Language-Driven Policy Distillation for Cooperative Driving in Multi-Agent Reinforcement Learning

Jiaqi Liu, Chengkai Xu, Peng Hang, Jian Sun, Wei Zhan, Masayoshi Tomizuka, and Mingyu Ding

Abstract— The cooperative driving technology of Connected and Autonomous Vehicles (CAVs) is crucial for improving the efficiency and safety of transportation systems. Learning-based methods, such as Multi-Agent Reinforcement Learning (MARL), have demonstrated strong capabilities in cooperative decision-making tasks. However, existing MARL approaches still face challenges in terms of learning efficiency and performance. In recent years, Large Language Models (LLMs) have rapidly advanced and shown remarkable abilities in various sequential decision-making tasks. To enhance the learning capabilities of cooperative agents while ensuring decision-making efficiency and cost-effectiveness, we propose LDPD, a language-driven policy distillation method for guiding MARL exploration. In this framework, a teacher agent based on LLM trains smaller student agents to achieve cooperative decision-making through its own decision-making demonstrations. The teacher agent enhances the observation information of CAVs and utilizes LLMs to perform complex cooperative decision-making reasoning, which also leverages carefully designed decision-making tools to achieve expert-level decisions, providing high-quality teaching experiences. The student agent then refines the teacher's prior knowledge into its own model through gradient policy updates. The experiments demonstrate that the students can rapidly improve their capabilities with minimal guidance from the teacher and eventually surpass the teacher's performance. Extensive experiments show that our approach demonstrates better performance and learning efficiency compared to baseline methods.

I. INTRODUCTION

Recent advancements in Connected and Autonomous Vehicle (CAV) technology have enabled vehicles to communicate and collaborate, revolutionizing the transportation industry by enhancing overall traffic management [1], [2], [3]. Cooperative driving technology is a critical component of CAV deployment, which enables them to make efficient decisions in dynamic environments, thereby enhancing safety and traffic flow. However, despite the significant potential of cooperative decision-making technology, current methods still face numerous challenges in terms of scenario applicability, decision-making efficiency, and safety, making large-scale real-world deployment a distant goal [4].

The rapid development in deep learning has made learning-based approaches, such as Multi-Agent Reinforce-

ment Learning (MARL), effective for cooperative decision-making [5]. In MARL, each agent explores strategies to maximize its own reward, thereby identifying optimal decision-making policies of CAV, which has been employed by many researchers [6], [7], [8]. Despite the notable advantages of MARL, challenges remain, such as difficulties in algorithm convergence, low exploration efficiency, and high sampling costs [8]. On the other hand, recent advancements in Large Language Models (LLMs) have demonstrated impressive zero-shot generalization and complex logical reasoning capabilities across various decision-making tasks [9]. These capabilities are particularly well-suited for assessing complex situations in human traffic environments and achieving coordinated control between vehicles and traffic systems. A few studies have explored the integration of LLMs into traffic and vehicle systems to enhance the safety and efficiency of transportation networks [10], [11]. However, in real-time, dynamic traffic scenarios, existing LLMs struggle to enable CAVs to make rapid and efficient decisions. Additionally, the high deployment costs hinder large-scale implementation, significantly limiting the potential of LLMs [12].

To address these challenges, we propose a novel approach that leverages the reasoning efficiency of MARL-based agents alongside the extensive world knowledge of LLMs. This approach involves designing a teacher-student policy distillation framework to facilitate the efficient transfer of LLMs' world knowledge to MARL agents. we present LDPD, a language-driven policy distillation framework designed to guide MARL exploration. The framework features a teacher agent powered by LLM and multiple student agents (CAVs) modeled with MARL, enabling effective knowledge transfer and collaborative learning. During the training phase, the teacher agent receives observation information from students, leveraging its powerful zero-shot and reasoning capabilities to make expert-level decisions. It then guides the smaller student agents through its actions, enabling them to perform efficient sampling and policy learning. After the learning phase, these Student policies will function as the controllers for CAVs, making decisions independently of the teacher during deployment. For teacher agent, the Observation module preprocesses and enhances student observations, which are then sent to the LLM-based Planner for decision-making. The Planner integrates the data using agent tools, and expert-level outputs are generated. These expert decisions are used to train the student policies. The student agents are trained in a decentralized MARL framework, using the Actor-Critic method to update their action policies.

As shown in Fig.1, we use a ramp merging environ-

Jiaqi Liu and Mingyu Ding are with the Department of Computer Science at University of North Carolina at Chapel Hill, NC, USA.{jqliu,md}@cs.unc.edu

Chengkai Xu, Peng Hang and Jian Sun are with the College of Transportation and Key Laboratory of Road and Traffic Engineering, Ministry of Education, Tongji University, Shanghai 201804, China.

Masayoshi Tomizuka and Wei Zhan are with the Department of Mechanical Engineering at the University of California, Berkeley, CA, USA.{tomizuka,wzhan}@berkeley.edu
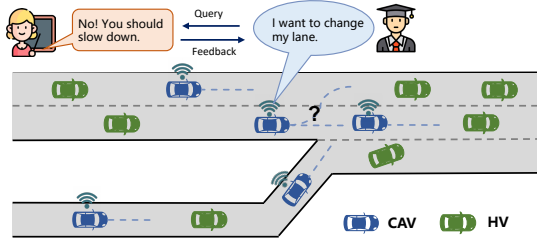
Corresponding author: Mingyu Ding

Fig. 1. Illustration of the multi-CAV decision-making task with LLM guidance at the merge scenario. Each CAV agent will make decision with the consideration of reasoning and advice from LLM and update policy with knowledge distilled by LLM.

ment to train and test our method. Our experiments show that a small amount of high-quality teaching from the teacher significantly improves student policies' exploration efficiency, leading to rapid performance gains that eventually surpass the teacher's. Extensive experiments demonstrate that our teaching system effectively enhances CAV learning, outperforming baseline methods in learning efficiency and performance.

Our contributions are summarized as follows:

- We model the cooperative decision-making problem of CAVs as a MARL problem and utilize LLM to guide the exploration of CAVs.
- We propose LDPD, a language-driven policy distillation framework to facilitate rapid learning and efficient exploration for smaller policy networks.
- Through the teacher's guidance, the performance of the Student agents demonstrates significant advantages. Our method outperforms baseline approaches in terms of both learning efficiency and overall performance.

II. RELATED WORKS

Cooperation Decision-Making of CAVs. In complex mixed-autonomy traffic environments, cooperative decision-making technologies can effectively improve the efficiency and safety of the CAV system. Numerous methods have been proposed to address vehicle coordination, including rule-based methods [13], optimized-based methods [14], and learning-based methods [15]. Among them, learning-based methods are highly effective in simulating vehicle interaction dynamics and reasoning, while their lack of interpretability and limited generalization capabilities hinder their application in safety-critical scenarios.

Large Language Model for Decision-making. Large Language Models (LLMs), known for their human-like reasoning and commonsense generalization, have recently demonstrated potential beyond traditional natural language processing tasks [16], [17]. In recent years, LLMs have been utilized as employed as end-to-end agents capable of autonomous decision-making. For cooperative driving tasks, Hu et al. proposed AgentsCoDriver [11], a LLM-driven life-long learning framework to empower CAV decision-making. Meanwhile, CoDrivingLLM, proposed by Fang et al. [10], is designed to adapt different levels of Cooperative Driving Automation, which could effectively address the dilemma

of cooperative scenarios. While LLMs have demonstrated remarkable potential, significant challenges remain. Their high computational demands and the complexity of real-time decision-making introduce latency issues, which limit their application in time-critical environments.

MARL and LLM-based MARL. Multi-Agent Reinforcement Learning (MARL) is an evolving research area that addresses systems composed of multiple interacting agents, such as robots, vehicles, and machines operating within a shared environment [7], [18], [8]. However, MARL approaches often face challenges in scalability, inter-agent coordination, and exploration when applied to more dynamic and uncertain real-world scenarios. To address these limitations, integrating Large Language Models (LLMs) into MARL frameworks has emerged as a promising direction. LLM-based MARL approaches aim to harness LLMs' reasoning and knowledge capabilities to improve agents' decision-making processes [19]. Recent studies have explored LLMs' potential to boost the learning efficiency and performance of Reinforcement Learning (RL) agents. For example, in [20], a semi-parametric RL agent was developed by integrating a Rememberer module that utilizes LLMs to retain long-term experiential memory. Similarly, [12] employed LLM-based instructions to train agents and distill prior knowledge, resulting in enhanced RL performance. Despite these advancements, LLMs' application in MARL remains underexplored. Current approaches have yet to fully capitalize on LLMs' ability to interpret environmental cues and negotiate high-level strategies and intentions among agents.

III. PROBLEM FORMULATION

The cooperative decision-making problem for multiple CAVs can be modeled as a partially observable Markov decision process (POMDP) [21]. We define the POMDP using the tuple $\mathcal{M}_{\mathcal{G}} = (\mathcal{V}, S, [\mathcal{O}_i], [\mathcal{A}_i], \mathcal{P}, [r_i])$, where $\mathcal{V}$ represents the finite set of all controlled agents (CAVs), and S denotes the state space encompassing all agents. $\mathcal{O}_i$ represents the observation space for each agent $i \in \mathcal{V}$, $\mathcal{A}_i$ denotes the action space, and r_i is the reward associated with CAV i. The transition distribution is represented by $\mathcal{P}$. At any given time, each agent i receives an individual observation $\mathcal{O}_i$ and selects an action $a_i \in \mathcal{A}_i$ based on a policy $\pi_i : \mathcal{O}_i \times \mathcal{A}_i \to [0, 1]$.

A. Observation Space

The observation matrix for agent i, denoted as $\mathcal{O}_i$, is a matrix with dimensions $|\mathcal{N}_i| \times |\mathcal{F}|$, where $|\mathcal{N}_i|$ represents the number of observable vehicles for agent i, and $|\mathcal{F}|$ is the number of features used to describe a vehicle's state. The feature vector for vehicle k is expressed as

$$\mathcal{F}_k = [x_k, y_k, v_k^x, v_k^y, \cos \phi_k, \sin \phi_k], \tag{1}$$

where x_k, y_k v_k^x and v_k^y are the longitudinal and lateral positions and speeds, respectively. ϕ_k is the vehicle's heading angle. The overall observation space of the system is the combined observation of all CAVs, i.e., $\mathcal{O} = \mathcal{O}_1 \times \mathcal{O}_2 \times \cdots \times \mathcal{O}_{|\mathcal{V}|}$.

B. Action Space

Given that the strength of LLMs lies in their reasoning capabilities based on world knowledge rather than numerical computation, we design the decision actions of CAVs as discrete semantic decisions rather than direct vehicle control actions. The action space $\mathcal{A}_i$ for agent i is defined as a set of high-level control decisions, including $\{slow\ down, cruise, speed\ up, change\ left, change\ right\}$. Once a high-level decision is selected, lower-level controllers generate the corresponding steering and throttle control signals to manage the CAVs' movements. The overall action space is the combination of actions from all CAVs, i.e., $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_{|\mathcal{V}|}$.

C. Reward function

In this paper, with the objective of ensuring that all agents pass through the merging area safely and efficiently, based on previous work [18],the reward of ith agent at the time step t is defined as follows:

$$r_{i,t} = \omega_c r_c + \omega_s r_s + \omega_h r_h + \omega_m r_m. \qquad (2)$$

where $\omega_c, \omega_s, \omega_h$ and ω_m are positive weighting coefficients of collision reward r_c, stable-speed reward r_s, headway cost reward r_h and merging cost reward r_m, respectively.

IV. METHOD

In this section, the whole LDPD framework is first summarized. Then the teacher agent and student agent are introduced respectively.

A. Framework Overview

The LDPD framework is designed to facilitate the learning and exploration process of multi-agent systems with distilled knowledge from LLM, especially in complex cooperative driving scenarios. As shown in Fig.2, the LDPD framework comprises two main components: the teacher agent and the student agents. The teacher agent facilitates efficient policy learning for multiple student agents through its action demonstrations. The student agents, composed of multiple small policy networks, form an agent group where each network is responsible for controlling an individual CAV. Throughout this process, the student agents efficiently distill and apply the knowledge from the LLM, enabling rapid learning and adaptation.

B. Teacher Agent

The teacher agent is the core of our entire framework, comprising the Observation Enhancer, Planner, and Safety Checker.

1) Observation Enhancer

This module is primarily responsible for collecting perception information from the CAV and enhancing the state information. The teacher first receives the observation $\mathcal{O}$ from the Student agents and then reorganizes this information.

Specifically, we categorize the lanes into three types: Lanes = {lane_ego, lane_adj, lane_conf}, where lane_ego, lane_adj, and lane_conf represent the ego lane, adjacent lanes, and conflict lanes, respectively. The ego lane is the lane in which the vehicle is currently driving, the adjacent lanes are those on either side of the ego lane (if they exist), and the conflict lanes are those intersecting with the ego lane. Similarly, based on the relationship between the vehicle's lane and the ego lane, we classify vehicle information into front vehicle (Veh_f), rear vehicle (Veh_r), surrounding vehicles (Veh_s), and conflict vehicles (Veh_c). From these classifications, we can summarize the surrounding environment information for each CAV.

Additionally, based on the state information of the CAV, we predict and infer the driving intentions of each CAV, including the desired target lane and intended driving behavior. At time t, the driving intention information for CAV i is represented as $In_{i,t} = \{lane'_{i,t}, behavior_{i,t}\}$, where $lane'_{i,t}$ and $behavior_{i,t}$ refer to the desired lane intention and the desired behavior intention, respectively.

Using the state and intention information, we generate a semantic driving scenario description for each CAV:

$$Sce_i = Enhancer\{O_i, Veh_i, Lanes_i, In_i\}. \qquad (3)$$

2) Core Planner

The LLM-based Planner is the core of the teacher agent, integrating reasoning and decision-making actions within the LLM using the ReAct method [22]. We have designed a series of Agent tools to assist the LLM in making expert-level teacher decisions. These tools include lane querying, vehicle state prediction, conflict checking, conflict resolution, and more.

We use the conflict checking function as an example to introduce our tools: After the LLM obtains the semantic description of the scenario, it will call the conflict checking tool $conflict\_check()$ before initiating reasoning to detect potential safety risks.

$$Risk = LLM\{conflict\_check(Sce_i)\} \qquad (4)$$

The conflict checking tool first retrieves all potential conflicts in the current environment and assesses the risk level based on the current states of the conflicting parties. Here, we utilize the time-to-conflict-point (TTCP) difference [10] as a surrogate metric for the risk level, which is widely used to evaluate collision risks at potential conflict points between vehicles, calculated as follows:

$$\Delta TTCP = \left| TTCP_i - TTCP_j \right| = \left| \frac{d_i}{v_i} - \frac{d_j}{v_j} \right| \qquad (5)$$

where $TTCP$ is the time to conflict point based on the vehicle's current distance to the conflict point d and speed v.

By invoking the conflict checking module, we analyze the conflict information in the ramp merging scenario and convert this information into a semantic description that is provided to the LLM. Based on the scenario and conflict information, the LLM generates an initial decision:

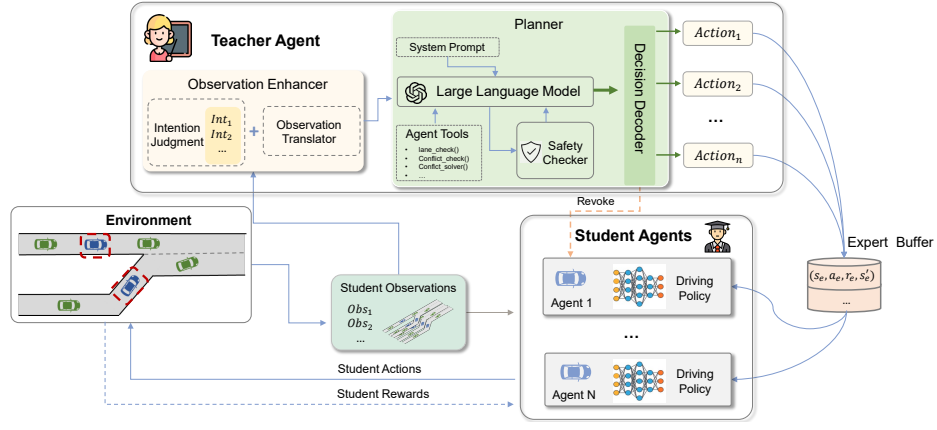$$A = LLM(Sce, conflict\_info) \qquad (6)$$

Fig. 2. Illustration of the proposed LDPD framework, consisting of two components: a teacher agent powered by LLM and student agents modeled by MARL. The teacher agent processes observations from the student agents and, using LLM-based planning, outputs expert-level action recommendations. In the early exploration stage, the student agents accelerate the learning and updating of their policy networks by distilling knowledge from the teacher's demonstrations.

3) Safety Checker

After the initial decision action set is generated, it enters a Safety Checker for further safety verification to ensure the accuracy of the teacher's decision results. The Safety Checker first re-evaluates conflicts and intentions based on the previously predicted intentions and the LLM's decision output. It begins by acquiring the intention trajectories of all CAVs and their surrounding vehicles for the next T_n steps.

Firstly, the formula is designed to calculate the priority score of each CAV [18]:

$$p_i = \alpha_1 p_m + \alpha_2 p_e + \alpha_3 p_h + \sigma_i. \quad (7)$$

where p_i denotes the priority of agent i, α_1, α_2, and α_3 are the positive weighting parameters for the merging metric p_m, merging-end metric p_e, and time headway metric p_h, respectively. A random variable $\sigma_i \sim \mathcal{N}(0, 0.001)$ is added as a small noise to avoid the issue of identical priority values.

Specifically, p_m is defined as:

$$p_m = \begin{cases} 0.5, & \text{If on merge lane,} \\ 0, & \text{Otherwise.} \end{cases} \quad (8)$$

which indicates that vehicles on the merge lane should be prioritized over those on the through lane due to the urgency of the merging task.

Next, since merging vehicles closer to the end of the merge lane should be given higher priority due to a greater risk of collision and deadlocks, the merge-end priority value is calculated as follows:

$$p_e = \begin{cases} \frac{x}{L}, & \text{If on merge lane,} \\ 0, & \text{Otherwise.} \end{cases} \quad (9)$$

where L is the total length of the merging lane, and x is the position of the ego vehicle on the ramp.

Finally, we define the time-headway priority as:

$$p_h = -\log\left(\frac{d_{headway}}{t_h v_t}\right) \quad (10)$$

indicating that vehicles with smaller time-headway are more dangerous than others.

With the priority scores of CAVs calculated above, at time step t, the safety checker first produces a priority list P_t, which consists of a list of priority scores and their corresponding ego vehicles. Then, the safety checker sequentially examines the future trajectory of the vehicles in the list to determine if there are any conflicts between vehicles. We use the Intelligent Driver Model (IDM) [23] to predict the longitudinal acceleration of HVs, based on their current speed and distance headway, and we utilize the MOBIL lane change model [24] to predict the lateral behavior of HVs.

If a collision is detected, the Safety Checker module generates a new safety decision to correct the actions of the Planner. The correction process is as follows:

$$a_t' = \arg \max_{a_t \in A_{available}} \left(\min_{k \in T_n} d_{sm,k}\right). \quad (11)$$

where $A_{available}$ is a set of available actions at time step k for the selected agent, and $d_{sm,k}$ is the safety margin at prediction time step k. The safety margin is defined as follows:

$$d_{sm,k} = \begin{cases} \min\left\{|P_{v_t,k} - P_{v_e,k}|, |P_{v_c,k} - P_{v_e,k}|\right\}, & \text{If change lane,} \\ P_{v_{pre},k} - P_{v_e,k}, & \text{Otherwise.} \end{cases} \quad (12)$$

where $P_{v_t,k}$ and $P_{v_c,k}$ denote the longitudinal positions of the preceding and following vehicles relative to the ego vehicle on both the target and current lanes, respectively. $P_{v_{pre},k}$ represents the position of the preceding vehicle at time step k, and $P_{v_e,k}$ is the position of the ego vehicle.

Finally, the teacher agent decodes and outputs the decision results, which include the set of actions guiding all CAVs: $A^* = \{a_1^*, a_2^*, \ldots, a_n^*\}$, where

$$A^* = SafetyChecker(A). \quad (13)$$

After the entire decision-making process of the teacher is completed, the decision results are stored in the Expert Buffer $\mathcal{D}$ for subsequent training of the Student Agents.
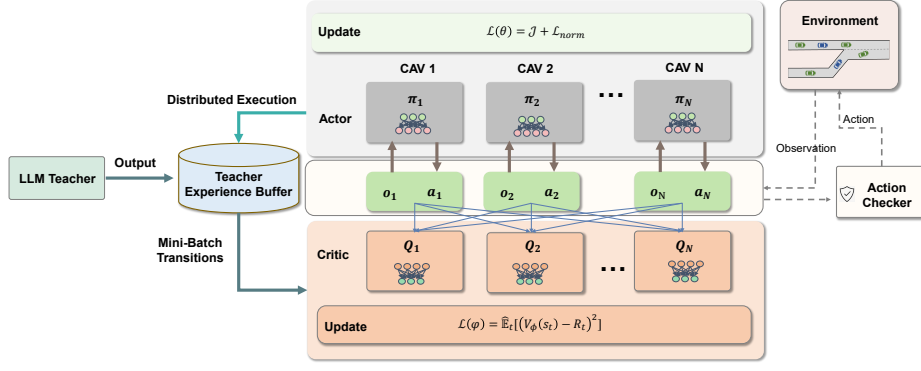
Fig. 3. Illustration of policy updating and optimal action generation by student agents. The policy gradients of all student agents are updated using the actor-critic method. Expert-level demonstrations from the teacher agent are stored in the experience buffer. During the early exploration stage, the teacher's demonstrations are sampled and used to update the students' policies. Afterward, the student agents proceed with independent exploration.

Algorithm 1: Teacher Agent

Inputs : Student observations $\mathcal{O}^t$, Student list $\mathcal{C}$
Output: Decisions set of teacher $\mathcal{A}_{Te}^{t+1}$

---

1 Initialize scenario description buffer $\boldsymbol{Sce}$;
2 **for** $i \in \mathcal{C}$ **do**
3   Generate the scenario description sce_i of student i based on current states $\mathcal{S}_i^t$;
4   Add sce_i to buffer $\boldsymbol{Sce} \leftarrow \boldsymbol{Sce}.\text{add}(sce_i)$;

5 **for** $i \in \mathcal{C}$ **do**
6   Obtain the action a_i^{t+1} of LLM by (6);
7   Check the safety of action a_i^{t+1} and solve the conflict with Safety Checker;
8   Obtain the optimal action a_i^{*t+1};
9   Add decision to buffer $\mathcal{A}_{Te}^{t+1} \leftarrow \mathcal{A}_{Te}^{t+1}.\text{add}(a_i^{*t+1})$;

C. Student Agents

The Student Agents are composed of multiple small policy networks that form an agent group. In our scenario, each agent represents a single CAV, with its policy network functioning as the CAV's controller. We model these Student Agents using the Actor-Critic framework and train them under a CTDE approach. This means that each CAV consists of two models: the actor and the critic. Each actor undergoes centralized training via a centralized critic while making distributed, independent action decisions.

For student agent i, let π_θ^i represent the actor network approximating the policy, and V_ϕ^i denote the critic network approximating the value function, where θ and ϕ are the parameters of the actor and critic networks, respectively.

The policy network of each CAV can be updated as follows:

$$\mathcal{J}_i(\theta) = E_{\pi_\theta^i}\left[\log \pi_\theta^i(a_{i,t}|s_{i,t})A_i^t\right], \qquad (14)$$

where $A_i^t = r_{i,t} + \gamma V_{\phi_i}(s_{i,t+1}) - V_{\phi_i}(s_{i,t})$ is the advantage function and $V_{\phi_i}(s_{i,t})$ is the state value function.

To ensure the effectiveness of learning and the independence of the agent's policy, we introduce a regularization term in the loss function:

$$\mathcal{L}_{i_{\text{norm}}} = \lambda \mathbb{E}_{s \sim \pi_\theta} \mathcal{H}\left(\pi_T(\cdot|s)||\pi_\theta^i(\cdot|s)\right) \qquad (15)$$

where the regularization term $\mathcal{H}\left(\pi_T(\cdot|s)||\pi_\theta^i(\cdot|s)\right)$, describes the difference between the teacher policy π_T and the student policy π_θ^i, calculated using the Kullback-Leibler (KL) divergence between these two policies. λ is an annealing parameter we introduce to control the student agent's dependence on the teacher agent. When λ is set to zero, the student's learning process simplifies to a standard RL process, unaffected by the teacher agent. At the initial stage of training, we initialize the annealing parameter λ with a large value. This setup ensures that the student agent focuses more on the guidance provided by the LLM-based teacher agent, aiming to align its policy with the teacher's policy. As training progresses, we gradually decay λ linearly, allowing the student agent to shift its focus towards maximizing its expected return. By reducing the influence of the teacher's guidance, the student agent becomes more independent in its decision-making process and emphasizes its own learning strategy.

Therefore, for each agent i, the actor ultimately updates the policy by minimizing the following objective function:

$$\begin{aligned} \mathcal{L}_i(\theta) &= \mathcal{J}_i(\theta) + \mathcal{L}_{i_{\text{norm}}} \\ &= \mathcal{J}_i(\theta) + \lambda \mathbb{E}_{s \sim \pi_\theta} \mathcal{H}\left(\pi_T(\cdot|s)||\pi_\theta^i(\cdot|s)\right) \end{aligned} \qquad (16)$$

For each student agent i, the critic network is trained using a stochastic gradient descent (SGD) method to minimize the value loss, defined by the following loss function:

$$\mathcal{L}^i(\phi) = \hat{\mathbb{E}}_t[(V_\phi^i(s_{i,t}) - R_{i,t})^2], \qquad (17)$$

where R_t is the cumulative return, and $V_\phi(s_{i,t})$ is the current value estimate of agent i. The schematic diagram is shown in Fig.3.

Additionally, during the actual exploration process, the student agents are allowed to leverage the teacher agent's decision-making tools to further enhance the quality of their decisions. The teaching process will be executed for a couple of iterations E_{te} to improve the learning efficiency of students. After that, the student agents will continue

to explore by themselves to facilitate policy diversity. The learning process of the student agents is summarized in Algorithm 2.

Algorithm 2: The Procedure of Student Agents Group's Policy Learning

---

Inputs : Teacher agent, students' policies $\{\pi_1, \cdots, \pi_k\}$, number of teaching iterations E_{te}, number of self-learning iterations E_{self}, maximum timestep T_{max}

Output: θ

---

1 Phase 1: Policy Distillation
2 **for** $episode = 1$ to E_{te} **do**
3   **for** $t = 1$ to T_{max} **do**
4   Collect rollouts following the student's initial policy;
5   Get action of the teacher according to Alg. 1;
6   Update $\mathcal{D}_i \leftarrow (\mathbf{o}, a^*_{i,t}, r_{i,t}, \mathbf{o}'_{i,t})$;
7   **for** *Student* $i = 1 \in \mathcal{V}$ **do**
8 $\theta^i \leftarrow \theta^i - \alpha\nabla_{\theta^i}(\mathcal{J}_i(\theta^i) + \lambda_i \mathbb{E}_s \mathcal{H}(\pi_T(\cdot|s)||\pi^i_\theta(\cdot|s)))$;

9 Phase 2: Self learning
10 **for** $episode = 1$ to E_{self} **do**
11   **Initialize** buffer of student agent $\mathcal{D} \leftarrow \emptyset$;
12   **for** $Student\ i = 1 \in \mathcal{V}$ **do**
13   Get observation $o_{i,t}$;
14   Update action $a_i = \pi_{\theta_i}(o_{i,t})$;
15   Update $\mathcal{D}_i \leftarrow (\mathbf{x}_{i,t}, a^*_{i,t}, r_{i,t}, \mathbf{x}'_{i,t})$;
16   **for** *Student* $i = 1 \in \mathcal{V}$ **do**
17   Update critic network: $\phi^i \leftarrow \phi^i - \alpha\nabla_{\phi^i}(\mathcal{J}_i(\phi^i))$;
18   Update actor network: $\theta^i \leftarrow \theta^i - \beta\nabla_{\theta^i}(\mathcal{J}_i(\theta^i))$;

V. EXPERIMENT

A. Experimental Setups

Our environment simulator is developed based on highway-env [25], an open-source and flexible simulation platform for autonomous driving. We evaluate our method in a ramp merging scenario, which is one of the most complex traffic scenarios in the real world. CAVs on the ramp must accurately assess safe distances and merging timing, while coordinating with other CAVs on the ramp and main road to ensure safe merging. As shown in Fig.4, we design two ramp merging scenarios, and for each scenario, we define three different traffic densities (easy, medium, and hard) to simulate a range of real-world conditions. The three traffic densities generate between 2-4, 4-6, and 6-8 CAVs and HVs randomly. In these scenarios, all CAVs are controlled by our model, while HVs are managed by IDM and MOBIL to simulate basic interaction behaviors. We select MAACKTR [26], MAPPO [27], and MAA2C as baselines to compare against the performance of our method.
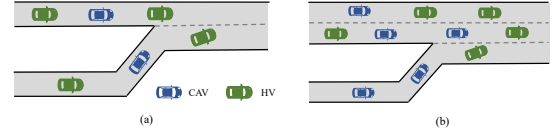

Fig. 4. The two scenarios used in our experiments, (a) scenario #1, (b) scenario #2.

B. Implementation details

For the MARL algorithm, we set the discount factor $\gamma = 0.99$, exploration rate $\epsilon = 1e - 5$, and advantage decay rate $\alpha = 0.99$. The learning rates for both the actor and critic networks are set to $5e^{-4}$. The priority coefficients α_1, α_2, and α_3 are uniformly set to 1. The algorithms are evaluated by running 3 episodes every 200 training episodes, with all models being trained over 20,000 episodes in total. GPT-4o is employed as our base LLM model, facilitating high-level logical reasoning and decision-making for the teacher agent. The experiences derived from teacher agent are utilized for the initial 2,000 episodes of training. After this phase, student agents are allowed to explore independently. To ensure reliability, all scenarios under different settings are replicated three times with distinct random seeds to obtain consistent results.

C. Performance Analysis

1) Overall Performance

The training curves for all methods under different conditions and scenarios are shown in Fig.5. As observed, in all experimental scenarios, our method (LEMAA2C) consistently outperforms the MARL baselines. Furthermore, they demonstrate significantly higher exploration efficiency in the early stages of training, which effectively boosts the training efficiency of the student agents. This strongly validates the effectiveness of our approach. Meanwhile, we also compare our method with some LLM-based approaches. As shown in Table I, our method exhibits the best performance across various tests, owing to its stronger exploration capability in the later stages of training.

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT MODELS UNDER VARYING DIFFICULTY LEVELS.

Scenario	Model	Easy	Medium	Hard
Scenario 1	GPT-3.5 [28]	91%	69%	23%
	GPT-4o [29]	98%	82%	58%
	DiLu [30]	96%	77%	60%
	Ours	**100%**	**96%**	**87%**
Scenario 2	GPT-3.5	92%	72%	31%
	GPT-4o	**100%**	78%	30%
	DiLu	99%	82%	34%
	Ours	**100%**	**81%**	**67%**

2) Safety Performance

The model's overall performance is tested under various experimental conditions, as shown in Tab.II. Post Encroachment Time (PET), a key safety metric that quantifies the
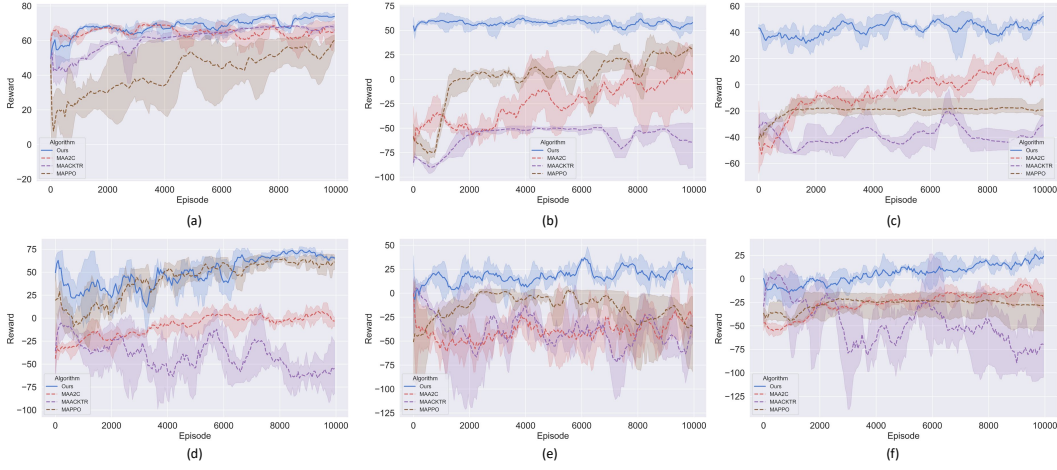
Fig. 5. The training results under different experimental settings, (a), (b) and (c) are experimental results with easy, medium and hard mode at scenario #1, respectively; (d), (e) and (f) are experimental results with easy, medium and hard mode at scenario #2, respectively.

TABLE II

PERFORMANCE ACROSS SCENARIOS FOR VARIOUS METHODS UNDER DIFFERENT DIFFICULTY SETTINGS

Difficulty	Model	Evaluation Reward		Collision Rate		Average Speed		Average PET	
		Sce-1	Sce-2	Sce-1	Sce-2	Sce-1	Sce-2	Sce-1	Sce-2
Simple	MAA2C	67.52	53.03	0.01	0.02	28.14	25.26	18.68	19.97
	MAPPO	30.31	58.63	0.17	0.09	23.65	28.79	13.29	19.61
	MAACKTR	13.96	-29.92	0.27	0.51	25.51	25.46	11.08	7.97
	Ours	**80.44**	**75.15**	**0.00**	**0.00**	27.14	27.70	19.07	17.67
Medium	MAA2C	16.85	-1.58	0.23	0.30	23.96	25.17	8.76	7.67
	MAPPO	47.68	2.35	0.09	0.34	21.76	22.65	9.33	12.59
	MAACKTR	0.07	-14.99	0.44	0.43	23.48	25.64	7.48	8.27
	Ours	**56.91**	**35.97**	**0.04**	**0.19**	24.11	26.07	7.16	7.40
Hard	MAA2C	17.76	-0.64	0.30	0.53	22.29	22.63	7.01	4.37
	MAPPO	-37.03	-35.55	0.81	0.96	25.29	22.66	4.05	4.18
	MAACKTR	-12.47	-25.11	0.65	0.51	20.65	25.50	5.74	9.28
	Ours	**36.87**	**14.33**	**0.13**	**0.33**	21.98	23.11	5.88	4.00

interaction and risk level between two vehicles, is used as one of the primary evaluation criteria. It is evident that our method consistently achieves the lowest collision rate across all tests. We also evaluate the safety performance of the teacher agent under different conditions and compare it with the student agents, as shown in Tab. III. In most scenarios, LAMAA2C outperforms LLM in the final performance, demonstrating that our method possesses strong exploration capabilities that further enhance performance. We believe this is due to the fact that as training progresses, the student model is able to integrate its own exploration and learning through policy updates more freely, in addition to receiving guidance from the teacher.

3) Efficiency Performance

The average speed of all CAVs, shown in Tab.II, is analyzed to assess the efficiency of cooperative driving across different methods. We observe that as the difficulty of tasks increases, the average speed of the CAVs decreases, which likely represents a compromise made to ensure safety over efficiency. However, compared to other baseline methods, such as MAPPO and MAACKTR, our approach not only maintains better safety performance but also ensures driving

TABLE III

PERFORMANCE OF TEACHER AND STUDENTS UNDER VARYING DIFFICULTY LEVELS.

Scenario	Role	Easy	Medium	Hard
Scenario 1	Teacher	98%	82%	58%
	Students	**100**%	**96**%	**87**%
Scenario 2	Teacher	**100**%	78%	30%
	Students	**100**%	**81**%	**67**%

efficiency. This demonstrates that our method effectively balances driving speed with safety.

4) Cross-Validation

To evaluate the adaptability and generalization capabilities of the learned strategies, we test the models in task environments beyond their training conditions. For example, models trained under the hard mode are now tested on easy and medium mode tasks, with results shown in Tab.IV. We observe that models trained in hard mode not only outperform the original medium mode models in terms of safety and efficiency but also excel in other scenarios. This

demonstrates the exceptional adaptability of our model.

TABLE IV
PERFORMANCE COMPARISON OF MODELS ACROSS DIFFERENT
SCENARIOS AND DIFFICULTY LEVELS.

Original Level	Applied Level	Evaluation Reward		Collision Rate	
		Sce-1	Sce-2	Sce-1	Sce-2
HARD	Simple	62.66	46.18	0.02	0.13
	Medium	51.16	70.78	0.07	0.00
MEDIUM	Simple	69.31	61.64	0.00	0.02

VI. CONCLUSIONS

To enhance exploration and learning efficiency in cooperative decision-making for CAVs, we introduced LDPD, a language-driven policy distillation framework to facilitate MARL agents learning. Within this framework, the language model, as the teacher, makes complex reasoning decisions based on observations from student agents and stores these decision-making experiences. Student agents learn the teacher's prior knowledge through gradient updates in their policy networks. Extensive experiments demonstrate that distilled information from the teacher significantly enhances the exploration efficiency of student agents. Compared to baseline methods, there is a notable overall improvement in the agents' performance, eventually surpassing the expert decision-making level of the teacher. In the future, we plan to extend our method to test additional MARL algorithms and apply it across a broader range of decision-making scenarios.

REFERENCES

[1] Peng Hang, Chao Huang, Zhongxu Hu, and Chen Lv, "Decision making for connected automated vehicles at urban intersections considering social and individual benefits," *IEEE transactions on intelligent transportation systems*, vol. 23, no. 11, pp. 22549–22562, 2022.

[2] Jiaqi Liu, Ziran Wang, Peng Hang, and Jian Sun, "Delay-aware multi-agent reinforcement learning for cooperative adaptive cruise control with model-based stability enhancement," *arXiv preprint arXiv:2404.15696*, 2024.

[3] Xiaobo Qu, Dawei Pi, Lei Zhang, and Chen Lv, "Advancements on unmanned vehicles in the transportation system," 2023.

[4] Peng Hang, Chen Lv, Chao Huang, Yang Xing, and Zhongxu Hu, "Cooperative decision making of connected automated vehicles at multilane merging zone: A coalitional game approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 4, pp. 3829–3841, 2021.

[5] Jiawei Zhang, Shen Li, and Li Li, "Coordinating cav swarms at intersections with a deep learning model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 6, pp. 6280–6291, 2023.

[6] Jiaqi Liu, Peng Hang, Xiaoxiang Na, Chao Huang, and Jian Sun, "Cooperative decision-making for cavs at unsignalized intersections: A marl approach with attention and hierarchical game priors," *arXiv preprint arXiv:2409.05712*, 2024.

[7] Dong Chen, Mohammad R Hajidavalloo, Zhaojian Li, Kaian Chen, Yongqiang Wang, Longsheng Jiang, and Yue Wang, "Deep multi-agent reinforcement learning for highway on-ramp merging in mixed traffic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 11623–11638, 2023.

[8] Ruiqi Zhang, Jing Hou, Florian Walter, Shangding Gu, Jiayi Guan, Florian Röhrbein, Yali Du, Panpan Cai, Guang Chen, and Alois Knoll, "Multi-agent reinforcement learning for autonomous driving: A survey," *arXiv preprint arXiv:2408.09675*, 2024.

[9] Yang Zhang, Shixin Yang, Chenjia Bai, Fei Wu, Xiu Li, Xuelong Li, and Zhen Wang, "Towards efficient llm grounding for embodied multi-agent collaboration," *arXiv preprint arXiv:2405.14314*, 2024.

[10] Shiyu Fang, Jiaqi Liu, Ding Mingyu, Chen Lv, Peng Hang, and Jian Sun, "Towards interactive and learnable cooperative driving automation: a large language model-driven decision-making framework," *arXiv preprint arXiv:2409.12812*, 2024.

[11] Senkang Hu, Zhengru Fang, Zihan Fang, Xianhao Chen, and Yuguang Fang, "Agentscodriver: Large language model empowered collaborative driving with lifelong learning," *arXiv preprint arXiv:2404.06345*, 2024.

[12] Zihao Zhou, Bin Hu, Chenyang Zhao, Pu Zhang, and Bin Liu, "Large language model as a policy teacher for training reinforcement learning agents," *arXiv preprint arXiv:2311.13373*, 2023.

[13] Leslie N Jacobson, Kim C Henry, and Omar Mehyar, *Real-time metering algorithm for centralized control*, Number 1232. 1989.

[14] Yuning Wang, Jinhao Li, Tianqi Ke, Zehong Ke, Junkai Jiang, Shaobing Xu, and Jianqiang Wang, "A homogeneous multi-vehicle cooperative group decision-making method in complicated mixed traffic scenarios," *Transportation Research Part C: Emerging Technologies*, vol. 167, pp. 104833, 2024.

[15] Zhi Li, Qichao Wang, Junbo Wang, and Zhaocheng He, "A flexible cooperative marl method for efficient passage of an emergency cav in mixed traffic," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[16] Oier Mees, Jessica Borja-Diaz, and Wolfram Burgard, "Grounding language with visual affordances over unstructured data," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11576–11582.

[17] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al., "Competition-level code generation with alphacode," *Science*, vol. 378, no. 6624, pp. 1092–1097, 2022.

[18] Yicheng Guo, Jiaqi Liu, Rongjie Yu, Peng Hang, and Jian Sun, "Mappo-pis: A multi-agent proximal policy optimization method with prior intent sharing for cavs' cooperative decision-making," *arXiv preprint arXiv:2408.06656*, 2024.

[19] Chuanneng Sun, Songjun Tu, and Dario Pompili, "Llm-based multi-agent reinforcement learning: Current and future directions," *arXiv preprint arXiv:2405.11106*, 2024.

[20] Danyang Zhang, Lu Chen, Situo Zhang, Hongshen Xu, Zihan Zhao, and Kai Yu, "Large language models are semi-parametric reinforcement learning agents," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[21] Matthijs TJ Spaan, "Partially observable markov decision processes," *Reinforcement learning: State-of-the-art*, pp. 387–414, 2012.

[22] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao, "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2022.

[23] Martin Treiber, Ansgar Hennecke, and Dirk Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, pp. 1805, 2000.

[24] Arne Kesting, Martin Treiber, and Dirk Helbing, "General lane-changing model mobil for car-following models," *Transportation Research Record*, vol. 1999, no. 1, pp. 86–94, 2007.

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

[26] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," *Advances in neural information processing systems*, vol. 30, 2017.

[27] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24611–24624, 2022.

[28] OpenAI, "Introducing ChatGPT," 2023.

[29] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al., "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[30] Licheng Wen, Daocheng Fu, Xin Li, Xinyu Cai, Tao Ma, Pinlong Cai, Min Dou, Botian Shi, Liang He, and Yu Qiao, "Dilu: A knowledge-driven approach to autonomous driving with large language models," *arXiv preprint arXiv:2309.16292*, 2023.