Collaborative Embodied Reasoning in Autonomous Driving

Anonymous authors
Paper under double-blind review

Abstract

Deep reinforcement learning (DRL) is becoming an increasingly common technique to train agents to accomplish autonomous driving tasks (Kiran et al., 2021). However, DRL models, trained end-to-end, lack internal reasoning and planning for complex states and large action spaces (Dasgupta et al., 2023). Large language models (LLMs) could provide reasoning for autonomous vehicle systems and have demonstrate high-level reasoning across various tasks such as text generation (Wei et al., 2023), visual question and answer (VQA) (Liu et al., 2023a), and image generation (Huang et al., 2022). We investigate how to integrate cognitive reasoning from LLMs into autonomous vehicles that are trained with DRL. In this paper, we adapt the Planner-Actor-Reporter framework for autonomous driving tasks in Highway-Env and CARLA (Dasgupta et al., 2023; Leurent, 2018; Dosovitskiy et al., 2017). The work from Dasgupta et al. (2023) introduce the **Planner-Actor-Reporter** framework and apply it in gridworld reinforcement learning environments. We extend on their research by applying their framework to two autonomous driving environments, leveraging LLaVA as a visual reporter, and demonstrating common-sense driving reasoning using GPT-3.5-Turbo as the planner (Liu et al., 2023a). This paper opens new possibilities for safe, reliable, and interpretable decision making for autonomous vehicles by leveraging LLMs.

1 Introduction

LLMs have become widespread across many domains such as medicine (Thirunavukarasu et al., 2023), education (Moore et al., 2023), and autonomous driving (Fu et al., 2023). Their ability to demonstrate high-level reasoning in text generation (Brown et al., 2020) can be a promising tool to leverage in robotics, where an agent accomplishes tasks in an environment. For example, the work in Huang et al. (2022) demonstrates a pre-trained LLM as a planner giving a sequence of instructions to a robotic arm for table-top tasks. Frameworks presented in the research of Dasgupta et al. (2023) and Huang et al. (2022) employ a grounded-closed feedback loop between a LLM and a pre-trained agent. Since these methods use a pre-trained reinforcement learning (RL) agent, the agent was trained to maximize rewards. However, in tasks such as autonomous driving, the safety of the passenger should be the first priority in addition to cumulative rewards. Often times agents will learn "short-cuts" or non-human-like behavior that exploit the environment to maximize reward. Such "short-cuts" may not reflect actual human behavior, where an optimizal action is not necessarily the safest in driving.

The work from Dasgupta et al. (2023), propose a **Planner-Actor-Reporter** paradigm for their LLM and actor feedback system. Their work features a **Planner**, which serves as the pre-trained LLM, providing logical reasoning to breakdown statements into finer subtasks for an **Actor**; they, use Chinchilla's 7 Billion and 70 Billion parameter architectures (Hoffmann et al., 2022). The **Actor** is a DRL agent that carries out actions in a grid-world environment. The agent takes in images as its input and returns actions. Lastly, a **Reporter** translates the Actor's action and state space into the **Planner**'s modality. In work shown in Dasgupta et al. (2023), they showcase a hard-coded

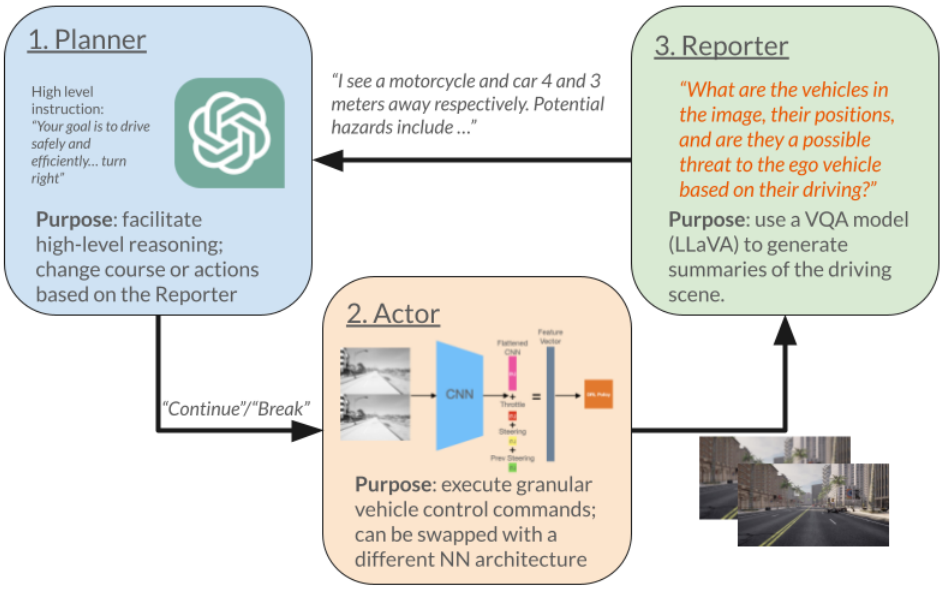

Figure 1: Planner-Actor-Reporter framework adapted to autonomous driving.

reporter to translate the actor's state and action space into the **Planner's** text space Dasgupta et al. (2023). However, this framework is limited to gridworld environments.

DRL has also shown success across a wide variety of domains such as Atari video games (Mnih et al., 2015), autonomous driving (Wang et al., 2022), and robotics (Schulman et al., 2015). However, many DRL implementations are brittle, sensitive to hyper-parameter changes, and require well-defined environments, making them challenging to generalize. Logical thinking patterns that LLMs exhibit have drawn researchers to incorporate them in embodied tasks, where an agent has motion capabilities in an environment; current work seeks to build feedback systems that tie in robotics elements with LLMs (Huang et al., 2022; Dasgupta et al., 2023). Currently, two aforementioned approaches applied their methods in simulated robotic arm or gridworld tasks. We look to expand emobdied reasoning and DRL to autonomous driving.

Current end-to-end approaches for autonomous driving tasks are efficient and can handle multi-modal inputs by utilizing a deep neural network (Kiran et al., 2021). However, the main challenges with end-to-end learning in autonomous driving include a lack of interpretability, generalizability, and multi-task learning (Chen et al., 2023). Thus, we propose a hybrid end-to-end and modular pipeline to add cognitive reasoning on top of end-to-end DRL decision-making by adapting the **Planner-Actor-Reporter** framework. In this framework, an end-to-end DRL model serves as the **Actor**, making granular motor control; the **Reporter** is either a hard-coded summarizer or VQA model - in this case LLaVA; the **Planner** is ChatGPT-3.5-Turbo who's role is to issue high-level commands to the actor. We apply the **Planner-Actor-Reporter** framework on a suite of driving tasks such as lane following and obstacle avoidance.

With this framework adaption, we present the following three contributions:

1. Demonstrate ChatGPT's commonsense reasoning capabilities within Highway-Environment.
2. Provide a pipeline to collect data and fine-tune LLaVA to generate driving reports of a driving scene for both Highway-Environment and CARLA.
3. Adapt the Planner-Actor-Reporter framework in CARLA.

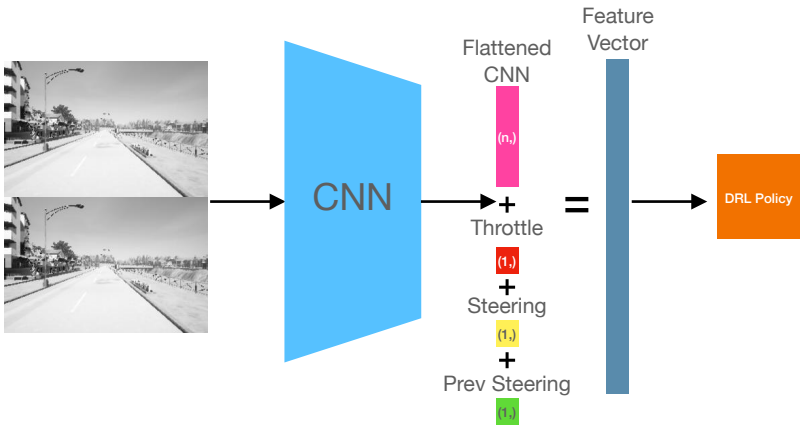

Figure 2: DRL end-to-end pipeline for CARLA

2 Methods

The methods section is divided into three parts: **Actor**, **Planner**, and **Reporter**. Figure 1 illustrates the entire framework adapted to autonomous driving tasks. The entire pipeline begins with a pre-trained DRL actor on some driving task. The **Reporter** gets initial state information from the actor. Here, the **Reporter** transforms the state space into text. This is the generated reporter, which is sent to the **Planner** to interpret. Finally, using OpenAI's API Liu et al. (2023b), we configure ChatGPT-3.5-Turbo to reason in the context of a human driver.

2.1 Actor

The Actor component of this framework adaptation serves two purposes: (i) give fine-control commands to the ego vehicle; and (ii) send images and actions to the reporter. For both driving environments, we train the agents using state-of-the-art DRL algorithms in an end-to-end manner. We demonstrate this framework in Highway-Env (Leurent, 2018) and CARLA (Dosovitskiy et al., 2017).

Highway-Env is an autonomous vehicle simulator that facilitates on-the-road decision-making scenarios; it is a 2D environment that provides kinematic and spatial data for each actor in the environment. Thus, the state space is a vector of physics-focused information. The action space is either continuous or discrete driving commands. Highway-Env has an accessible install process and requires few dependencies, which made it an ideal environment where we could measure ChatGPT-3.5's reasoning abilities. Given the simpler state and action space, we train DQN (Mnih et al., 2015) and A2C (Mnih et al., 2016).

Furthermore, we apply our method to CARLA, an autonomous vehicle simulator (Dosovitskiy et al., 2017). In addition to driving decision-making scenarios, CARLA is a 3D-rendered environment, so it provides a wide selection of sensors, modalities, and weather conditions Dosovitskiy et al. (2017). We implement a simple reinforcement learning environment where a car has two tasks: (i) follow a car lane; and (ii) avoid a direct obstacle. We use *Town01* as our default training map. We also follow an environment implementation for multi-agent reinforcement learning but demonstrate single-agent learning (Palanisamy, 2019).

We train two DRL algorithms in the CARLA simulation environment in an end-to-end manner with Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Soft-Actor-Critic (SAC)

(Haarnoja et al., 2018). The observation space comprises a 2-frame stacked grayscale image from an RGB[1] front-facing camera, represented as a three-dimensional tensor of size $(2, 160, 168)$, which is concatenated with the throttle value, steering value , and the previous steering value. The action space consists of continuous throttle and steering commands, with each ranging from [-1, 1]. For both algorithms, the neural network architecture utilizes a three-layer convolutional neural network (CNN) as the backbone. The layers contain 500, 300, and 100 units, respectively, designed to process the input data and output the control commands.

Figure 2 illustrates our DRL model for CARLA. The figure showcases a sample observation that is being passed into the CNN. Afterward, the flattened CNN is concatenated with a 1-single dimension scalar of the ego vehicle's throttle, steering, and previous steering command. Frame stacking and adding the previous steering serve two purposes: (i) mitigate catastrophic forgetting and (ii) reduce jittering in steering maneuvers.

2.2 Planner

The planner is designed to take in prompts from the user and the environment inference from the reporter and produce an appropriate action for the actor, which is the reinforcement learning agent. We use pre-trained large language models as the planner. In this case, we have used GPT-3.5-turbo as our planner, utilizing the OpenAI's API. We have modeled the API output to only give the appropriate safe action for our vehicle.

We start by giving the following prompt to the Planner, 'Your goal is to drive safely and efficiently. You are directing the ego vehicle in our simulation, selecting actions when prompted. Respond with the action name only and nothing else.' The inference generated by the reporter is then appended to this prompt during every sequence of tasks and used as the new prompt to get a new instruction from the Planner. The output from the Planner is the following action that the vehicle must take to complete a task safely and prevent any hazards. Given the user prompt mentioned above and appending the reporter inference, the planner can make a decision from the available safe actions for the ego vehicle.

2.3 Reporter

The modular characteristic of the **Planner-Actor-Reporter** framework allows for multiple interpretations for the **Reporter** (Dasgupta et al., 2023). In particular, we implement two types of **Reporters**: hard-coded and VQA.

2.3.1 Hard-coded reporter

In Highway-Env, **Actor** states are represented as numeric or categorical feature vectors. Key featuers we used were (x, y) coodinates, vx (x velocity), and vy (y velocity) (Leurent, 2018). This is convenient in that we could generate our own reporter by filling in a pre-made script with the state of the environment. Key information the **Actor** sent to the **Reporter** to synthesize feature geometric information of other vehicles (such as Euclidean distance in relation to the ego vehicle), speed, and steering angle.

Given that safety should be the utmost priority for all actors in a driving scene, we also use the geometric information to label driving scenarios as hazardous or not. For example, if a vehicle was within 5 meters from the ego vehicle, we would have the **Reporter** flag this scenario of *hazardous* so that the **Planner** can get a realistic description from the point of view of a human driver.

2.3.2 LLaVA

A hard-coded **Reporter** is ideal in simple and deterministic environments where state information is easily accessible: enter Highway-Env and Gridworld. This is because it is quick to summarize

[1] red-green-blue

state spaces in a pre-built script. However, realistic environments with oracle-level knowledge of the state are far and few in between. Thus, we turn to VQA models to further robustify the **Planner-Actor-Reporter** framework in a modular fashion. We make the claim that VQA models can exhibit greater generalizability in summarizing complex, image-based, and partial observations such as those seen in autonomous vehicle environments.

LLaVA is a multi-model model that combines a vision encoder, namely CLIP, and LLM (LLaMA) for VQA (Liu et al., 2023a). LLaVA was pre-trained from an ensemble of vision-related datasets such as COCO, GQA, and TextVQA Liu et al. (2023a). The pre-trained data consists of real-life everyday objects with varying quality. As such, we took two approaches with LLaVA: fine-tuning and zero-shot.

Given that CARLA is a 3D-rendered simulator and LLaVA is trained on real-life images, there is a reality gap. LLaVA's zero-shot demonstrations were primarily conducted on real-life images Liu et al. (2023a). This discrepancy in domains motivated us to fine-tune LLaVA on CARLA images.

To collect the custom dataset's images, we attached an RGB camera, obstacle detector, lane invasion detector, collision detector, and semantic segmentation camera to an ego vehicle. We simulated `Town10` and generated 50 NPC vehicles and 30 walkers. After which, the ego vehicle is spawned with `autopilot=True`; this made our ego vehicle roam around the map adhering to basic driving rules. At each simulated timestep, we record the RGB camera image, gather object labels from the semantic segmentation camera, and any immediate obstacles, collisions, or lane invasions. All of this information then is written to a JSON format to fine-tune LLaVA. This custom dataset comprised of the images and QA labels, which in this case were labeled objects in the driving scene. We ran the data-collector for 20,000 simulation ticks, thus garnering 20,000 data samples.

```
"from":  "human",
"value":  "<image>What are objects worth noting in the current
scenario?"

"from":  "gpt",
"value":  "Road, Sidewalk, Building, Pole, Traffic sign, Car,
Static, Roadline"
```

Figure 3: This is the caption for the quoted text.

Figure 3 showcases captions for our custom CARLA fine-tune dataset. Unforeseen hazard detection is an important goal to achieve in safe driving. Therefore, we design the captions to label objects in CARLA and follow up the conservation by asking about immediate hazards within the ego vehicle's proximity. While it is feasible to have LLaVA's language model also serve as the **Planner**, we decided to keep the **Planner** and **Reporter** as distinct and separate models to avoid hallucinations induced by long querying and subjecting the model to multiple contexts (Xu et al., 2024). Hallucinations can happen in the inference phase by lacking attention-context (Xu et al., 2024).

Figure 4 illustrates the data distribution of our custom CARLA dataset. Each wedge of the pie denotes the total number of pixels seen for that semantic label. Roadlines, buildings, sidewalks, cars, and traffic make up the majority of objects seen throughout the dataset. While 'static' and 'other' are ambiguous, they remain in the dataset since the focus of the **Reporter** is to summarize potential hazards from other vehicles.

With our custom dataset, we fine-tune LLaVA. On two GeForce RTX-3090 GPUs, we leveraged LORA and deepspeed to fine-tune the model with fewer parameters without sacrificing performance. Hyperparameters include the learning rate $= 2 \times 10^{-4}$, epochs $= 2$, and baseline model set to the seven billion parameter version of LLaVA (Liu et al., 2023a).

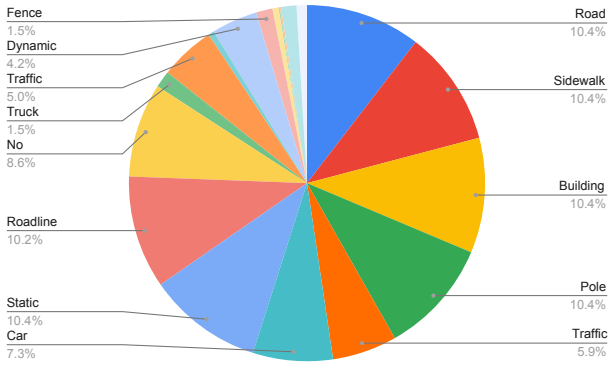

Figure 4: Data distribution of our custom CARLA dataset for LLaVA fine-tuning.

3 Results

In Highway-Env, we apply our DRL pipeline with the hard-coded **Reporter** to tackle four different tasks. The first task is the standard highway-v0 Highway-Env gym environment, which can be described as driving on a 4-lane highway with standard traffic conditions. In order to test the robustness of our framework against long-tail safety cases, we built three other custom environments, making variations of the highway-v0 environment to simulate cases that current autonomous vehicles may not be optimized to handle. Such cases include the WrongWay environment, the AnimalCrossing environment, and the StoppedCar environment. Each of these environments utilizes the same lane network as highway-v0, however they are modified to produce different vehicle and object behavior. WrongWay spawns cars driving in the opposite direction of traffic at a configurable probability, AnimalCrossing spawns objects traveling perpendicular to the flow of traffic at a configurable probability, and StoppedCar spawns vehicles that are moving at 0 velocity, or in other words, stopped on the highway. We select these three custom environments to create a comprehensive testing framework for evaluating the LLM's capability to reason like a human in driving scenarios. We test our embodied reasoning model against the DQN and A2C reinforcement learning algorithms on zero-shot pass-rate for driving tasks. The pass-rate is represented by the percentage of each agent successfully navigating the environment such that at least one adversarial obstacle or collision is avoided. As shown in Figure 5, we see superior performance of the embodied reasoning

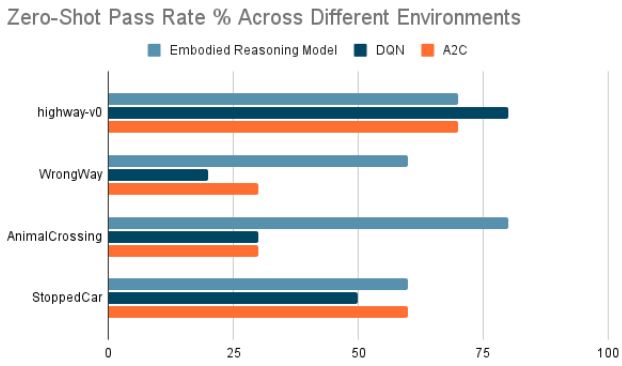

Figure 5: Zero-shot Pass Rate % Across Different Environments.

agent utilizing the hard-coded **Reporter** to navigate environments. The most notable performance increases can be seen in the AnimalCrossing and WrongWay environments, with moderate perfor-

mance increase in the StoppedCar environment compared to the DQN agent and no performance improvements compared to the DQN and A2C agents in the original highway-v0 environments.

We also implement a similar framework to collect baseline data and test our embodied reasoning agent. Drive Like a Human utilizes a similar embodied reasoning framework, with differences in how helper functions are called and separate threat-checking functions for their LLM to call before selecting an action (Fu et al., 2023). Our embodied reasoning model achieves similar performance to this baseline, with marginal improvements in the WrongWay custom long-tail environment. Figure 6 shows our approach compared against Drive Like a Human in the adversarial Highway environments. Over 100 trials, we measure the success rate of each embodied reasoning system in three tasks:

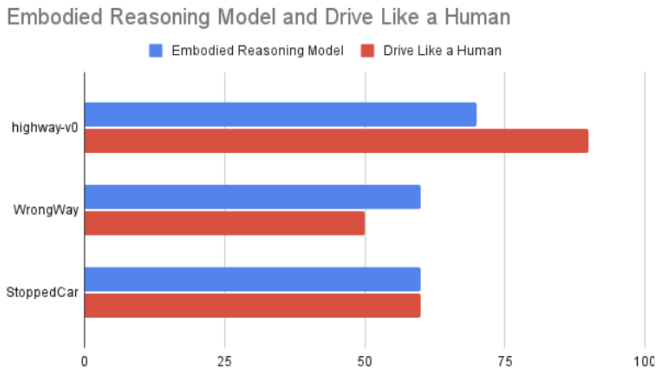

Figure 6: Drive Like a Human Baseline

highway-v0, WrongWay, and StoppedCar.

3.1 CARLA

In CARLA, we apply our DRL pipeline to one task: lane follow and collision avoidance. First, we demonstrate the DRL baselines for PPO and SAC. The maximum reward is 200 units. Figure 7 shows baseline results for the lane following task. A noteworthy observation is how SAC outperforms PPO. We speculate that the lack of multiprocessing throughout training and SAC's sample efficiency lends were reasons why PPO's results pale in comparison to SAC. After training the baselines, we chose

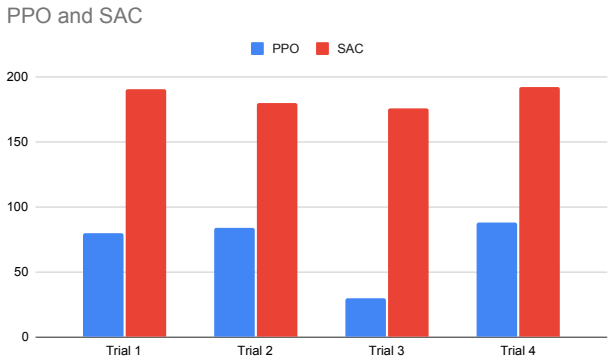

Figure 7: Baseline RL performance for lane-following in CARLA.

the best-performing PPO model to serve as the **Actor** in the collaborative reasoning framework.

When applying the **Planner-Actor-Reporter** framework with a PPO model, we used a more challenging environment. This harder environment features an NPC vehicle spawning in front of the

ego vehicle. We hypothesize that under this scenario, the LLaVA model would determine that the NPC vehicle is close by. Afterwards, the Planner would suggest the ego vehicle to brake.

We conduct four experiments across 10 trials: lane-follow with zero-shot and fine-tuned LLaVA and obstacle braking with zero-shot and fine-tuned LLaVA. For each obstacle brake experiment, we restarted the environment with a different vehicle to measure generalizability in addition to common-sense reasoning. In lane-follow, we spawned the agent on a pre-defined point for each trial. Both lane-follow and obstacle braking in CARLA were conducted on `Town01`.

LLaVA Model	Lane Follow	Obstacle Braking
Zero-shot	70%	50%
Fine-tuned	80%	20%

Table 1: Planner-Actor-Reporter CARLA task accuracy.

Table 1 shows the results for our framework applied in CARLA tasks. Specifically, the table exhibits the accuracy the **Planner** and **Reporter** were able to collaborate and determine the correct high-level command for the **Actor**. Overall, lane-follow was an easier task for the framework to issue correct commands. Conversely, obstacle braking caused the framework to perform worse. We speculate that the lack of motion information from the **Reporter's** summarization made it challenging for the **Planner** to issue the appropriate high-level command. Furthermore, the obstacle results from Table 1 could suggest our fine-tuned LLaVA model is overfitting the data. The fine-tuned model exhibiting lower accuracy obstacle braking as opposed to the zero-shot model supports the claim that overfitting is present.

4 Discussion and Conclusion

In adapting the **Planner-Actor-Reporter** framework, we hope to instill discussion on how LLMs can be safety integrated to autonomous driving. Both fine-tuned and zero-shot approaches with LLaVA had an above 60% success rate and generated reports that induced the right command from the planner. However, we see a remarkable decrease in accuracy when braking before an obstacle for both zero-shot and fine-tuning approaches. We accredit this behavior to the lack of visual motion information provided by the pair of image inputs.

Future work could involve passing in longer sequences of RGB image frames throughout the task. Longer sequences could potentially help the LLaVA model determine the movement of objects in a driving scenario. Another idea to mitigate this shortcoming is to incorporate some attention-based architecture in the **Actor** or an asynchronous framework.

Another direction for future work is incorporating more modalities in the VQA **Reporter**. In our work, we demonstrate LLaVA, which can take images. However, a more comprehensive survey and benchmark of other models is warranted. Other multi-modal VQA models that are worth exploring include LLaMA-Adapter 2 (Zhang et al., 2023), ImageBind (Han et al., 2023) (for point cloud data which could be useful for LiDAR information), and NuScenesQA (Qian et al., 2024), which is an autonomous driving and multi-modal specific VQA model.

Lastly, a thorough safety exam with unseen scenarios is warranted. For CARLA, we only examined lane following and obstacle braking. While we used different vehicles for obstacles, vehicles do not make up all entities on the road. Further examination with cyclists and pedestrian is called for.

We present an end-to-end and modular hybrid approach that incorporates LLMs as embodied reasoners. We hope that this framework can spark discussion, innovation, and safety in combining generative artificial intelligence with autonomous driving.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers, 2023.

Ishita Dasgupta, Christine Kaeser-Chen, Kenneth Marino, Arun Ahuja, Sheila Babayan, Felix Hill, and Rob Fergus. Collaborating with language models for embodied reasoning. *arXiv preprint arXiv:2302.00763*, 2023.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.

Daocheng Fu, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Yu Qiao. Drive like a human: Rethinking autonomous driving with large language models, 2023.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

Jiaming Han, Renrui Zhang, Wenqi Shao, Peng Gao, Peng Xu, Han Xiao, Kaipeng Zhang, Chris Liu, Song Wen, Ziyu Guo, Xudong Lu, Shuai Ren, Yafei Wen, Xiaoxin Chen, Xiangyu Yue, Hongsheng Li, and Yu Qiao. Imagebind-llm: Multi-modality instruction tuning, 2023.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *arXiv preprint arXiv:2207.05608*, 2022.

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey, 2021.

Edouard Leurent. An environment for autonomous driving decision-making. https://github.com/eleurent/highway-env, 2018.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023a.

Yiheng Liu, Tianle Han, Siyuan Ma, Jiayue Zhang, Yuanyuan Yang, Jiaming Tian, Hao He, Antong Li, Mengshen He, Zhengliang Liu, Zihao Wu, Lin Zhao, Dajiang Zhu, Xiang Li, Ning Qiang, Dingang Shen, Tianming Liu, and Bao Ge. Summary of chatgpt-related research and perspective towards the future of large language models. *Meta-Radiology*, 1(2):100017, September 2023b. ISSN 2950-1628. doi: 10.1016/j.metrad.2023.100017. URL http://dx.doi.org/10.1016/j.metrad.2023.100017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Steven Moore, Richard Tong, Anjali Singh, Zitao Liu, Xiangen Hu, Yu Lu, Joleen Liang, Chen Cao, Hassan Khosravi, Paul Denny, et al. Empowering education with llms-the next-gen interface and content generation. In *International Conference on Artificial Intelligence in Education*, pp. 32–37. Springer, 2023.

Praveen Palanisamy. Multi-agent connected autonomous driving using deep reinforcement learning, 2019.

Tianwen Qian, Jingjing Chen, Linhai Zhuo, Yang Jiao, and Yu-Gang Jiang. Nuscenes-qa: A multi-modal visual question answering benchmark for autonomous driving scenario, 2024.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8):1930–1940, 2023.

Tsun-Hsuan Wang, Alexander Amini, Wilko Schwarting, Igor Gilitschenski, Sertac Karaman, and Daniela Rus. Learning interactive driving policies via data-driven simulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 7745–7752. IEEE, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. Hallucination is inevitable: An innate limitation of large language models, 2024.

Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.