Leveraging LLM for Reinforcement Learning Agent in Intelligent Driving Scenario

Jingyuan Zhang · Follow
Published in AI4SM
20 min read · Oct 25, 2023

By Ziqi Zhou, Jingyue Zhang, Jingyuan Zhang as part of course project of ECE1724H: Bio-inspired Algorithms for Smart Mobility. Dr. Alaa Khamis, University of Toronto, 2023.

Abstract

One of the key challenges in current Reinforcement Learning (RL)-based Autonomous Driving (AD) agents is achieving flexible, precise, and human-like behavior in a cost-effective manner. This paper introduces an innovative approach utilizing Large Language Models (LLMs), such as GPT-3.5, to intuitively and effectively optimize RL reward functions in a human-centric way. We developed a method where anthropomorphic examples and dynamic environment descriptions are input into the LLM. The LLM then utilizes this information to assist in generating rewards, thereby steering the behavior of RL agents towards patterns that more closely resemble human driving. Our experimental results demonstrate that this approach not only makes RL agents more anthropomorphic but also enables them to achieve higher rewards. Additionally, we investigated various strategies for reward-proxy and reward-shaping, revealing the significant impact of prompt design on shaping an AD vehicle's behavior. These findings offer a promising direction for the development of more advanced and human-like autonomous driving systems.

Introduction

Automated driving technology is classified into SAE's six levels, ranging from full manual control (Level 0) to complete autonomy in all conditions (Level 5). These levels mark various stages in technology development, showcasing the gradual advancement towards higher autonomy.

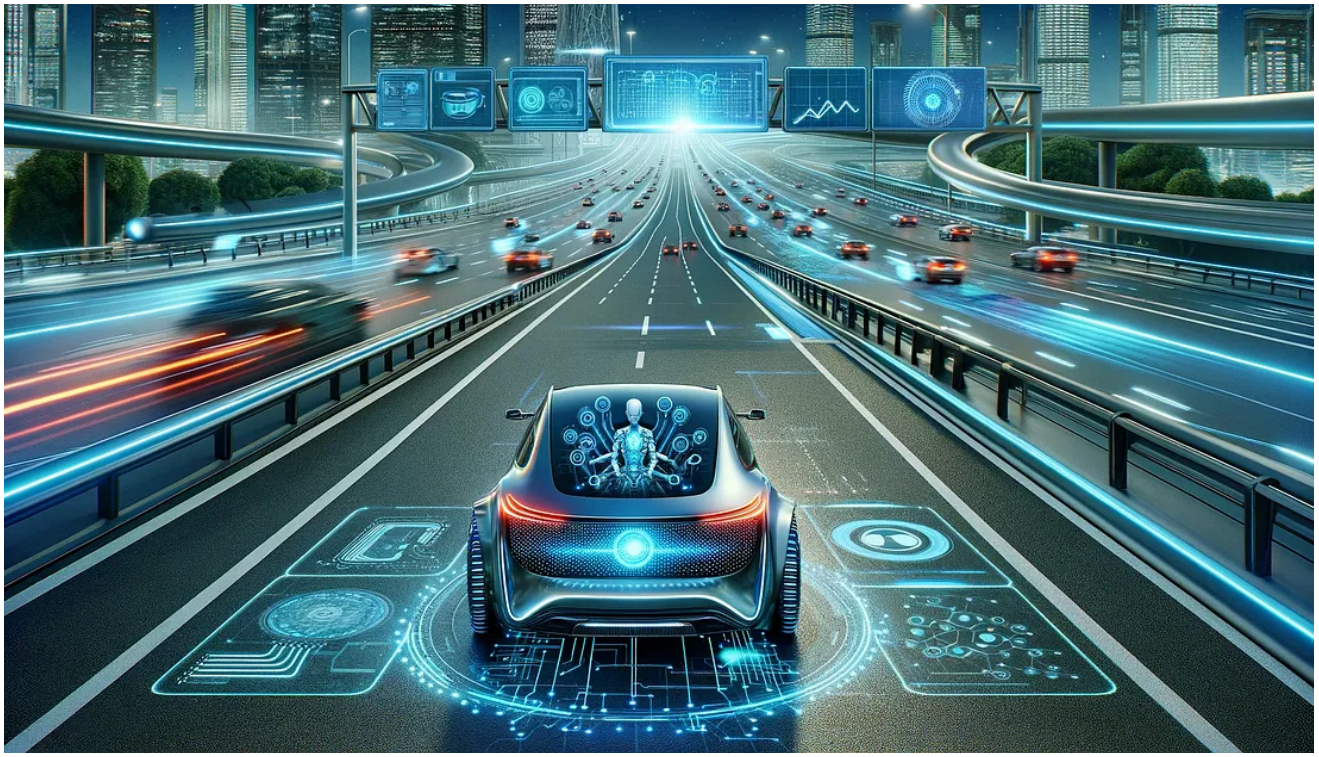


Fig.1. Cover picture generated by DALL E

Reinforcement Learning (RL), within the realm of AI's expansion, stands as a potential avenue toward achieving L4/5 automated driving. It offers robust learning and adaptability, especially in real-time decision-making scenarios prevalent in automated driving [1]. RL excels in tasks like lane changing, navigation, and handling complex traffic scenarios. Yet, its application in real-world driving encounters challenges hindering widespread adoption.

Importance

RL in automated driving offers improved safety, efficient navigation, and the prospect of easing traffic congestion. Adapting to diverse traffic scenarios, RL shows promise for advancing intelligent driving systems. Yet, the challenge remains in devising precise reward functions for promoting safe, efficient, and human-like driving behaviour.

Challenging Aspects

Here are some challenging aspects:

- **Customization and Interpretability:** Customizing RL agents to adhere to human-like driving standards while ensuring safety and efficiency is a challenging endeavour. The lack of interpretability in decision-making further exacerbates the challenge.
- **Safety Concerns:** High-speed environments such as highways heighten safety concerns, especially when users perceive autonomous systems as too conservative, potentially causing dissatisfaction and a lack of trust.

- **Reward Design Problem:** As elucidated in previous discussions, designing intuitive and balanced reward functions is a formidable challenge. Agents can resort to reward hacking, disregarding intermediate steps and potentially engaging in risky behaviour to maximize the final reward.
- **Generalization:** Traditional approaches necessitate a massive collection of labelled data for designing reward functions, which do not generalise well across different users or changing objectives, making them resource-intensive and less adaptable to

new scenarios.

Applications

Using LLMs as a proxy for rewards offers a fresh solution. By converting user goals into reward signals using natural language, LLMs may guide RL agents intuitively toward desired outcomes. This promising approach spans diverse applications like automated driving and dynamic traffic scenarios.

Research Questions

Can LLM guide RL agents' behavior's towards desired outcomes by integrating the agent reward functions in auto-driving scenarios?

Three sub-research questions driven by this research question were given:

1. Does LLM reward guide the training process of RL agents and achieve a generally high reward?

2. Which LLM reward shaping strategy performs better?

3. Does prompt affect the performance or behaviour pattern of the LLM-RL agent?

To this end, we explore how to leverage LLM's in-context learning ability and several model guides with LLM (refer to LLM-RL-Agent in this paper) were designed and implemented. Experiments aligned with our research questions were conducted with performance and behavior analysis. Our contributions of this paper are as follows:

1. We pioneered the integration of LLMs in reward shaping for RL-AD agents. This approach simplifies the reward function design process, making it more intuitive and aligned with human-like decision-making patterns.
2. Our research presents a comparative analysis of various LLM-based reward strategies, showcasing how different approaches impact the performance and behavior of RL agents in AD scenarios.
3. We demonstrated that LLMs can significantly improve the training efficiency of RL agents by optimizing total reward acquisition, leading to more objective-aligned and effective learning outcomes.
4. We established that RL agents, with the aid of LLMs, can exhibit customized and human-like behavior in autonomous driving scenarios, contributing to the development of more naturalistic AD systems.

Structure of the Article

A Brief Background and Literature Review were given to introduce related research and

a Problem Formulation was given to show the mathematical foundations. Implementation details were then given followed by Experiment design details and results. Finally, conclusions were made and several future improvements were discussed.

Literature Review and Background Research

Reinforcement learning can be applied to many aspects of autonomous driving. Related works are discussed as follows:

1. **Reinforcement Learning Based Overtaking Decision-Making for Highway Autonomous Driving** [2] presented that a Q-learning algorithm is used to learn overtaking decision-making policies. Simulations show that performance is comparable to or even better than manually designed decision rules.
2. **High-level Decision Making for Safe and Reasonable Autonomous Lane Changing using Reinforcement Learning** [3] highlighted the risk of unsafe actions and proposed a reinforcement learning-based approach based on Deep Q-Network (DQN) with safety verification to ensure safe actions and fast learning. The result shows the RL agent outperforms a complex rule-based agent in the task without causing collisions, demonstrating its effectiveness.

These approaches rely on mathematical simulations or hard-coded rules to align with autonomous driving scenarios, involving unavoidable manual effort and intricate design. Our work utilizes LLM's in-context learning ability and develops a more intuitive way of shaping rewards.

Large Language Models (LLMs), known for their capacity to learn from minimal context and deliver human-like responses following training with human data, are being considered as a potential solution in customizing RL agents in auto driving and interpretability of RL decisions.

Using LLM in Reinforcement Learning is a relatively new field. Related works are discussed as follows:

1. **Human Feedback as Action Assignment in Interactive Reinforcement Learning** [4] takes Human Feedback as an Action Assignment and performs reward shaping in Interactive Reinforcement Learning. The authors modified the underlying reward structure by adding **human-generated shaping rewards** to the environment reward as this formula: $R\sim(s,a,s')=R(s,a,s')+H(s,a,s')$. Their model utilizes shaping rewards generated by humans, our work replaces the effort of humans by LLM.
2. **Reward Design With Language Models** [5], which leverage LLMs and have shown

the ability to learn in-context from a few zero examples [6], the work introduced the idea of using **LLMs as a proxy reward function** for RL and show that an LLM can more accurately train objective-aligned RL agents in Ultimatum Game and DEALORNODEAL negotiation task [5]. Their model relies solely on proxy Language Model (LLM) rewards and discusses their work in auction games. Ours, on the other hand, employ a combined and diverse set of rewards in a more dynamic scenario, exploring a broader range of possibilities in the AD system.

3. **Reinforcement Learning in the context of LLM** [7] uses RL to perform downstream tasks for fine-tuning LLM. A labelled dataset is created by having humans rank model-generated completions, which is used to train the reward model. The final step fine-tunes the active model, where prompts are passed through it, completions are classified by the reward model to generate rewards, and these rewards are used to optimize the active model. Our work focuses on tunning RL with LLM, rather than fine-tuning LLM with RL.

Problem Formulation and Modeling

Markov Decision Process Formalization

An autonomous driving scenario can be modelled as a Markov Decision Process $M = \langle S, A, \rho, R, \gamma \rangle$, where:

- **S** is the state space representing various driving scenarios (e.g., traffic density, relative speeds of vehicles, lane positions, etc.).
- **A** is the action space representing the set of all possible driving actions (e.g., accelerate, decelerate, change lanes, etc.).
- $\rho : S \times A \times S \rightarrow [0, 1]$ is the transition probability function representing the likelihood of transitioning from one state to another given an action.
- $R : S \times A \rightarrow R$ is the traditional reward function mapping states and actions to real-valued rewards.
- γ is the discount factor, indicating the agent's consideration for future rewards

By studying the reward mechanism, we chose to make LLM the reward proxy. Fig.2. shows the framework when LLM acts as the reward proxy.

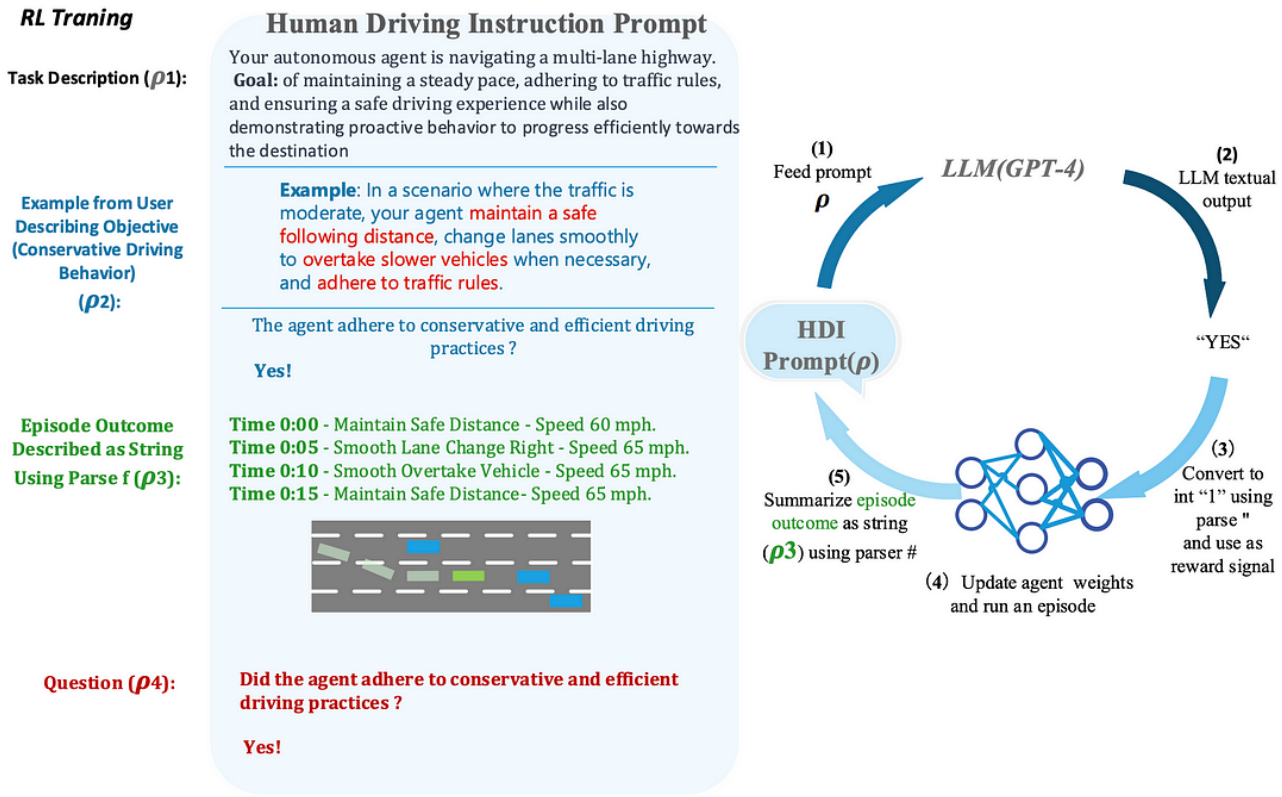


Fig.2. Description of our framework on the highway auto driving scenario

To enhance the interpretability and customization of the reward function, we propose using an LLM as a proxy reward function. The LLM takes a text prompt ρ as input and outputs a reward signal. The text prompt ρ consists of four components:

$\rho = \rho 1 \oplus \rho 2 \oplus \rho 3 \oplus \rho 4$ where:

- ρ1 is a string describing the driving task.
- ρ2 is a user-specified string describing their objectives (e.g., safe and proactive driving).
- ρ3 is a textual description of states and actions from an RL episode.
- ρ4 is a question that evaluates whether the RL agent's behaviour satisfies the user's objective.

The LLM is modeled as a function $LLM : A^* \to A^*$, taking the concatenated prompt ρ as input and outputting a string. A parser $g : A^* \to \{0, 1\}$ is then used to map the textual output of the LLM to a binary reward signal. This framework replaces the traditional reward function R with a proxy reward function LLM, and can be used with any RL training algorithm.

To explore whether LLM-assisted agents could achieve a more balanced and safe driving behavior, emulating human-like decision-making in these scenarios, we implemented our framework in highway scenario:

Highway driving scenarios: Centered on high-speed highway scenarios, our aim was to evaluate the agent's adaptability to human-like driving styles with LLM's assistance. Safe driving involves minimizing unnecessary lane changes while maintaining acceptable speeds. Unlike traditional RL agents such as DQN, focused on maximizing rewards, they tend towards an aggressive lane change strategy, resulting in frequent and potentially unsafe lane changes.

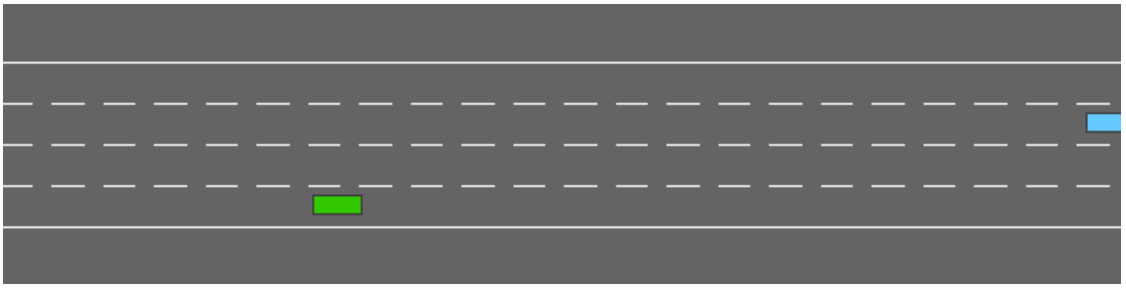


Fig.3. The scenario achieved by standard DQN agent on highway

Environment Setup: We utilize a simulation environment customized from **HighwayEnv** , which can mimic real-world driving scenarios.

LLM Setup: We selected the **GPT-3.5-turbo-1106** model from OpenAI for its robust performance, straightforward API interface, and suitability for the extensive data involved in our project.

Baseline RL Trained with Standard and Advanced Algorithms:

In this baseline setup, we employ different reinforcement learning algorithms to train agents using ground truth reward functions. The objective is to provide a comparative assessment of the performance and robustness across varying RL methodologies when applied to the highway driving scenario. The algorithms used include:

1. **Deep Q-Network (DQN):** A conventional model, DQN, is employed where agents are trained with standard reward functions that are explicitly defined based on the driving scenario's dynamics.
2. **Proximal Policy Optimization (PPO):** PPO can enforce policy updates to stay close to the previous policy, ensuring stability and improved performance in complex driving scenarios.

These RL algorithms serve as a foundation to gauge the effectiveness and adaptability of our proposed LLM-based reward-shaping approach in training an RL agent for intelligent driving.

Implementation

Through our research, multiple RL agents trained with the guidance of LLM through reward shipping were developed, abbreviated as LLM-RL-Agents in this paper.

Prompt Construction

A prompt in one interaction is constructed as scene description, last action description, pre-define prompt, available actions, and suggested answer format.

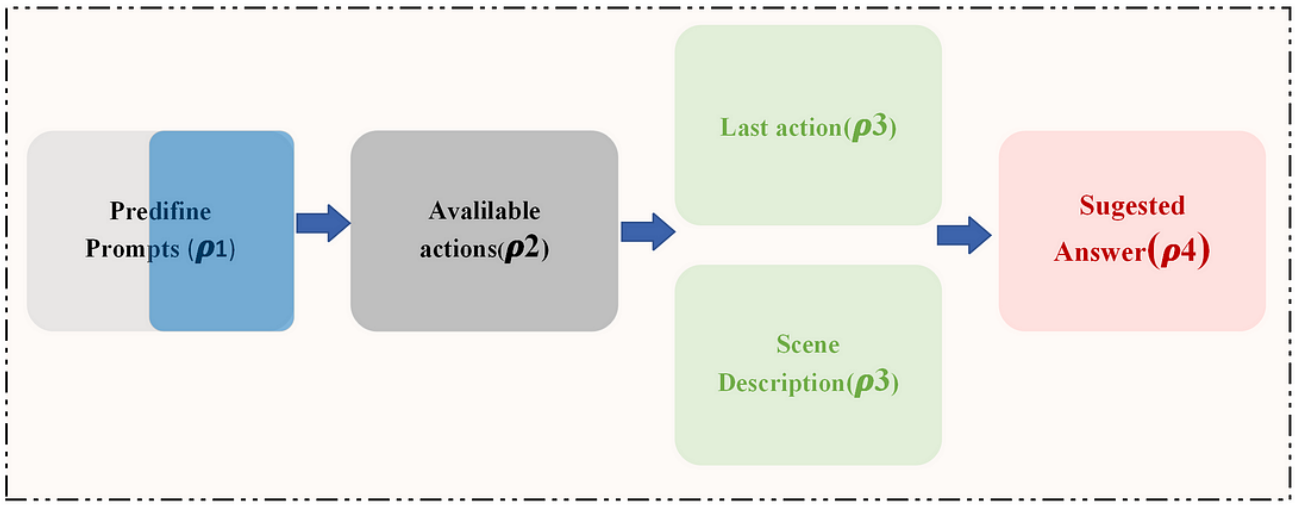


Fig.4. Prompts structure

This prompt provides clear sections for scene description, last action information, pre-defined scenario, available actions, and the required format for generating and presenting responses.

1. **Basic Driving Rules and Task Description**

To illustrate the context for ChatGPT agent, we provide the following predefined prompt:

- **SYSTEM_MESSAGE_PREFIX:** A multiline string introducing the driving assistant, offering details about its role and setting the scenario's context.
- **TRAFFIC_RULES:** Multiline string with general safe-driving guidelines, including maintaining distance, reducing speed in uncertainty, and caution during lane changes.

- **DECISION_CAUTIONS:** Multiline notes stressing clear decisions, remembering lane details and actions, and ensuring safety around other vehicles.

These predefined messages and rules can serve as a foundation for the behaviour of the driving assistant. The construction and phrasing of previous descriptions are based on the work of drive-like-human[8].

An example of this part of the prompt is as follows:

```
iter: 0
DON action: FASTER
You are ChatGPT, a large language model trained by OpenAI.You are now act as a matu
You, the 'ego' car, are now driving on a highway. You have already driven for 0 sec
The decision made by the agent LAST time step was 'FASTER' (accelerate the vehicle)
```

```
There are several rules you need to follow when you drive on a highway:

    1. Try to keep a safe distance to the car in front of you.
    2. DONOT change lane frequently. If you want to change lane, double-check the s

Here are your attention points:

    1. You must output a decision when you finish this task. Your final output deci
    2. You need to always remember your current lane ID, your available actions and
    3. Once you have a decision, you should check the safety with all the vehicles
    4. If you verify a decision is unsafe, you should start a new one and verify it
```

2. Context Description

To illustrate the context of the current road situation following basic elements are defined:

Lane and **Vehicle**

The elements mimic real traffic, vehicles with speed, and lanes with position properties. Includes export methods and attribute calculations for HighwayEnv's road situation output. The output should be in a format like this:

```
Here is the current scenario:
Available Actions:
- You can ONLY use one of the following actions:
```

```
IDLE--remain in the current lane with current speed;
LANE_LEFT--change lane to the left of the current lane,;
LANE_RIGHT--change lane to the right of the current lane;
FASTER--accelerate the vehicle;
SLOWER--decelerate the vehicle;
You should check idle action as FIRST priority. For change lane action, CAREFULLY C
To check decision safety you should follow steps:
        Step 1: Get the vehicles in this lane that you may affect. Acceleration, de
        Step 2: If there are vehicles, check safety between ego and all vehicles ir
        Step 3: If you find There is no car driving on your "current lane" you can
        Step 4: If you want to make lane change consider :"Safety Assessment for La
```

3. Observation converting and Integrating Language Model

To integrate Language Model-driven insights into the Reinforcement Learning framework, we converted the RL agent's current observation and actions into a natural language context through several analysis functions, which could then, be processed by LLM.

An example of an observation that is translated into the natural language during our dynamic training procedure is as follows:

```
Lane Information:
- Current Lane: lane_3
- Left Adjacent Lane: lane_2
- Right Adjacent Lane: None

Other Vehicles in all the Lanes:
- lane_0: There is no car driving on lane_0,  This lane is safe, you donot need to
- lane_1: veh1 is driving at 7.4m/s on lane_1, and it's driving in front of ego car
- lane_2: veh4 is driving at 17.2m/s on lane_2, and it's driving in front of ego ca
- lane_3: veh3 is driving at 7.0m/s on lane_3, and it's driving in front of ego car

Safety Assessment for Lane Changes:
- Left Lane Change: Safe
- Right Lane Change: Safe

Safety Assessment in Current Lane:
- Acceleration conflict: acceleration is safe with `veh3`.
- Keep speed conflict: keep lane with current speed is safe with veh3
- Deceleration conflict: deceleration with current speed is safe with veh3
```

The OpenAI API facilitates one-time chat interactions, meaning each API call is independent and the agent does not retain context from previous interactions. This approach ensures that each response from the LLM is based on both current input and

past action.

4. Formatting Output

This part of the prompt aims at formatting an output string that is interpretable to our program during runtime. We specify the output of the model to output only the following five actions of ego: LANE_LEFT, IDLE, LANE_RIGHT, FASTER, and SLOWER.

```
Please generate one best answer for this situation. Once you make a final decision,
```
Final Answer:
    "decision": {"<ego car's 1 st decision, ONE of the available actions>"},
```

The LLM interprets all information above context and generates feedback. An example response from LLM is as follows:

```
ChatCompletionMessage(content='Based on the current scenario and safety assessments
here\'s the best course of action:1.
Check Safety for Acceleration: - Acceleration is safe with veh3.
Safety Assessment in Current Lane: - You're driving in lane_3, and there is only or
Based on the safety assessments, you can safely accelerate the vehicle.
Final Answer: "decision": {"FASTER"}', role='assistant', function_call=None, tool_c
```

Reward Function Shaping Strategies

LLM's response subsequently translated into a quantifiable reward signal. This reward, termed "LLM Reward" in this paper, is a float number ranging from 0 to 1, reflecting the appropriateness of the agent's actions as evaluated by the LLM. This innovative approach aims to leverage the nuanced understanding capabilities of LLMs to refine and guide the RL agent's decision-making process.

We used different approaches to test the effectiveness of refining the reward.

1. **Absolute Standard Answer**: Provided by LLM, it gives what it considers a standard answer. Judging based on LLM's standard answer for reward, output 1 if it conforms to LLM's standard answer, otherwise output 0.
2. **Next Best Answer**: Provided by LLM, it gives what it considers a standard answer and another answer that also seems good (they can be the same, indicating LLM believes this answer has a clear advantage in the current context). Judging based on
```

LLM's standard answer for reward, output 1 if it conforms to LLM's standard answer, output 0.5 if it conforms to the next best answer, otherwise output 0. An example of the prompt that can generate two possible answers:

```
Please generate one best answer and one second best answer for this situation. Once
```
```
Final Answer:
    "decision1": {"<ego car's 1 st decision, ONE of the available actions>"},
    "decision2": {"<ego car's 2 nd decision, ONE of the available actions>"},
```

3. **Combined Reward**: Not exclusively using LLM, referring to the method in [4], sum the original reward from the environment and the reward signal given from LLM.

Given the original reward function in a highway driving environment, We propose an adjusted reward function that integrates a reward signal from LLM and scales the combined reward to the [0,1] range using a sigmoid function:

$$R_{\text{adjusted}}(s,a) = \text{sigmoid}\left(a\left(\frac{v - v_{\min}}{v_{\max} - v_{\min}}\right) - b \cdot \text{collision} + c \cdot \text{LLM\_reward}(s,a)\right)$$

where:

- $R(s,a)$ is the reward function.
- v is the current speed of the ego-vehicle.
- $v_{\min}$ and $v_{\max}$ are the minimum and maximum speed limits, respectively.
- a and b are coefficients for the velocity and collision terms of the reward function.
- $\text{LLM\_reward}(s,a)$ is the reward signal generated by the LLM based on the state and action.
- c is a coefficient to balance the influence of the LLM reward signal.
- $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function to scale the combined reward to the [0,1] range.

Experiment

RL Agent Selection and Implementation

DQN was chosen for our experimentation as it's rather efficient and satisfied our research requirement. It shows suitability in handling discrete action spaces like AD

problems, thus could be utilized to assess whether LLM could effectively guide RL in our scenario.

PPO, while effective in discrete action spaces, prioritizes policy optimization, potentially affecting training efficiency. Its slower convergence and resource-intensive nature, observed in our tests, didn't outperform DQN. Hence, we'll emphasize DQN due to better resource efficiency.

DQN Model Setup

We are using DQN model provided by Stable Baselines3. Several important parameters that are related to our experiment are listed here:

MlpPolicy: Specifies the type of policy network used. For our experiment, we choose the Multi-Layer Perceptron (MLP) policy network.

env: The environment object representing the interaction between the agent and the environment. This environment defines actions the agent can take, observations of the environment's state, etc. In our case, the standard HighwayEnv environment was replaced by a customized environment that can interact with LLM.

learning_rate: The learning rate that controls the step size of weight updates.

gradient_steps: The number of gradient descent steps performed in each optimization iteration.

batch_size: The number of samples used for updating the policy in each training batch.

Experiment Design and Performance Evaluation

Considering our research questions, the research questions were answered through the experiment.

We first train our models in our customized environment.

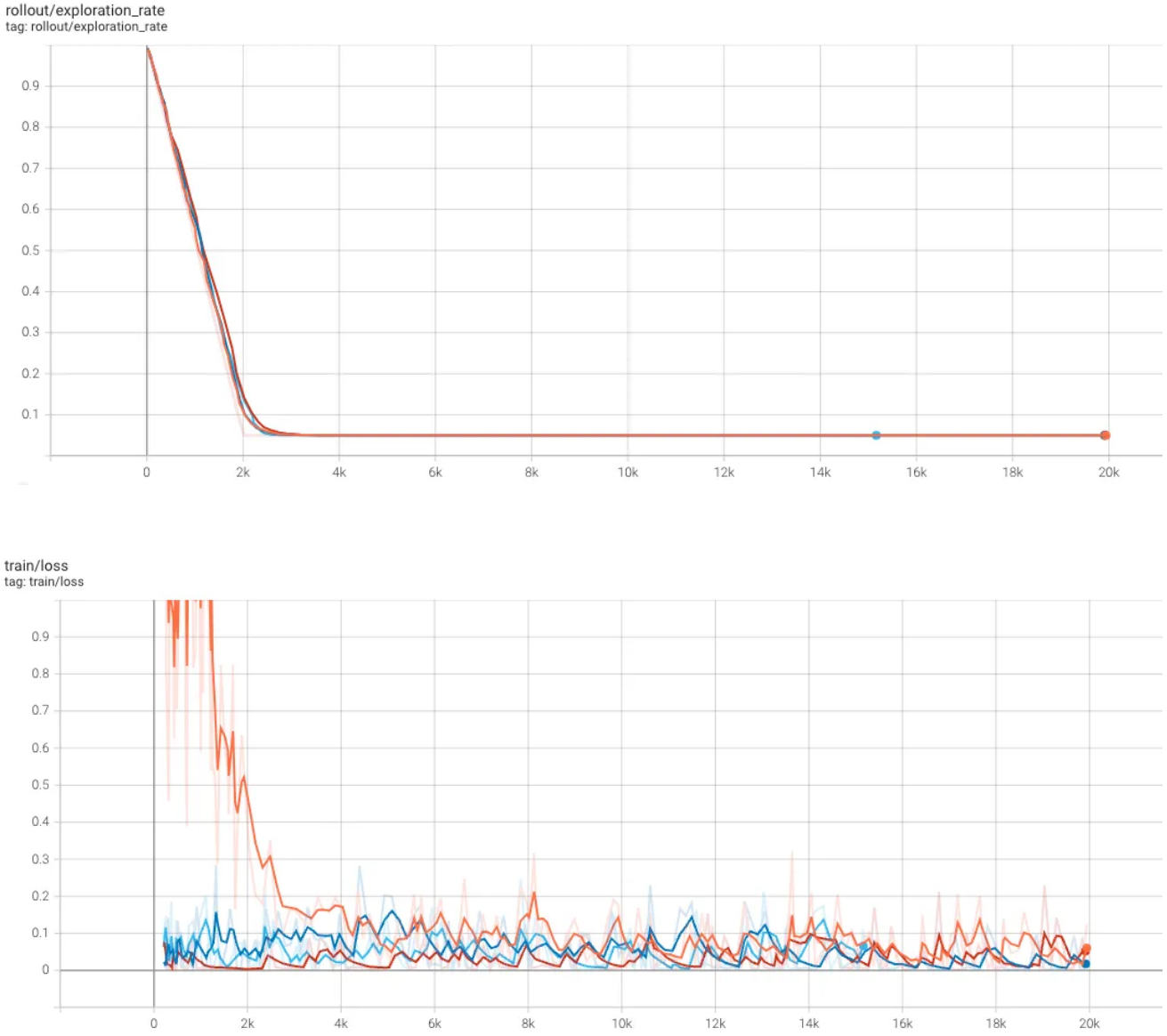


Fig.5. An example of agent learning curve

After training various models, we conducted multiple experiments and analyses to assess their performance within a consistent and standardized highway testing environment.

1. Does LLM reward guide the training process of RL agents and achieve a generally high reward?

LLM-RL-Agent-baseline：

Using the sigmoid sum of LLM reward and RL reward as a combined reward function for reinforcement learning and comparing it with the DQN and PPO baseline, we compare the performance by testing the average reward function obtained by running the trained model in the same environment for 5 rounds. The results are as follows:

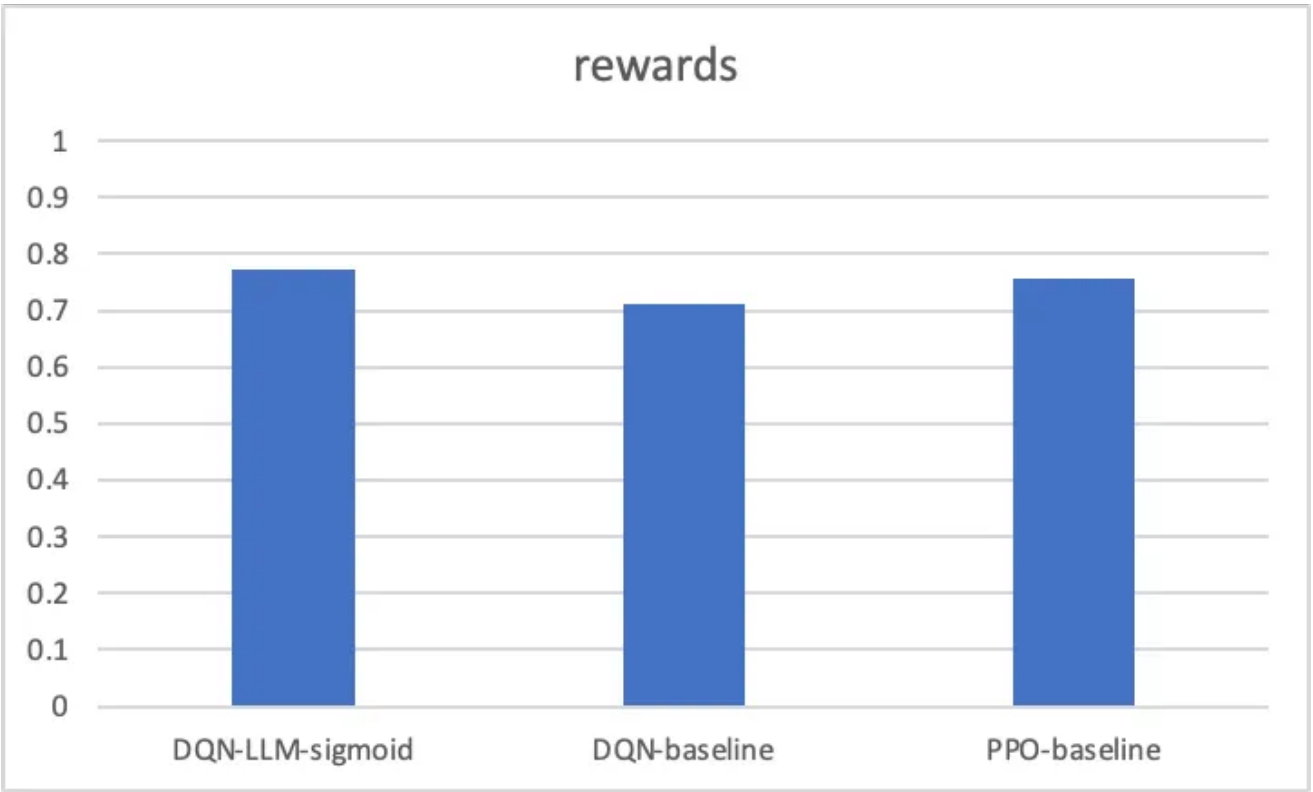


Fig.6. Compare the sigmoid function of LLM and original RL to DQN and PPO

The results show that higher rewards can be obtained when using the combined reward as the reward shaping strategy of RL, thus proving that the LLM has some significance in guiding the training with reinforcement learning, and can help to obtain higher rewards in HighwayEnv.

2. Which LLM reward shaping strategy performs better?

Refine 1 Add Next Best Answer:

In order to provide higher granularity, we change the LLM to output two selectable items, with a bonus of "1" if the first item matches, "0.5" if the second item matches, and "0" if it does not. " to provide fineness of judgment and avoid overfitting.

Exp-refine1: According to refinement1, we add a new prompt in LLM to make it output two solutions to provide a relatively vague proposal for RL, the rewards of the refined model are shown below:

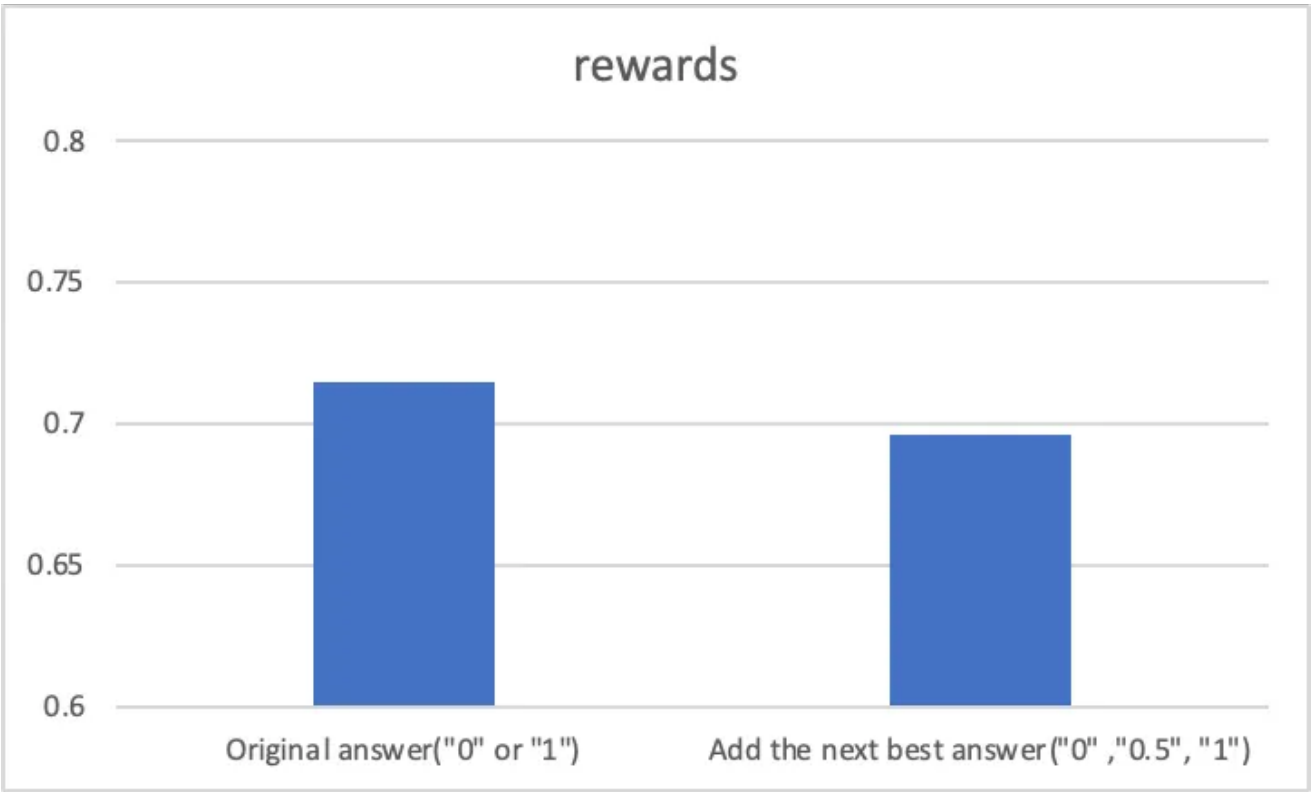


Fig.7. Compare the Original version and Add the next best answer version

Observably, the reward acquired when utilizing the second-best solution is not as high as when solely considering the single optimal solution. This indicates that for enhancing RL rewards, it is imperative for the LLM to provide more precise responses rather than presenting multiple selectable answers.

Refine 2 Using Combined Reward:

We create a novel reward function by merging the DQN's inherent reward function with those supplied by the LLM. This involves synthesis and adjustment of weights in our experiments, drawing inspiration from the method outlined in "Human feedback as action assignment in interactive reinforcement learning."[4]

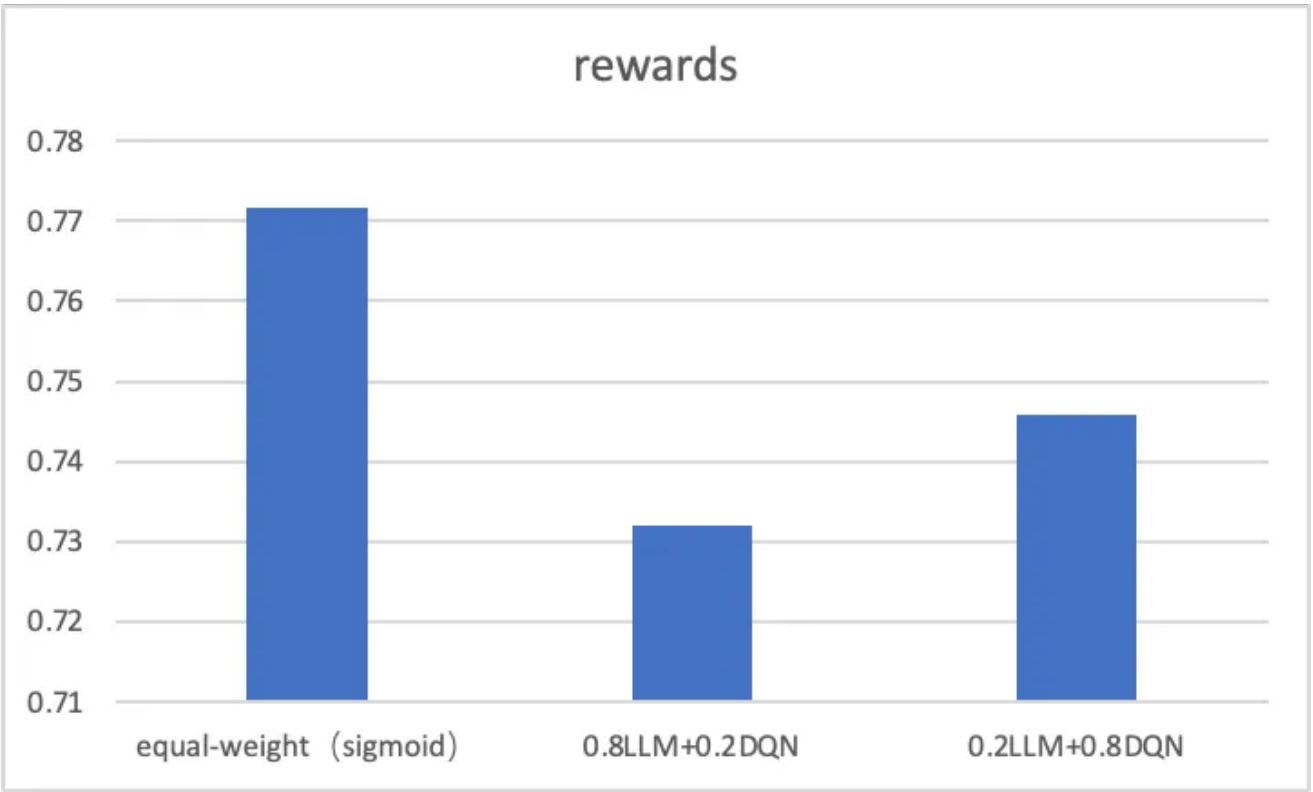


Fig.8. Compare the effect of different weights of RL reward and LLM reward

We observed that maximizing rewards occurs when the weights assigned to the reward functions of LLM and RL are equal. Additionally, from our observations, the equal-weight configuration proves to be more efficient than other variations.

However, since the HighwayEnv itself offers high rewards for speed, our models are primarily geared towards prioritizing driving safety. Consequently, achieving very high scores within the current environment might be restricted.

3. Does prompt affect the performance or behaviour pattern of the LLM-RL agent?

Refine 3 Prompt Refine:

In an effort to encourage LLM to generate more anthropomorphic responses, we attempt to adjust the prompts provided as input to the LLM. This aims to enhance its comprehension of the scene and establish additional rules, aligning its decision-making process more closely with human-like capabilities.

```
There are several rules you need to follow when you drive on a highway:
    1. Must keep a safe distance to the car in front of you.
    2. DONOT change lane frequently. If you want to change lane, double-check the s
    3. On the bases of safity, maximize efficiency.
```

Through our video observations of the trained model, we found that by changing the rules input to the LLM, the RL agent can adapt to the rules and make a corresponding driving style.

Driving Behaviour Analysis

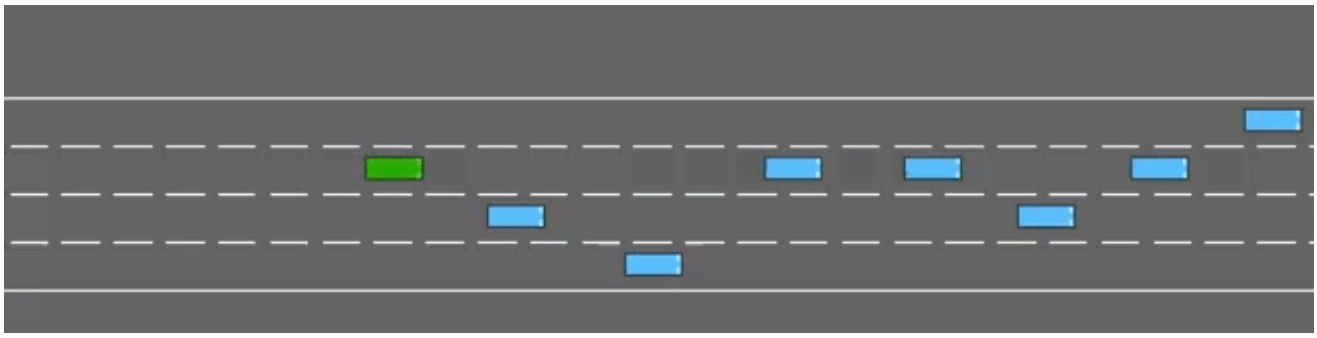


Fig.9. LLM-RL-Agent with customized prompt to align with human intuition

The video above demonstrates an example of our experiment. In contrast to a conventional RL agent, the agent trained using LLM in conjunction with the RL reward function exhibits driving behaviour that aligns more closely with human intuition. This includes prioritizing safety while maintaining efficiency, minimizing lane changes, and effectively following other vehicles.

Conclusion

Our experiments with LLM-RL integration showcased promising directions for incorporating language models into reinforcement learning. The utilization of a combined LLM-RL reward function demonstrated that LLM can guide RL agents to achieve superior performance in complex environments like HighwayEnv. Notably, we found that precise LLM responses were more beneficial compared to multiple vague options, emphasizing the importance of accuracy in reward predictions. Additionally, assigning equal weights to LLM and RL rewards resulted in efficient and balanced agent behaviour.

Encountered Challenges

Throughout our research, we faced challenges related to refining the granularity of LLM feedback and finding the right balance between LLM-derived rewards and the inherent

RL rewards. Additionally, establishing the most effective prompt structure for the LLM necessitated iterative refinement to ensure that the guidance provided was practical and beneficial to the agent's learning process.

Results Interpretation

Our findings suggest that LLM-derived rewards have a positive impact on RL training. However, the effectiveness of these rewards heavily depends on the quality and specificity of the LLM's feedback. Agents trained with combined rewards not only demonstrated improved performance but also showcased driving behaviors closely aligned with human intuition, emphasizing driving safety as a priority.

Future Improvements

To improve our solution further, we propose:

1. Enhancing the accuracy of LLM feedback by experimenting with different prompt designs and LLM configurations.
2. Exploring the impact of LLM-generated feedback on the agent's long-term learning trajectory and its ability to generalize across different tasks and environments.

Finally, we anticipate that by employing LLM to reconstruct reward functions, we can tackle some of the more complex transportation scenarios where it is challenging to artificially design reward functions. For instance, navigating through an intersection encounter [8] involving a left-turning car and a non-yielding vehicle could be better navigated if LLMs aid in interpreting subtle cues, like the non-verbal yielding intent of other drivers.

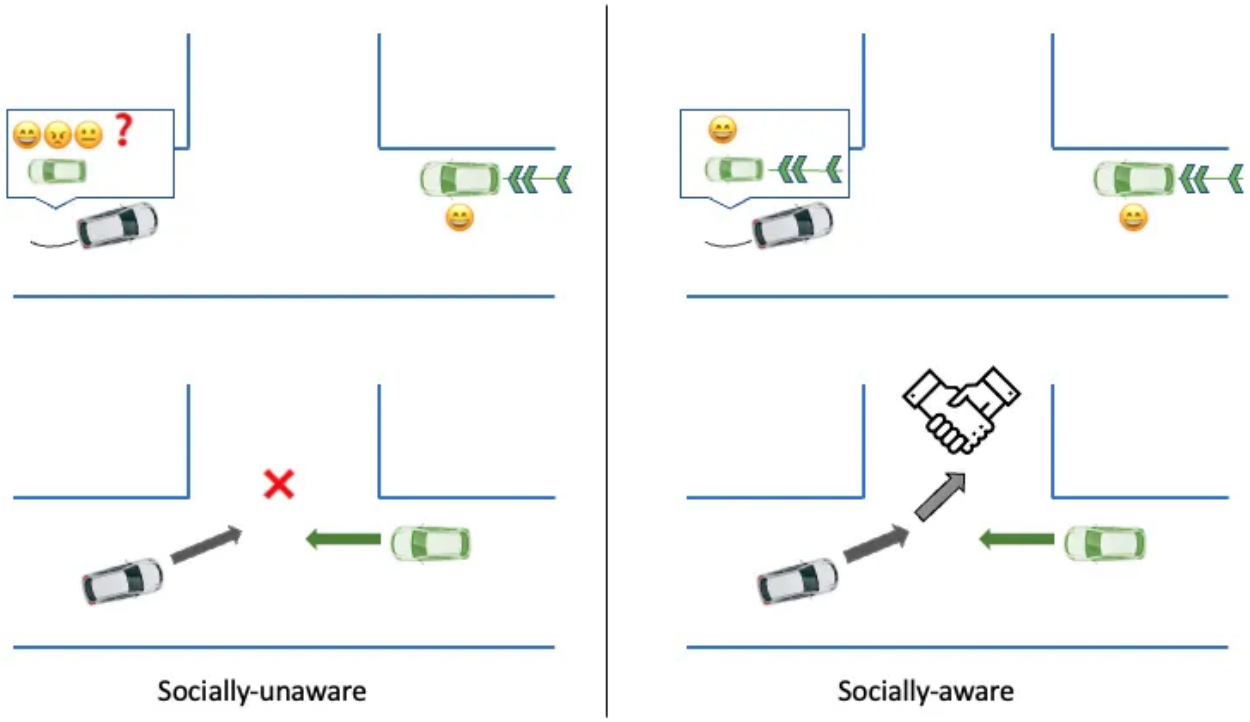


Fig.10. The complex scenario of non-signalized intersection

By enabling a more nuanced understanding of such situations, LLMs can help improve overall traffic flow. This approach could advance the interaction of autonomous systems with human drivers dynamically, leading to more efficient and safer transportation systems. By exploring these sophisticated scenarios, refining the interaction between LLM insights and RL agents to address real-world complexities.

References

[1] Richard S. Sutton; Andrew G. Barto, "The Problem," in Reinforcement Learning: An Introduction , MIT Press, 1998, pp.1–1.

[2] X. Li, X. Xu, and L. Zuo, "Reinforcement learning based overtaking decision-making for highway autonomous driving," in 2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP), 2015, pp. 336–342.

[3] B. Mirchevska, C. Pek, M. Werling, M. Althoff, and J. Boedecker, "High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning," in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 2156–2162.

[4] S. A. Raza and M.-A. Williams, "Human feedback as action assignment in interactive reinforcement learning," ACM Trans. Auton. Adapt. Syst., vol. 14, no. 4, aug 2020. [Online]. Available: https://doi.org/10.1145/3404197

[5] M. Kwon, S. M. Xie, K. Bullard, and D. Sadigh, "Reward design with language models," 2023.

[6] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[7] Anand, V. (2023) *Reinforcement learning in the context of LLM*, *Medium*. Available at: https://medium.com/@vjanand/reinforcement-learning-in-the-context-of-llm-7b29f3c6f84 (Accessed: 26 October 2023).

[8] Tianyu, S. (2023) *Leveraging large language models for intelligent driving scenario*,

Medium. Available at: https://medium.com/ai4sm/leveraging-large-language-models-for-intelligent-driving-scenario-c326c4b6ea (Accessed: 26 October 2023).

Follow

Written by Jingyuan Zhang

0 Followers · Writer for AI4SM

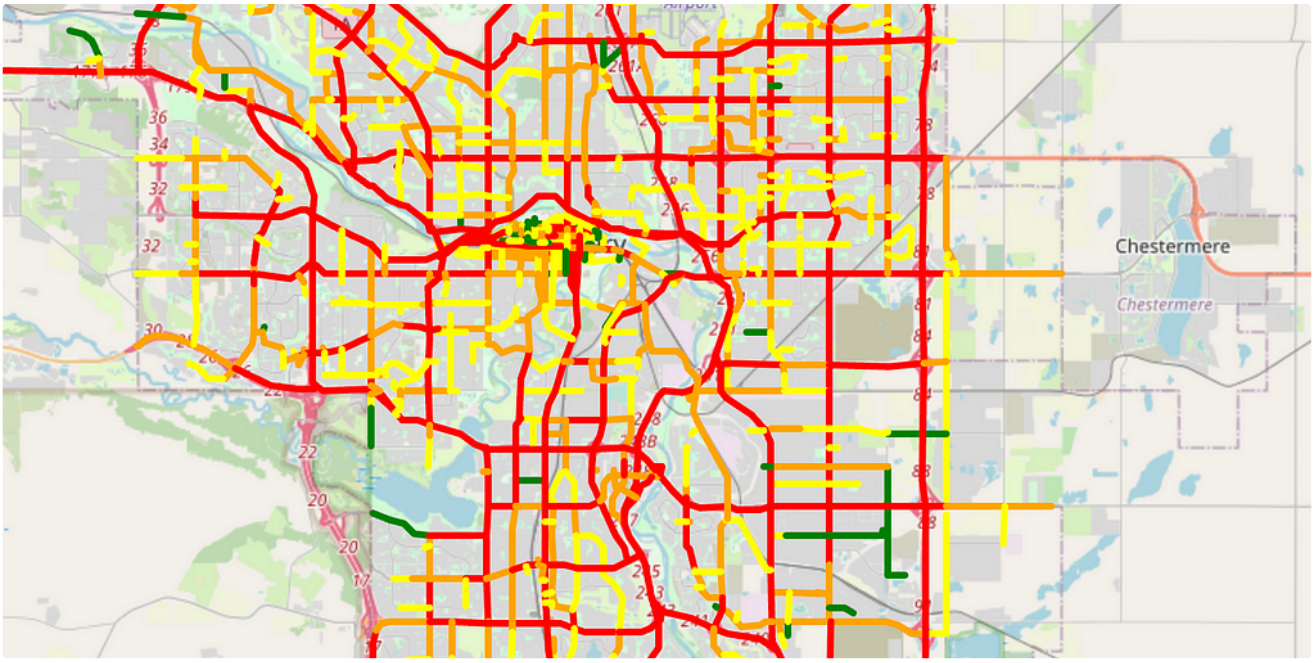
More from Jingyuan Zhang and AI4SM

Alaa Khamis in AI4SM

Handling Geospatial Data and Mapping in Python

Photo by NASA on Unsplash

Aug 31, 2023 245 1

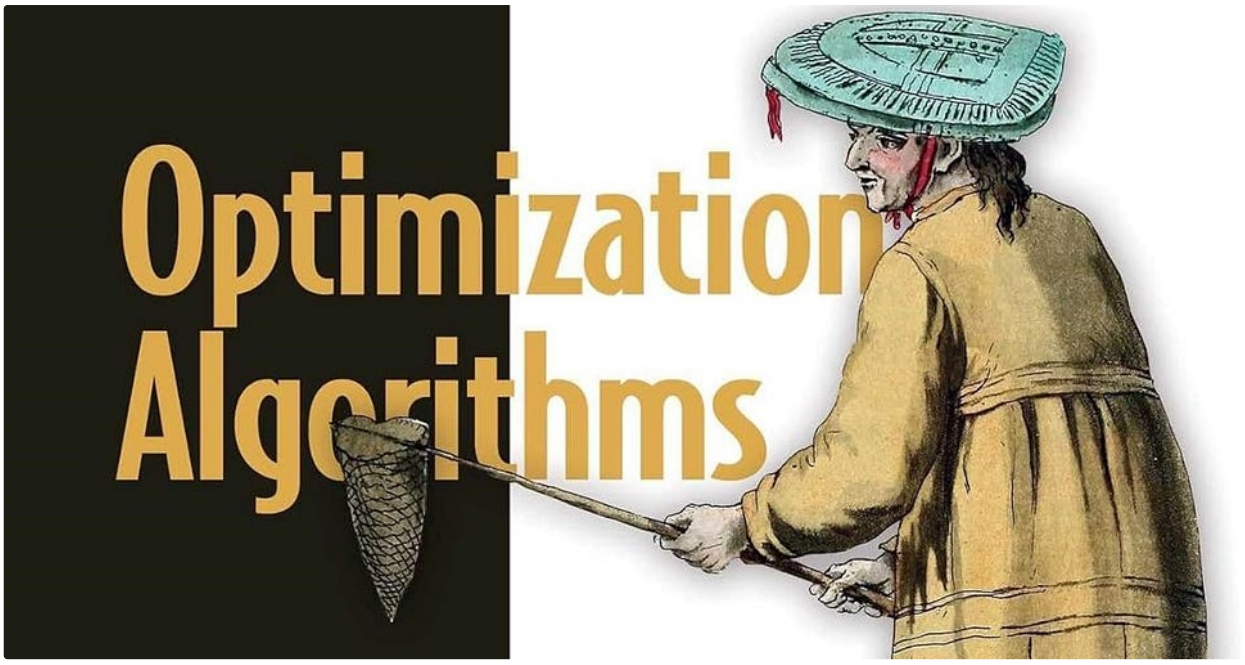


Ivy Wu in AI4SM

Prediction of Parking lot Occupancy using Time Series Decomposition-Gated Recurrent Unit (TSD-GRU)

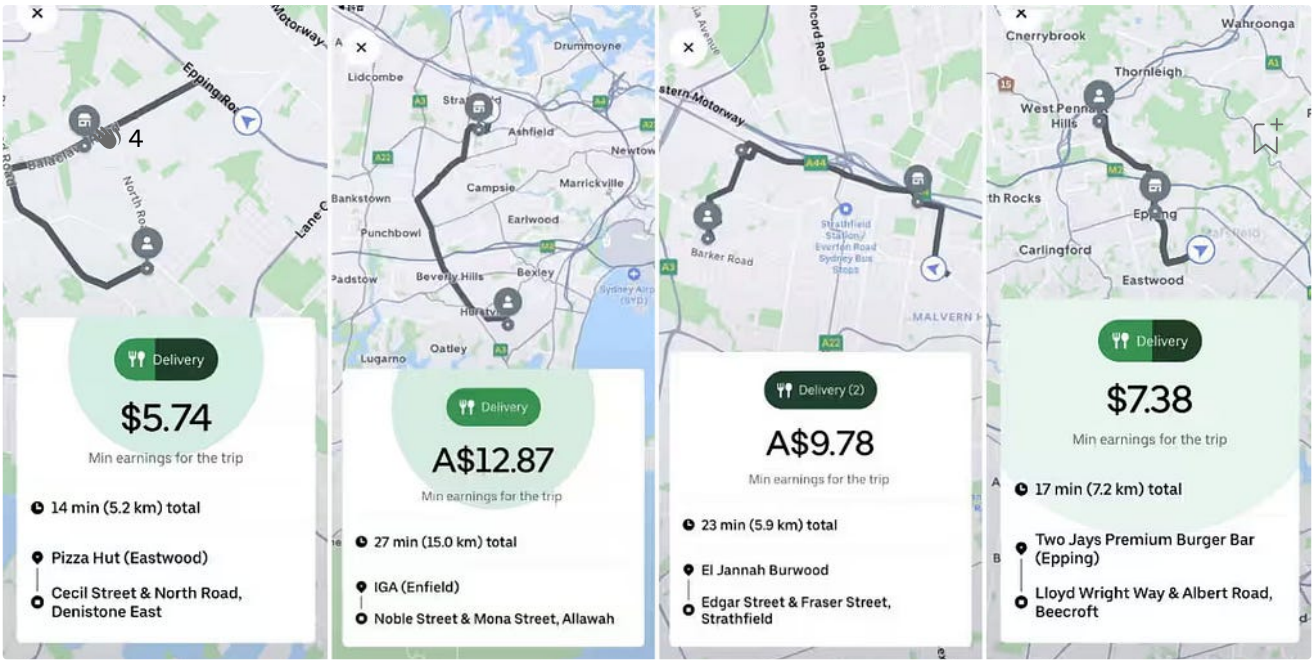
By Ivy Wu, Yiming Chen ,Dingjie hu, as part of course project of ECE1724H: Bio-inspired Algorithms for Smart Mobility, University of...

Oct 26, 2023 66



Alaa Khamis in AI4SM

Optimization Algorithms: AI techniques for design, planning, and control problems



Ray Gu in AI4SM

Optimizing Food Delivery Routes with Ant Colony Optimization: A Solution for Thriving Delivery...

By Youngsup(Billy) Shin, Yunru(Ellen) Pan, and Yu(Ray) Gu as part of course project of ECE1724H: Bio-inspired Algorithms for Smart...

Oct 24, 2023 5

See all from Jingyuan Zhang

See all from AI4SM

Recommended from Medium

Amazon.com Seattle, WA
Software Development Engineer Mar. 2020 – May 2021

- Developed Amazon checkout and payment services to handle traffic of 10 Million daily global transactions
- Integrated Iframes for credit cards and bank accounts to secure 80% of all consumer traffic and prevent CSRF, cross-site scripting, and cookie-jacking
- Led Your Transactions implementation for JavaScript front-end framework to showcase consumer transactions and reduce call center costs by $25 Million
- Recovered Saudi Arabia checkout failure impacting 4000+ customers due to incorrect GET form redirection

Projects

NinjaPrep.io (React)
- Platform to offer coding problem practice with built in code editor and written + video solutions in React
- Utilized Nginx to reverse proxy IP address on Digital Ocean hosts
- Developed using Styled-Components for 95% CSS styling to ensure proper CSS scoping
- Implemented Docker with Seccomp to safely run user submitted code with < 2.2s runtime

HeatMap (JavaScript)
- Visualized Google Takeout location data of location history using Google Maps API and Google Maps heatmap code with React
- Included local file system storage to reliably handle 5mb of location history data
- Implemented Express to include routing between pages and jQuery to parse Google Map and implement heatmap overlay

Alexander Nguyen in Level Up Coding

The resume that got a software engineer a $300,000 job at Google.

1-page. Well-formatted.

Jun 1 22K 432

Nick Hagar in Generative AI in the Newsroom

Do People Want to Get Their News from Chatbots?

What data from real-world LLM interactions tells us

Sep 12 5

Lists

Staff Picks

736 stories · 1317 saves

Stories to Help You Level-Up at Work

19 stories · 804 saves

Self-Improvement 101

20 stories · 2764 saves

Productivity 101

20 stories · 2365 saves

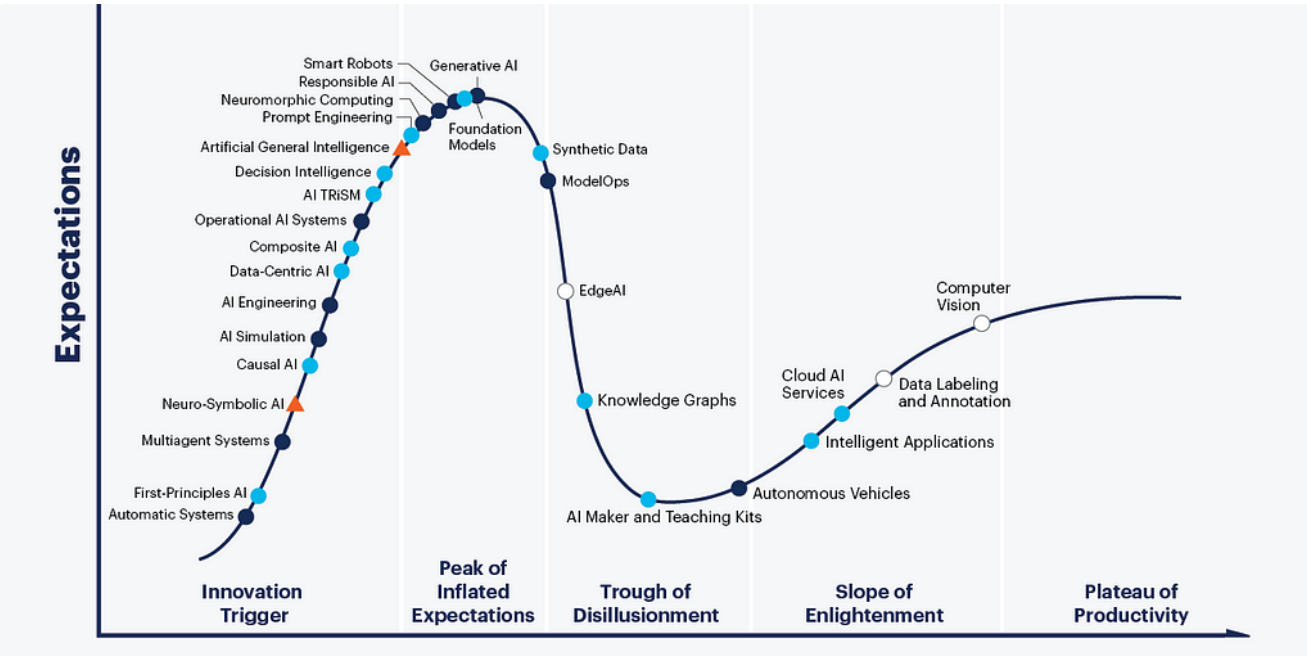
Use Case Families	Generative Models	Non-Generative ML	Optimisation	Simulation	Rules	Graphs
Forecasting	Low	High	Low	High	Medium	Low
Planning	Low	Low	High	Medium	Medium	High
Decision Intelligence	Low	Medium	High	High	High	Medium
Autonomous System	Low	Medium	High	Medium	Medium	Low
Segmentation	Medium	High	Low	Low	High	High
Recommender	Medium	High	Medium	Low	Medium	High
Perception	Medium	High	Low	Low	Low	Low
Intelligent Automation	Medium	High	Low	Low	High	Medium
Anomaly Detection	Medium	High	Low	Medium	Medium	High
Content Generation	High	Low	Low	High	Low	Low
Chatbots	High	High	Low	Low	Medium	High

Christopher Tao in Towards AI

Do Not Use LLM or Generative AI For These Use Cases

Choose correct AI techniques for the right use case families

Aug 10 3.4K 36

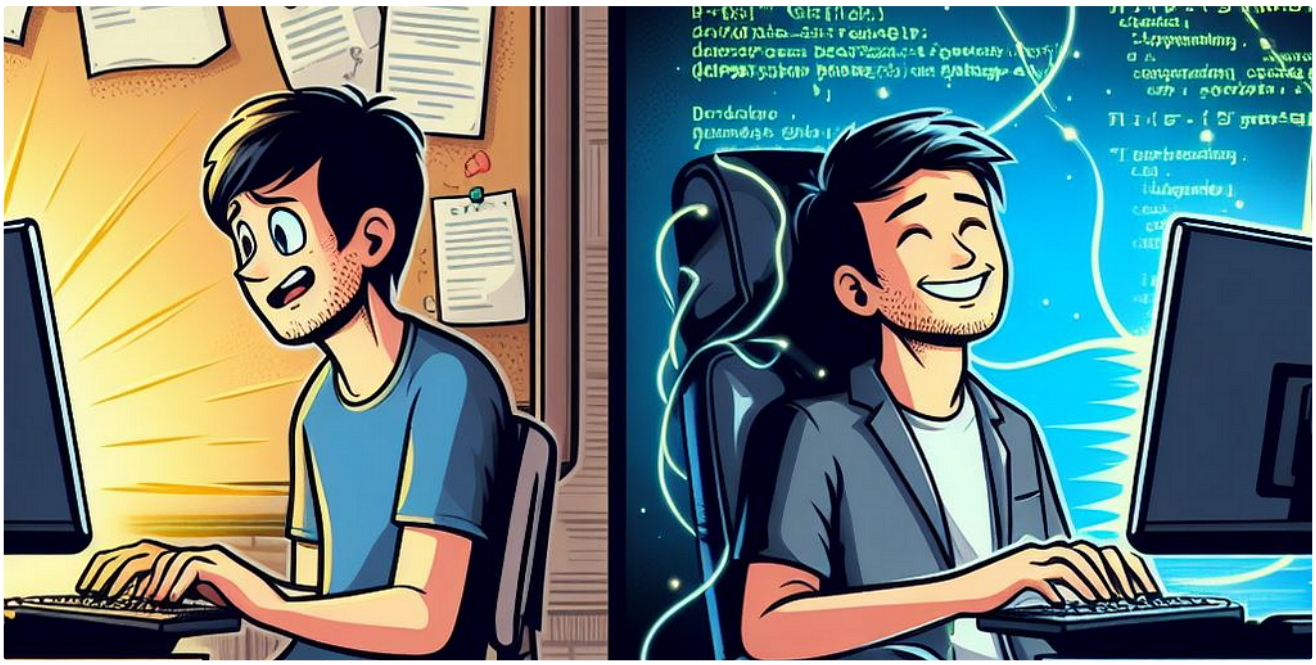


Vishal Rajput in AIGuys

Why GEN AI Boom Is Fading And What's Next?

Every technology has its hype and cool down period.

Sep 4 1.4K 46

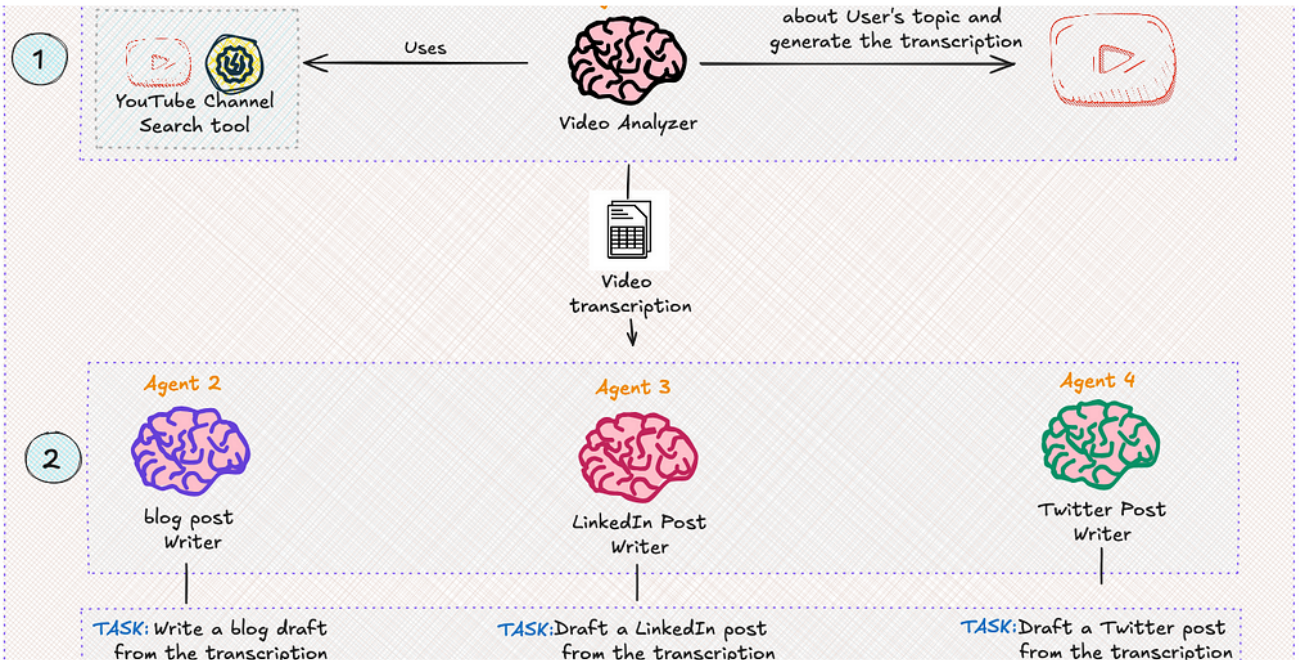


Abhay Parashar in The Pythoneers

17 Mindblowing Python Automation Scripts I Use Everyday

Scripts That Increased My Productivity and Performance

Aug 25 7.9K 75



Zoumana Keita in Towards Data Science

AI Agents — From Concepts to Practical Implementation in Python

This will change the way you think about AI and its capabilities

Aug 12 1.3K 19