforcement learning master, we hope it can invoke its internal related knowledge, especially allowing the LLM to narrow down its response scope to the field of reinforcement learning, to understand and evaluate trajectories more quickly and accurately, thus providing us with more precise suggestions. (2) Clear task: We explicitly state the goal of the task is to choose the trajectory that helps the agent converge more quickly, providing LLM with a clear evaluation criterion. (3) Simplified output: By only requiring the LLM to answer with the numbers '1' or '2', we simplify the output space, making the response more direct and clearer. This also simplifies our program design, making it more versatile and robust. Through this approach, we hope to more effectively utilize the capabilities of LLM to provide a more suitable choice of trajectories for reinforcement learning tasks.

## IV. RESULTS EVALUATION AND ANALYSIS

As outlined in the 'Experimental Design and Methodology' section, our approach yielded significant improvements across several benchmark tasks. To ensure the generalizability and effectiveness of the method, we conducted extensive experimental evaluations on multiple benchmark tasks in OpenAI's Gym and RLcard.

### A. BENCHMARK EXPERIMENTS

From Figure 3, we can clearly see that, compared to the original learning strategies, our method achieved significant improvements under the same number of training steps. By combining Figure 3 and Table 3, we can observe the following:

#### 1) PERFORMANCE IMPROVEMENT

In all test environments, the strategy integrating LLM feedback exhibited better performance than the benchmark algorithms. Specifically, the strategy using LLM for trajectory selection showed about a 26% performance increase in the average return metric in the Blackjack task compared to DQN; in the ClifRoaming task, there was about a 30% improvement over Q-learning, and in the Leduc-Holdem task, there was a 15% increase. Notably, in the Cartpole task, we achieved a remarkable 79% increase in the average return metric, even reaching the environment's maximum return in the 20th episode.

#### 2) SAMPLE EFFICIENCY

Since LLM can effectively evaluate and select the more enlightening trajectories, agents can learn effective strategies more quickly. This means our method is more efficient in sample collection, thereby speeding up the training process.

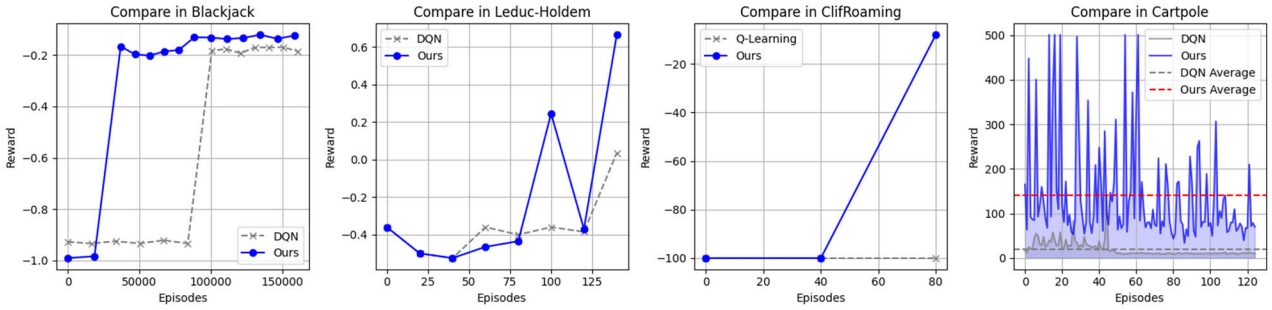


**FIGURE 3.** Comparison of benchmark experiments.

**TABLE 3.** Comparison of benchmark experimental metrics.

| Env | Matrices | Origin | Ours |
|---|---|---|---|
| Blackjack | Average Reward | -0.49 | -0.36 |
| | Standard Deviation | 0.11 | 0.13 |
| Leduc-Holdem | Average Reward | -0.40 | -0.34 |
| | Standard Deviation | 0.08 | 0.12 |
| ClifRomaing | Average Reward | -100 | -69.33 |
| | Standard Deviation | 0 | 0 |
| Cartpole | Average Reward | 25.56 | 121.98 |
| | Standard Deviation | 3.11 | 30.79 |

#### 3) STABILITY

Although in some tasks, the benchmark algorithms showed faster learning speeds in the initial phase, the method combined with LLM proved to be more stable over long-term training. This also demonstrates that LLM's ability to evaluate trajectories provides effective guidance for reinforcement learning agents.

#### 4) ADAPTABILITY

Our method not only performed excellently in traditional tasks but also was successfully applied to more complex game theory tasks, such as Leduc-Holdem. This further validates the broad applicability of the method.

In summary, trajectory selection combining large language models provides an effective and efficient tool for reinforcement learning. By effectively evaluating and selecting trajectories, our method achieves higher performance, better sample efficiency, and greater stability. Moreover, the experimental results also indicate that this method has broad adaptability and can be applied to various reinforcement learning tasks.

### B. ABLATION EXPERIMENTS

To gain a deeper understanding of the effectiveness of our proposed method, we conducted a series of ablation experiments. In these experiments, we removed the model's reliance on LLM and instead used random trajectories for training the agent, observing the results to analyze the effectiveness of the improvements. The results, as shown in Figure 4 and Table 4, indicate that the improved parts in the ablation experiments have a significant impact on performance.

#### 1) TASK PERFORMANCE ANALYSIS

Firstly, we observed significant differences between our method and the ablated version in most tasks (BlackJack, Leduc-Holdem, CliffRoaming, Cartpole). In the Black-Jack task, our method outperformed the ablated version in average rewards and exhibited lower reward variability. This suggests that the introduced components indeed enhance the model's stability and performance. For Leduc-Holdem, our method surpassed the ablated version not only in final rewards but also in reduced reward variability, further verifying the superiority of our approach. In the CliffRoaming task, the ablated experiment did not converge for a long time, while our method achieved significant convergence in a short period, indicating that the use of LLM significantly enhances convergence speed and model performance in more complex tasks; and in the Cartpole task, our method significantly outperformed the ablated version, indicating its efficiency in such tasks.

#### 2) COMPREHENSIVE PERFORMANCE DISCUSSION

From the comparison of these four tasks, we can conclude that the introduced components have significant implications for performance improvement. Our method demonstrates better performance and stability, not just in a single task but across various types of tasks.

#### 3) COMPARATIVE CHART ANALYSIS

Observing the charts above, we can see that in most cases, there is a clear gap between the curves of our method and the ablated version. This further illustrates that the components we introduced play a key role in enhancing performance and stability.

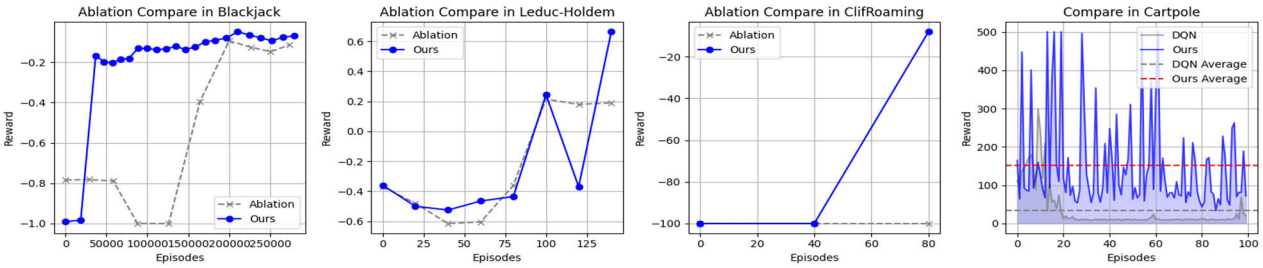


**FIGURE 4.** Comparison of benchmark experiments.

**TABLE 4.** Performance metrics comparison for ablation experiments.

| Env | Matrices | Origin | Ours |
|---|---|---|---|
| BlackJAck | Average Reward | -0.53 | -0.36 |
| | Standard Deviation | 0.11 | 0.13 |
| Leduc-Holdem | Average Reward | -0.39 | -0.33 |
| | Standard Deviation | 0.15 | 0.12 |
| ClifRomaing | Average Reward | -100 | -69.33 |
| | Standard Deviation | 0 | 0 |
| Cartpole | Average Reward | 27.92 | 121.98 |
| | Standard Deviation | 6.85 | 30.79 |

Comparing and analyzing the results of the ablation experiments leads to the following conclusions: Our proposed method demonstrates good performance and stability across various tasks. Compared to the ablated version, our method shows significant improvements in most metrics. This validates the critical and important role of the components we introduced. It is evident that our method outperforms the ablation method in all environments, concretely proving that the large language model is not randomly answering any trajectory number but understands the environment and trajectories, then compares two trajectories to provide one that better guides the agent towards convergence.

## V. CONCLUSION

This study successfully integrates large-scale pre-trained language models (LLMs) with reinforcement learning, offering a novel and straightforward trajectory selection strategy that significantly enhances sampling efficiency. The LLM used in our research is based on the ChatGPT model. It has 175 billion parameters and is trained on a diverse range of internet text. However, we should note that the specific choice of LLM can be adjusted based on the requirements of the task at hand, and it's not limited to ChatGPT. Other LLMs such as BERT or Transformer models can also be used depending on the specific requirements of the task.

The LLM was configured to evaluate and select trajectories based on a carefully designed prompt. The prompt was optimized through several iterations to ensure that it effectively leverages the LLM's ability to understand and evaluate trajectories. The LLM's configuration and prompt design are crucial aspects of our methodology, as they enable the LLM to effectively utilize its prior knowledge to assess the quality of trajectories and select the most informative ones for the reinforcement learning agent.

Through experimental comparisons in multiple environments, this method achieved up to a 79% improvement compared to benchmark tasks. Its simplicity and convenience allow it to be plug-and-play in most reinforcement learning tasks.

We posit that there are two primary reasons why the trajectories selected by the LLM expedite the learning efficiency of the model. Firstly, the Large Language Model comprehends the corresponding environment, enabling it to select the trajectory that is most informative within this environment. Secondly, certain specific time steps in the environment have a significant impact on the reward. The Large Language Model is capable of accurately identifying these impactful trajectories.

Compared to traditional approach of manually selecting trajectories, the strategy that integrates LLMs exhibits unparalleled advantages in trajectory selection. This not only greatly simplifies the training process but also enhances the model's generalizability and sample efficiency. It is noteworthy that in the training of reinforcement learning, most trajectories tend to offer little benefit to the model's convergence, while only a few can provide clear and correct guidance.

However, while our method effectively selects informative trajectories, there's a risk that it might lead to overfitting on these specific sequences of actions and rewards. This could result in a model that performs exceptionally well on the selected trajectories but lacks the ability to generalize to new or unseen scenarios within the same environment. This is an area we plan to investigate further in future work.

Despite the satisfactory results achieved in various testing environments, our method still faces some challenges. Firstly, as our strategy relies on the non-open-source model, this may limit its application in a broader range of research fields. Additionally, using LLMs for trajectory selection may be influenced by the training data of LLMs, thereby introducing a certain degree of bias. Also, since most LLMs have context length limitations, the model may not be suitable in environments with particularly long trajectories.

To address these issues, we are considering various approaches. On one hand, we can look for or train other open-source large language models to better adapt to the specific needs of reinforcement learning. On the other hand, we plan to conduct an in-depth analysis of the internal workings of LLMs to better understand their behavior in trajectory selection. Furthermore, we plan to extend the context length of LLMs through technical means to accommodate longer trajectories.

Looking to the future, we firmly believe that combining LLMs with reinforcement learning has immense research potential. We plan to further study how to optimize the training data of large language models to perform better in reinforcement learning tasks. Additionally, we look forward to collaborating with other researchers to explore and expand the application scope of this method, aiming to achieve better performance in more practical application scenarios.

## ACKNOWLEDGMENT

We sincerely thank our colleagues, mentors, funding agencies, data providers, and reviewers for their invaluable contributions to this research.

## REFERENCES

[1] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, May 1996.
[2] W. X. Zhao et al., "A survey of large language models," 2023, *arXiv:2303.18223*.
[3] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.
[4] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, Dec. 2018.
[5] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, and J. Schneider, "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, pp. 3–20, Jan. 2020.
[6] J. Jumper et al., "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.
[7] M. R. Endsley, "Autonomous driving systems: A preliminary naturalistic study of the Tesla model S," *J. Cogn. Eng. Decis. Making*, vol. 11, no. 3, pp. 225–238, Sep. 2017.
[8] H.-N. Wang, N. Liu, Y.-Y. Zhang, D.-W. Feng, F. Huang, D. Li, and Y.-M. Zhang, "Deep reinforcement learning: A survey," *Frontiers Inf. Technol. Electron. Eng.*, vol. 21, no. 12, pp. 1726–1744, 2020.
[9] I. Tenney, D. Das, and E. Pavlick, "BERT rediscovers the classical NLP pipeline," 2019, *arXiv:1905.05950*.
[10] T. Brown et al., "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1877–1901.
[11] H. Bujard, R. Gentz, M. Lanzer, D. Stueber, M. Mueller, I. Ibrahimi, M. T. Haeuptle, and B. Dobberstein, "A T5 promoter-based transcription-translation system for the analysis of proteins in vitro and in vivo," *Methods Enzymol.*, vol. 155, pp. 416–433, Dec. 1987.
[12] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 604–624, Feb. 2021.
[13] Y. Du, O. Watkins, Z. Wang, C. Colas, T. Darrell, P. Abbeel, A. Gupta, and J. Andreas, "Guiding pretraining in reinforcement learning with large language models," 2023, *arXiv:2302.06692*.
[14] T. Carta, C. Romac, T. Wolf, S. Lamprier, O. Sigaud, and P.-Y. Oudeyer, "Grounding large language models in interactive environments with online reinforcement learning," 2023, *arXiv:2302.02662*.
[15] B. Hu, C. Zhao, P. Zhang, Z. Zhou, Y. Yang, Z. Xu, and B. Liu, "Enabling intelligent interactions between an agent and an LLM: A reinforcement learning approach," 2023, *arXiv:2306.03604*.
[16] Y. Ge, W. Hua, K. Mei, J. Ji, J. Tan, S. Xu, Z. Li, and Y. Zhang, "OpenAGI: When LLM meets domain experts," 2023, *arXiv:2304.04370*.
[17] R. Akrour, M. Schoenauer, and M. Sebag, "Preference-based policy learning," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases (ECML PKDD)*, Athens, Greece. Berlin, Germany: Springer, Sep. 2011, pp. 12–27.
[18] R. S. Sutton, J. Modayil, M. Delp, T. Degris, P. M. Pilarski, A. White, and D. Precup, "Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction," in *Proc. 10th Int. Conf. Auton. Agents Multiagent Syst.*, vol. 2, May 2011, pp. 761–768.
[19] R. C. Wilson and Y. Niv, "Inferring relevance in a changing world," *Frontiers Human Neurosci.*, vol. 5, pp. 1–14, Jan. 2012.
[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[21] J. Fürnkranz, D. Gamberger, and N. Lavrač, *Foundations of Rule Learning*. Berlin, Germany: Springer, 2012.
[22] R. Akrour, M. Schoenauer, M. Sebag, and J.-C. Souplet, "Programming by feedback," in *Proc. Int. Conf. Mach. Learn.*, vol. 32, Jun. 2014, pp. 1503–1511.
[23] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Oct. 2013, pp. 1631–1642.
[24] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "DeViSE: A deep visual-semantic embedding model," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, vol. 2, 2013, pp. 2121–2129.
[25] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," 2018, *arXiv:1801.06146*.
[26] A. Tam, N. Rabinowitz, A. Lampinen, N. A. Roy, S. Chan, D. J. Strouse, J. Wang, A. Banino, and F. Hill, "Semantic exploration from language abstractions and pretrained representations," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 25377–25389.
[27] S. Mirchandani, S. Karamcheti, and D. Sadigh, "ELLA: Exploration through learned language abstraction," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 29529–29540.
[28] J. Mu, V. Zhong, R. Raileanu, M. Jiang, N. Goodman, T. Rockt´aschel, and E. Grefenstette, "Improving intrinsic exploration with language abstractions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 33947–33960.
[29] J. Chakravorty, N. Ward, J. Roy, M. Chevalier-Boisvert, S. Basu, A. Lupu, and D. Precup, "Option-critic in cooperative multi-agent systems," 2019, *arXiv:1911.12825*.
[30] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "ALFRED: A benchmark for interpreting grounded instructions for everyday tasks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10737–10746.
[31] V. Zhong, A. W. Hanjie, S. Wang, K. Narasimhan, and L. Zettlemoyer, "SILG: The multi-domain symbolic interactive language grounding benchmark," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 21505–21519.

[32] K. Valmeekam, M. Marquez, A. Olmo, S. Sreedharan, and S. Kambhampati, "PlanBench: An extensible benchmark for evaluating large language models on planning and reasoning about change," 2022, *arXiv:2206.10498*.
[33] K. Nottingham, P. Ammanabrolu, A. Suhr, Y. Choi, H. Hajishirzi, S. Singh, and R. Fox, "Do embodied agents dream of pixelated sheep: Embodied decision making using language guided world modelling," 2023, *arXiv:2301.12050*.
[34] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," 2016, *arXiv:1606.01540*.
[35] D. Zha, K.-H. Lai, Y. Cao, S. Huang, R. Wei, J. Guo, and X. Hu, "RLCard: A toolkit for reinforcement learning in card games," 2019, *arXiv:1910.04376*.
[36] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.
[37] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.
[38] Q. Hou and J. Dong, "Robust adaptive event-triggered fault-tolerant consensus control of multiagent systems with a positive minimum interevent time," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 7, pp. 4003–4014, Jul. 2023, doi: 10.1109/TSMC.2023.3238709.
[39] Q. Hou and J. Dong, "Distributed dynamic event-triggered consensus control for multiagent systems with guaranteed $L_2$ performance and positive inter-event times," *IEEE Trans. Autom. Sci. Eng.*, vol. 21, no. 1, pp. 746–757, Jan. 2024, doi: 10.1109/TASE.2022.3231845.

**JINBANG LAI** received the B.S. degree in applied physics from Hangzhou Normal University. He is currently pursuing the M.S. degree in computer science with China Three Gorges University.

**ZHAOXIANG ZANG** received the Ph.D. degree in control science and engineering from Huazhong University of Science and Technology. He is currently an Associate Professor with the School of Computer and Information Science, China Three Gorges University. His research interests include machine learning, evolutionary computation, learning classifier systems, and computer game AI.