CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Fine Tuning a Large Language Model for Tactical Decision Making in Level 3 Autonomous Trucks

Master's thesis in Computer science and engineering

Yifan Zhao
Mengyuan Wang

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2025

Master's thesis 2025

Fine Tuning a Large Language Model for Tactical Decision Making in Level 3 Autonomous Trucks

Yifan Zhao
Mengyuan Wang

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2025

Fine Tuning a Large Language Model for Tactical Decision Making in Level 3 Autonomous Trucks
Yifan Zhao, Mengyuan Wang

Supervisor: Deepthi Pathare, Department of Computer Science and Engineering, Chalmers & Volvo Group Trucks Technology
Examiner: Morteza Haghir Chehreghani, Department of Computer Science and Engineering, Chalmers

Master's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2025

A Chalmers University of Technology Master's thesis template for LaTeX
Yifan Zhao
Mengyuan Wang
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

This thesis investigates whether a Large Language Model (LLM) can be adapted to serve as the tactical brain of a Level-3 autonomous truck through supervised fine-tuning (SFT). We first generated highway driving scenarios in the SUMO simulator, pairing each coded scenario with high-level maneuvering decisions, which include ACC set speed, time gap, lane change intent, generated by a powerful LLM. The resulting scenario-decision pairs constitute a domain-specific dataset that captures a variety of safety-critical interactions between a self-propelled truck and surrounding traffic. Three open-source modelsMeta-Llama-3.1-8B, Qwen 2.5-14B, and DeepSeek-R1-Distill-Llama-8B-are then fine-tuned with Low-Rank Adaptation (LoRA). A modular control stack separates the LLMs high-level reasoning from a low-level Intelligent Driver Model (IDM) that executes longitudinal and lateral motion, mirroring real-world practice.

Evaluation of SUMO episodes showed that fine-tuning improved the quality of decisions. All models improve the achieve a high success rate. Despite the fact that the fine-tuned LLMs achieved a high success rate, we discovered that the LLMs does not fully learn a perfect set of driving strategies. The LLMs does not completely learn the truck's lane changing strategy. As a result, the LLMs behaved somewhat clumsily in some scenarios. After fine-tuning, some unsafe decisions were eliminated, which confirms the improvement of logical consistency. The models also generate concise natural language rationales, improving the interpretability and compliance of the system. This study shows that when equipped with a tailored driving dataset and efficient LoRA fine-tuning, a modestly sized LLM can provide a degree of safe, efficient, and interpretable but not perfect tactical decisions for self-driving trucks.

Keywords: Large Language Models (LLMs), Autonomous Driving, Open-Source Models, Supervised Fine-Tuning, Prompt Engineering

Acknowledgements

This project is carried out at Safe and Efficient Driving Division at Volvo Group Trucks Technology, and we extend our sincere gratitude for their invaluable support. We would like to especially thank our supervisor, Deepthi Pathare, for her exceptional guidance, insightful feedback, and continuous encouragement throughout the course of this project.

Special thanks to our examiner Morteza Haghir Chehreghani at Chalmers University of Technology. His profound knowledge and academic rigor have greatly enhanced the quality of our work.

We are also grateful for the equipment and computational resources provided by Volvo, which made the experimental work possible. Special thanks go to Alexander Bergersen for his assistance with cluster support and infrastructure access.

This work has greatly deepened our interest in the application of Large Language Models in real-world domains and has inspired us to further explore this exciting and rapidly evolving field.

Mengyuan Wang, Gothenburg, 2025-06-12
Yifan Zhao, Gothenburg, 2025-06-12

Contents

List of Figures

List of Tables

1

Introduction

1.1 Background

The recent advancements and development in autonomous driving have revolutionized the field of transportation, demonstrating its promising potential for enhancing safety and efficiency, and is gradually integrating into our daily life. Modern autonomous vehicles integrate multiple subsystems, including perception, planning, decision-making, and control to act as an intelligent agent on the road. Among these, the decision-making module serves as a "brain" for the autonomous vehicles, responsible for translating surrounding environment and goals into safe and efficient driving decisions. The long distance trucking takes a significant portion of global freight, which means it can benefits greatly from automation.

The Society of Automotive Engineers (SAE) defines Level 3 (L3) autonomy as a mode in which the vehicle can handle all aspects of the driving task in certain condition, but a human driver must be available to take over the control when the system requests. In a L3 vehicle, the automated system performs dynamic driving tasks, including lane change, acceleration and braking, without continuous human supervision. In terms of long distance trucking, L3 autonomy is particularly compelling for highway scenarios. Highway scenarios offer relative structured environment that can be suitable for automation very well. For example, there is no traffic lights in a highway, lane marking is structured, and traffic flow is consistent.

Tactical decision-making consists of adaptive cruise control (ACC) and lane change maneuvers. ACC is a type of advanced driver-assistance system for road vehicles that automatically adjusts the vehicle speed to maintain a safe distance from vehicles ahead. Lane change maneuver involves changes in both the longitudinal and lateral velocity as well as movement in the presence of other moving vehicles, which can be perceived as challenging.

Recently, Large Language Models (LLMs) have advanced significantly. Because of their remarkable ability of context understanding, reasoning, generating coherent response in natural language, they are good potential tactical decision makers in a highway scenario of L3 autonomous vehicles. Generative AI can perform complex task by following instructions expressed in a human language way, which shows a so-called common sense reasoning that was unexpected from previous AI systems. For examples, GPT-o1 shows a strong performance on different tasks range from daily

dialog, mathematical computation and logical reasoning. This ability that reason the instructions instantly has led to researchers to explore LLMs in a decision-making domain.

In this master's thesis, we conduct a feasibility study of how to leverage capabilities of LLMs, including natural language understanding, complex task reasoning, for decision-making. The primary focus of this study would be a scientific way to fine-tune a LLM, which enables model to learn the pattern of tactical decision-making for autonomous vehicles. Our methodology involves dataset generation, models fine-tuning and evaluation. An powerful LLM is applied for a simulation environment to generate the dataset. Then smaller LLMs take advantage of the generated dataset to fine-tune. Finally, We evaluate our models in a simulation environment, which uses some metrics designed by human evaluators.

1.2 Purpose

The primary objective of this project is to explore some effective methods, for examples, use fine-tuning techniques and prompt engineering, for an open-source LLM to make tactical decisions in a L3 autonomous vehicle. In addition, we tend to access the performance in comparison to different LLMs. Specifically, in a simulation environment, we test LLMs' success/collision rate, average speed of the ego vehicle, and driving efficiency.

To concretize the objective, we purpose a research question:

How can a LLM be adapted to a decision maker in a L3 driving vehicle

This research question can be divided into three specific sub-questions:

- 1. What is the dataset we need to fine-tune LLMs? How can we obtain this dataset.
- 2. What supervised fine-tuning methods we should use?
- 3. How should we design the evaluation test in a simulation environment?

1.3 Limitations

Although this project aims to explore some effective methods for an open-source LLM to make tactical decisions in a L3 autonomous vehicle, it is important to acknowledge certain limitations:

- **1. Scope**: The evaluation of the LLMs is limited to a simulation environment instead of the real-world testing, limiting the immediate applicability of findings. In our scenario, LLMs can be used in a real-world settings to validate and monitor the correctness of tactical decisions to improve the overall model robustness.

- **2. Dataset**: Due to the time constraints of this thesis, the synthesized dataset will be used for fine-tuning. Only a limited number of data samples will be manually validated due to workload constraints. The impact of data quality on model performance is comparable to the presence of noise in any driving dataset.

1.4 Thesis Outline

In Chapter 1 we introduced the background of our thesis, and we continue to explore previous studies that researches have done on decision-making in autonomous vehicles. Chapter 3 covers the theory that related to LLMs, prompt engineering, and supervise fine-tuning. Chapter 4 clarifies the methodology used in our study. Followed by Chapter 5, which presents the experiment's result. and discusses the observation and insights. Last but not least, Chapter 6 concludes the project by summarizing the findings and discusses the future work.

2

Related Work

The related work chapter investigates how previous studies have proposed some methods that tackle the tactical decision-making problem in a autonomous vehicles. One common approach is to use a rule-based system, which is efficient in ensuring safety in autonomous vehicles. Other more comprehensive strategies involve more advanced machine learning algorithms, which also include using large language models. Finally, we discuss how our work differs from previous research and the contribution it brings to this study area.

2.1 Classical Methods to Tactical Decision-Making

Early autonomous driving systems are based on knowledge-driven, rule-based strategies to make tactical decisions on highways [1]. These methods embed the knowledge of the traffic rules from the experts and driving heuristics into vehicles' driving logic. For instance, a finite-state machine can be used to switch between driving behaviors, such as lane-keeping, overtaking, or following based on preset conditions. In such rule-based systems, an if-then style rule and logic reasoning would be applied to select maneuvers, which ensures decisions remain interpretable and compliant with traffic laws [2]. In addition, a well fine-tuned rule database can yield safe and predictable behaviors; Pellkofer and Dickmanns proposed a behavior decision module, which execute task plans generated by task planning experts rule database [3]. Similarly, Noh and An proposed a decision-making framework for highway environments, capable of reliably and robustly assessing collision probabilities under current traffic conditions and automatically determining appropriate driving strategies [4].

These rule-based systems outperform in safety and Interpretability, however, they face limitations in complex, dynamic environments. For example, designing an finite-state machine to imitate human drivers' behaviors can be extremely challenging, because the number of states can explode as one enumerates more traffic situations and exceptions. Human driving behaviors often involve subtle judgments that are difficult to capture through strict rules. Therefore, rule-based systems are struggling with the scenarios that are not expected by their designers. Although highway scenarios are usually structured(e.g. well defined lanes, fewer edge cases), the rule-based systems are brittle if they confronted with out-of-distribution events (e.g. an accident scene or an aggressive cut-in) that werent encoded in the rules [6].

In conclusion, these classical approaches form the early toolkit for L3 autonomous

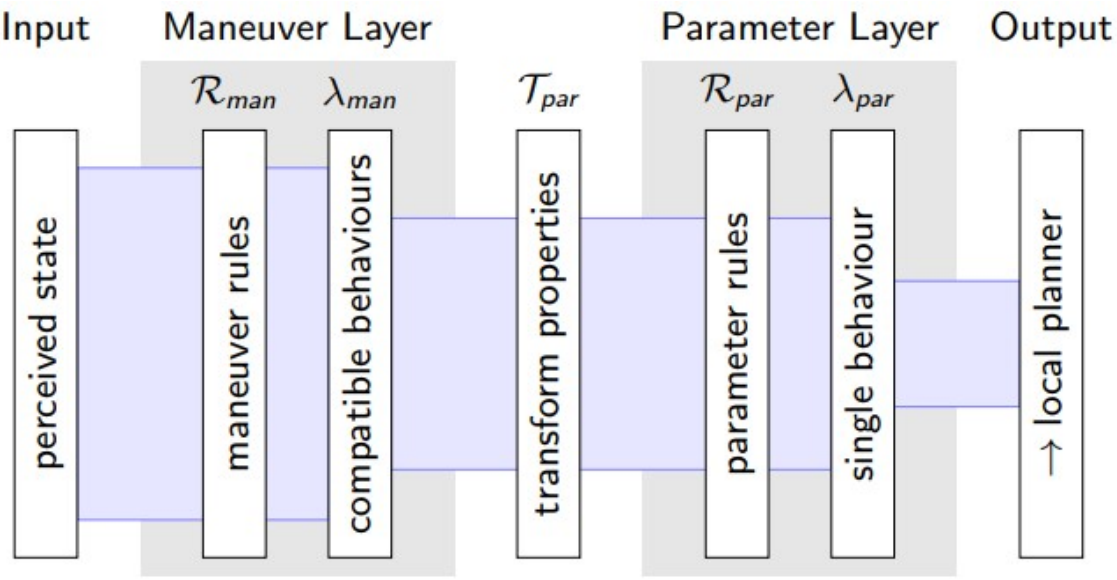

Figure 2.1: A rule-based decision-making system. Source: Adapted from [5]

driving on highway. They provides transparency and some basic guarantees, but at the cost of manual tuning and extremely limited adaptability.

2.2 Advanced Machine Learning Methods

To overcome the inflexibility of manual rules, researchers turn to the focus on data-driven methods. Machine learning plays an important role in this field. It allows an autonomous agent to learn the driving pattern from previous examples and experience, rather than relying solely on hand-crafted logic. Two major paradigms have been explored: imitation learning and reinforcement learning.

2.2.1 Imitation Learning

Imitation learning uses human driving behaviors logs to train a model that imitate human's driving decisions. For example, models may learn when to change the lane or what they should respond when there is a lower vehicle ahead by observing many human drivers in similar situations. NVIDIA proposed the landmark DAVE-2 system, which is an end-to-end neural network controllers that shows that a convolutional neural network could map camera images directly to steering commands by learning from human demonstrations [7]. This kind of models can reproduce the human driving behaviors quite well. However, the disadvantage of this model is also quite well-known - models may lack a deep understanding of the scenario and fail when encountering scenarios that are absent or rare in the training data. Kim et al. [8] elaborated on the shortcomings of this approach, pointing out that it lacks a semantic understanding of the content of the image, which makes it difficult to deal with data outside of the training distribution.

2.2.2 Reinforcement Learning

Reinforcement learning(RL) is one of the three basic machine learning paradigms. It concerns how an intelligent agent should take actions in a dynamic environment in order to maximize a reward signal. In the context of tactical decision-making on highway, RL agent can be awarded for safety, speed, and efficiency. Deepthi et al. [9] designed a total cost of operation (TCOP) reward for the RL agent, embedding trucking costs, such as fuel, brake wear, guides the agent toward commercially relevant behaviors.

Deep reinforcement learning (DRL) methods have achieved promising results in a simulation environment. The agent can learn efficient overtaking and lane-changing behaviors that might be hard to coded manually. It has been proved that DRL outperforms simple rule-based systems in collision avoidance scenarios [10]. Similarly, many works have used deep q-networks (DQN), which is a variant of DRL, or deep deterministic policy gradient (DDPG) to learn a highway policy for the RL agent. Peng et al [10] used an upper DQN to handle lane change decisions and a lower DDPG to control the trajectory; the two layers were trained in coordination to complete smooth lane changes in traffic.

Although RL outperforms other methods, it has its own challenges as well. A good RL agent needs a well-designed reward function, this relies heavily on the researchers designing ability and experience. In addition, the RL agent requires extensive exploration in simulation.

2.3 LLMs for Decision-Making

Given the limitations of machine learning methods, researchers begun to explore the possibility of using LLMs' reasoning ability to improve autonomous driving decision-making. LLMs such as GPT3 [11] are pre-trained on a vast generic dataset, which gives them generalized reasoning capabilities. Although LLMs are originally developed for text generation like chatbots, text translation, they have the potential of being a decision-maker. Recent works [12] implied that integrating the LLM into autonomous vehicle's planning module could make the system more generalizable and interpretable.

A prominent work product is the use of the LLM as a high-level planner that directs the vehicle's actions, while the low-level controller executes the plan. For instance, in the study [13], the authors propose an approach where an LLM serves as the core decision-making module in the driving stack. The LLM is prompted with a description of current scenarios and outputs a natural language reasoning about what the ego vehicle should do. These textual decisions are then translated into numerical driving commands by the Model Predictive Controller (MPC). This method enabled human-like handling of complex situations.

Similarly, the work [14] introduce DriveMLM, an LLM-based framwork for autonomous driving behavior planning. It employs a multimodal LLM as the planning module: it takes as input structured observations from various sensors, such as

camera and radar, high-level route commands, and even driving rules, and produces both a driving decision and an explanatory text justification.

Decisions for autonomous vehicles must not only be effective, but also legally and ethically acceptable. LLM has been conceived as a tool to inject knowledge of transportation laws, regulations and even ethical frameworks into the decision-making loop. In their work [15], the authors present a system that leverages a retrieval-augmented LLM to ensure traffic rule compliance. In their framwork, an agent called Traffic Regulation Retrieval (TRR) fetches relevant rules from a database of driving laws and manuals based on the vehicles current situation. Then a GPT-4-based reasoning module interprets these textual rules and evaluates the vehicles intended action against them. This approach can flag illegal and unsafe maneuvers from the knowledge of database.

One of the most immediate advantages of LLM is that it can give the reason why the model makes this decision by some prompt engineering techniques such as chain-of-thought(CoT) [16]. Unlike a typical neural network that maps inputs to outputs without human-readable reasons, which could largely enhance user trust and ease validation, an LLM can produce a CoT or a textual explanation of why a certain maneuver is recommended. CoT enables complex reasoning capabilities through intermediate reasoning steps, it not only increase the transparency of internal model's reasoning, but also improve the quality of the final output. Several systems have integrated a question-answering or commentary capability into driving control. For example, [17] proposes a DriveGPT4, which is a multimodal LLM that can take video frames from the cars cameras as input and output both low-level driving controls and a textual commentary. In use, the LLM might receive a query like "Why does this vehicle behave in this way?". These sort of queries trigger LLMs' reasoning ability in natural.

In summary, large language models are opening up new avenues for autonomous vehicle decision-making at the cognitive level. They provide natural language interface, making autonomous systems more transparent and interactive.

2.4 Motivation

Previous studies offer significant contribution to this field but also leave gaps that motivate our research. The rule based approach is suitable for structured scenarios and gives clear logical explanations, however, it could be too rigid when applied in complex and dynamic driving scenarios. And it lacks adaptability, it would be difficult to cope with emerging scenarios and requires a large number of experts to write its logic.

The machine learning approach generalizes its capabilities by learning the dataset in a way that makes it in easier to adapt to new scenarios. However, it requires massive labeled datasets, and the black-box nature of neural networks makes it difficult to explain errors when they occur.

In our research, we develop a large language models framework for tactical decision-

making, where the LLM serves as a "brain" in our system, and decides high-level actions including Adaptive Cruise Control (ACC) and lane change maneuvers in a highway scenario, as well as providing human-like text-based reasons. Specifically, we design and implement a high-level planner and low-level executor separated controller, and the LLM is responsible for high-level decision-making considering both safety and efficiency, while the low-level control is based on physical models for actual driving maneuvers, such as lane change and speed increase.

3

Theory

The theory chapter lays the theoretical foundations to support the research by summarizing the basic concepts and principles that guided the research. It starts with an overview of LLMs. Further, it investigates techniques that can enhance LLM performance such as prompt engineering and supervised fine-tuning. At last, it discusses the SUMO simulation environment used to evaluate LLM's performance.

3.1 Large Language Models

In this section, some essential background information about LLMs will be provided. LLMs is advanced generative AI models that undergo extensive unsupervised pre-training on vast text datasets to understand and generate human-like text. Examples of LLMs include OpenAI's GPT series [18], Anthropic's Claude series [19], and Meta's Llama families.

3.1.1 Terminology

Model Size: Model size refers to how large the model is, specifically, how many parameters does the model has. A typical large model size can be over 100B, which means the model has more than 100 billion parameters, such as, Llama3.1-405B. A small size, on the other hand, may only have 1-10 billion parameters. Usually, a larger model size means the model has a better ability to learn more complex patterns from the training dataset, however, it also means a larger model needs more computation resources for training and more memory space to store its parameters.

Base Model: A pre-trained language model that serves as a foundation for building more specific models for downstream tasks, such as text classification, translation.

Instruct Model: A model that is derived from base models but undergo additional fine-tuning on datasets of instructions and their corresponding outputs. This process imbues the model with the ability to follow specific directives and perform tasks more reliably [21].

Hallucinations: Model-generated content that is clearly untrue, where the model is attempting to deceive the user by making up content that doesn't exist [22].

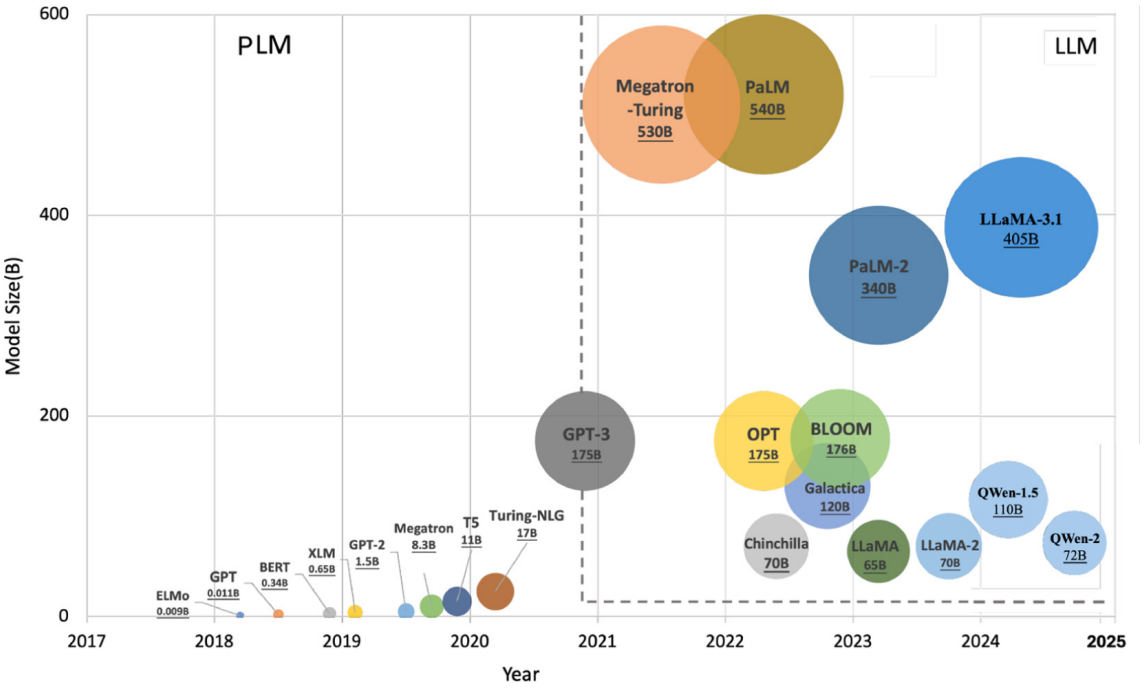

Figure 3.1: Comparison of different model sizes among popular LLMs. Source: Adapted from [20]

3.2 Prompt Engineering

Prompt engineering is an increasingly important skill set needed to converse effectively with LLMs. A typical well-designed prompt includes system instructions, questions, input data, and a example of the output.

3.2.1 Prompt Structure

System Instructions: Assigning roles to the LLM to guide text generation by offering specific context. For example, in the beginning of the prompt, use a role prompt like "You are a smart driving assistant."

Questions: The main content of the user query, on which the user expects the model to base its answer.

Input Data: A dynamic content can be changed according to different states, for example, cars' speed, lane number and relative distance.

3.2.2 Prompting Techniques

In order to retrieve the best response from the LLMs, there are several prompt techniques users can choose.

3.2.2.1 Zero-Shot

Zero-Shot Prompting is a prompting technique to guide LLMs towards new tasks. LLMs can answer queries they have never seen before. For example, an LLM can answer a prompt like "Please translate following Chinese to English for me: ", such prompt can trigger the LLM to find out the most relevant word vectors in its latent space to generate the words even it has never been trained on a translation task.

3.2.2.2 Few-Shot

Compared to zero-shot prompting, few-shot prompting offers a limited number of input-output examples to LLMs to better answer the query. This methods can significantly improve the answer format quality, which leads the LLMs to answer more logically.

3.2.2.3 Chain-of-Thought

LLMs usually struggle with complex reasoning tasks, which limits their ability and efficiency. Due to this challenge, Chain-of-Thought (CoT) are introduced as a solution to guide LLMs to think step-by-step [16]. CoT trigger model's inherent reasoning ability obtained during the pre-training by providing demonstrations that combine reasoning information with inputs and outputs, which helps model generating output through systematic steps. Unlike few-shot prompting, which provides examples to the LLMs, CoT provides a detailed reasoning chain within the examples. This detailed reasoning chain breaks down the complex task into small and easy tasks. With this divide-and-conquer mindset, the model can stimulate his latent reasoning ability, which in turn leads to the completion of complex tasks that were previously poorly performed.

3.2.2.4 Tree-of-Thought

CoT improves LLMs reasoning ability by teaching them to think loudly. Tree-of-Thought (ToT) is a framework that summaries thought chain prompts and encourages exploration of thought as an intermediate step towards solving general problems using language models. This enables LLM to self-evaluate the whole process by the intermediate thoughts made towards solving a problem through a deliberate reasoning process.

3.3 Supervised Fine-Tuning

Normally, pre-trained model are trained by unsupervised learning and may not be optimized to specific downstream tasks. Fine-tuning bridges the gap through take the advantage of general language understanding that models gained from pre-training and adapted it to a specific task by supervised learning.

During the fine-tuning process, the model's weight are adjusted based on the gradients computed by the specific task loss, which is the difference between model's prediction.

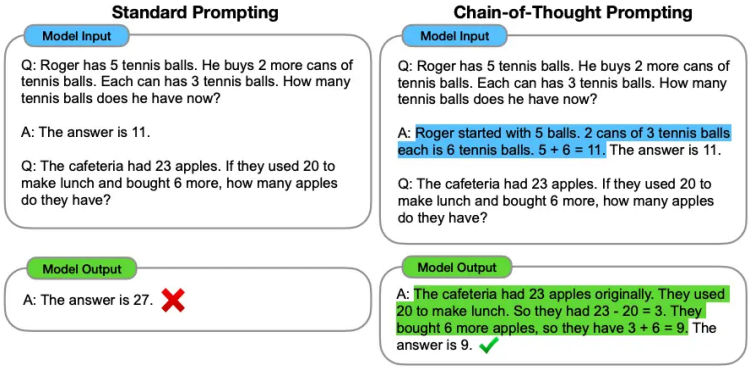

Figure 3.2: Example of a chain of thought CoT prompting. Source: Adapted from [16]

3.3.1 Full Fine-Tuning

Full fine-tuning is a traditional approach to fine-tuning that typically involves building on a pre-trained model and adjusting all of the model's parameters by continuing to train on data from a specific downstream task. The advantage of this approach is that the downstream task data can be leveraged to adjust the model's behavior, which makes the model better adapted to the specific application scenario. However, full fine-tuning may be extremely computational expensive due to the number of parameters.

3.3.2 Low-Rank Adaptation

Low-Rank Adaptation (LoRA) [23], is an efficient fine-tuning technique for large pre-trained language models compare to full fine-tuning. Its key idea is to achieve model fine-tuning by introducing trainable low-rank matrices in the Transformer layer of the model without changing the weights of the pre-trained model. This approach can significantly reduce the number of training parameters, thus reducing the demand for computational resources.

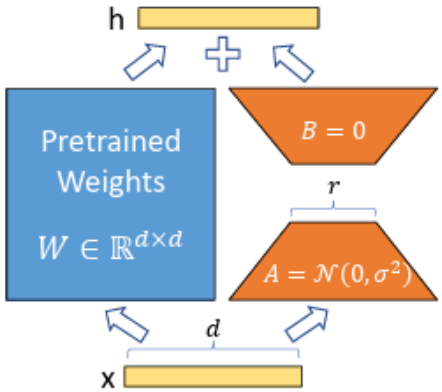

Figure 3.3: LoRA matrix decomposition method. Adapted from [23]

4

Methods

The methods chapter outlines the methodology used for this project. There are three major parts. It commences with dataset generation, which elaborates how the dataset could be obtained in this project. Next, it covers how the supervised fine-tuning be applied. Finally, it shows the process of SUMO evaluation and customizing evaluation metrics for the experiment.

4.1 Overview

We present a novel framework for tactical decision-making in autonomous truck systems, leveraging the capabilities of Large Language Models (LLMs). Our approach introduces an innovative method of translating complex driving environments into structured textual representations, allowing an LLM to reason over the scene and determine optimal driving maneuvers.

Figure 4.1 illustrates a high-level overview of the proposed system architecture. The pipeline begins with the dataset generation phase, where we utilize high-fidelity simulations to construct a diverse and comprehensive set of highway driving scenarios. A large, high-performance LLM is employed to interpret these simulated environments, generating high-quality textual descriptions alongside corresponding tactical decisions (e.g., "set desired ACC speed to 24m/s," "initiate lane change to the left," or "increase time gap to 3s").

Following dataset creation, we proceed to the fine-tuning stage. In alignment with industrial constraints regarding computational efficiency and deployment feasibility, we select a significantly smaller pre-trained LLM for this purpose. This compact model is fine-tuned on the previously generated dataset, learning to replicate the decision-making logic and contextual understanding of its larger counterpart.

The final stage of the framework is Evaluation. The fine-tuned model is deployed within a simulated environment, where its decision-making capabilities are rigorously tested across a wide array of highway scenarios. The evaluation focuses on the models ability to produce safe, efficient, and contextually appropriate tactical decisions in real-time. To support this evaluation, we utilize Simulation of Urban Mobility (SUMO), an open-source, microscopic traffic simulator. SUMO enables fine-grained modeling of individual vehicle behaviors and interactions, allowing for detailed and realistic simulations of highway traffic. It supports large-scale networks, multi-modal

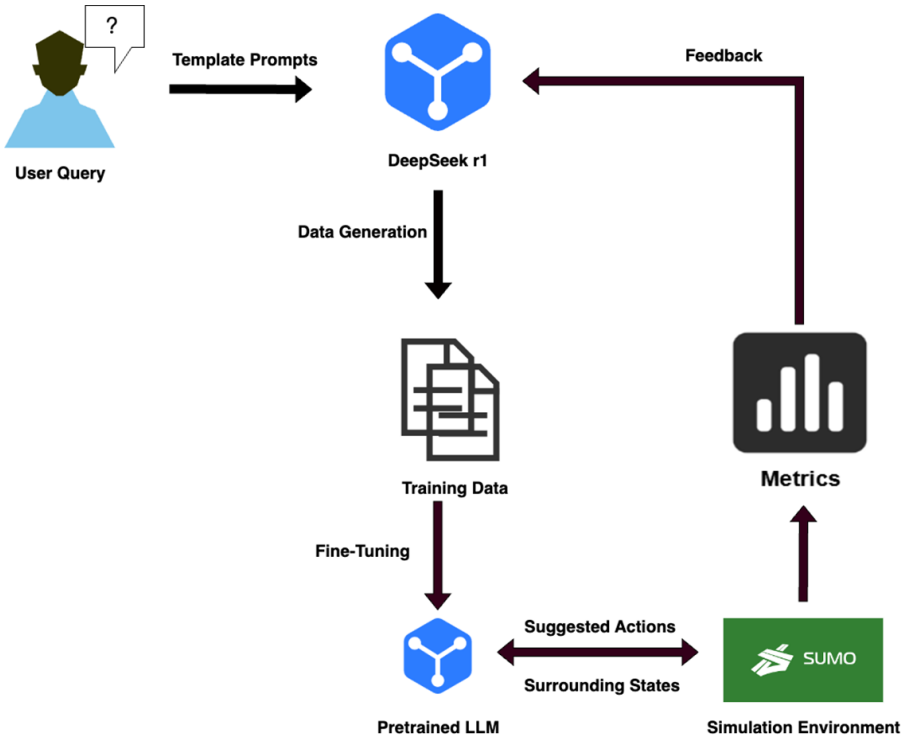

Figure 4.1: Overview structure of the project

transport systems, and time-discrete, space-continuous simulations. Due to its flexibility, extensibility, and broad adoption in both research and industry, SUMO serves as a suitable platform for testing autonomous driving strategies under a wide range of conditions.

4.1.1 Low Level Controller

To execute these decisions within the simulation, we implement a two-layer control architecture that separates high-level planning from low-level control. This modular separation mirrors real-world autonomous driving system design, where strategic reasoning is decoupled from the physical actuation of vehicle control.

We utilize LLMs to handle high-level tactical decision-making within our autonomous driving framework. This includes selecting maneuvers such as lane keeping, lane changing, or adjusting speed in response to dynamic traffic scenarios. In essence, the model determines what action the vehicle should take next, based on a textual understanding of the driving environment.

For low-level control--the component responsible for translating high-level decisions into executable physical actions–we implement the Intelligent Driver Model (IDM) [24] to govern longitudinal dynamics. Low-level control involves managing the vehicles motion along and across lanes, including precise regulation of speed (acceleration and deceleration), maintaining safe following distances, and executing lane changes when commanded by the high-level planner.

The IDM is a time-continuous car-following model that is designed to realistically

simulate human driving behavior, particularly in highway conditions. It computes the vehicles acceleration based on the current speed, desired speed, and the gap to the leading vehicle. Compared to earlier models such as Gipps' model, IDM maintains more realistic dynamics even in deterministic limits, offering smoother transitions and safer spacing behavior.

The longitudinal dynamics of the IDM are given by Eqs. 4.1. Equation 4.1a defines the acceleration $\dot{v}$ as a function of the current speed v, the desired speed v_0, and the spacing term in Eq. 4.1b.

$$\dot{v} = a\left[1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*(v, \Delta v)}{s}\right)^2\right], \tag{4.1a}$$

$$s^*(v, \Delta v) = s_0 + v\,T + \frac{v\,\Delta v}{2\sqrt{a\,b}}, \tag{4.1b}$$

$$\Delta v = v - v_{\text{lead}}, \tag{4.1c}$$

Table 4.1 defines the variables used in the IDM model.

Table 4.1: Variables in the Intelligent Driver Model

Symbol	Description
v	Ego-vehicle speed ($\mathrm{m\,s^{-1}}$)
v_0	Desiredor free-flowspeed ($\mathrm{m\,s^{-1}}$)
$\dot{v}$	Longitudinal acceleration ($\mathrm{m\,s^{-2}}$)
a	Comfortable acceleration limit ($\mathrm{m\,s^{-2}}$)
b	Comfortable deceleration limit ($\mathrm{m\,s^{-2}}$)
δ	Acceleration exponent (usually 4)
s	Actual gap to lead vehicle (m)
s_0	Minimum stand-still gap (m)
T	Desired time headway (s)
Δv	Approach rate $v - v_{\text{lead}}$ ($\mathrm{m\,s^{-1}}$)

The lateral controller control the lateral actions stay on lane, change to left lane, or change to right lane. Lane change is performed using the default LC2013 lane change model [25] in SUMO [9].

4.2 Dataset Generation

Although the LLM is already trained on a vast generic dataset, it needs a domain-specific dataset to dive deep to become a domain specialist. Hence, a specialized, vertical high-speed truck driving dataset is critical for subsequent model training.

In order to generate such a dataset, we created a simulation environment, which was built by SUMO. This simulation environment contains one truck and 7 surrounding

vehicles that would travel on a three-lane highway. At the beginning of the simulation, the surrounding vehicles and the ego truck would be randomly generated on the three lanes to ensure that there would not be any collision between them.

The entire simulation will run for 1000 steps, and we set each step to be 0.1 seconds, which means that for each step of the simulation, all vehicles will run for 0.1 seconds. The movement of the surrounding vehicles is controlled by sumo, which in principle ensures that there will be no traffic accidents. The driving of the ego-trucks is controlled by a complete system, which is divided into a high-level decision-making and a low-level controller. We use DeepSeek [26] to take care of the high-level decision-making, i.e. we let DeepSeek do the decision-making.

Equipped with DeepSeek and IDM, our ego truck can perform the simulation safely and efficient. To avoid simulations in which the surrounding traffic density is too small to generate the amount of data needed for the complex case, we maintain a window for the self-trucks and make sure that all surrounding vehicles are under that window. As soon as a vehicle exits this window (which may be too slow or too fast), we generate that vehicle on the reverse side of the exit window. Meanwhile, our ego truck will have a sensor capable of checking for vehicles in a 100 meters range, meaning that the truck will only make a decision based on the currently detected vehicles, more in line with a realistic truck autopilot.

DeepSeek makes a decision every 10 steps, which means that a high-level instruction will be sent to the low-level controller every 1 second. We ran a total of 50 simulations, obtaining 100 scenario-decision data per simulation, and ended up with a total of 5,000 highway truck driving decisions in our dataset. The dataset only contains scenario-decision pairs, and they would be integrated into prompts for the supervised fine-tuning. The fine-tuning prompt is shown in Appendix A.1.

Table 4.2 illustrates an example of generated dataset. An entry in the dataset consists of an input and an output, both of them are described by the json format. Where the input describes the current environment of the truck and the output is the decision made by the DeepSeek model, which mainly consists of acc settings and lane change.

4.3 Model Fine-Tuning

This section presents the model fine-tuning methods, which adapt weights of pre-trained LLMs to the highway scenario decision-making. We determine candidate LLMs, model fine-tuning methods, and hyperparameters.

4.3.1 Model Selection

Open-source models like Llama series and Qwen families, have demonstrated competitive performance against closed-source GPT-3.5-turbo benchmarks. Due to the commercial licenses and their ability, these open-source LLMs are more preferable for this thesis.

Table 4.2: Illustrative input-output pair from the driving decision dataset

Input	Output

<pre>{
"ego_speed": 24.8,
"ego_lane": 2,
"current_time_gap": 2.0,
"surrounding_vehicles": [
{"id": "veh1", "distance": 33.3,
"rel_position": "front",
"lane_relation": "right_lane", "speed": 21.7,
"lane": 0},
{"id": "veh3", "distance": 36.2,
"rel_position": "front",
"lane_relation": "right_lane", "speed": 21.6,
"lane": 1},
{"id": "veh4", "distance": 60.8,
"rel_position": "front",
"lane_relation": "right_lane", "speed": 21.7,
"lane": 0}
]
}</pre>
	<pre>{
	"acc_set_speed": 25,
	"time_gap": 2.0,
	"lane_change": "none",
	"reason": "No vehicles ahead in current lane,
	maintain max speed while keeping safe distance
	from adjacent-lane vehicles."
	}</pre>

In this thesis, we conduct the study using three LLMs - Llama-3.1-8B, Qwen2.5-14B, and DeepSeek-R1-Distill-Llama-8B [27], and compare their performances. These models are much smaller than the LLM we used in the dataset generation task, in the consideration of time efficiency and reasoning ability.

It is worth noting that during the fine-tuning, catastrophic forgetting could happen and this can largely effect model's ability. As new knowledge replaces prior learning, the model loses the ability to handle its original tasks. In order to avoid catastrophic forgetting, we use MMLU [28] dataset to evaluate models' reasoning ability. MMLU dataset is a large-scale, multi-task language comprehension program designed to assess and enhance the ability of language models on a variety of language comprehension tasks. The project covers a wide range of topics and domains, such as history, literature, science, mathematics, etc., and challenges the model's comprehension skills and breadth of knowledge through these diverse topics. We benchmark the score before fine-tuning and we evaluate them again after fine-tuning to check if catastrophic forgetting is happened during the fine-tuning.

4.3.2 Hyper-Parameters Tuning

In this project, LoRA method are applied for the fine-tuning. Unlike full fine-tuning, LoRA freezes model's parameters so model can remember what it learned in the pre-training in principle. In addition, LoRA allows us to train some of the dense layers in the neural network indirectly by optimizing the rank decomposition matrix of the dense layer changes during adaptation. The three models we chose are all transformer-based models, we applied LoRA to the Q,V layers of the attention layer, which were chosen for a few main considerations:

- **Parameter efficiency**: fine-tuning focuses on the parts of the model that most affect its output and performance. Q and V layers directly affect the computation of the attention weights and the final output representation, so

Table 4.3: Comparison of three large language models

Model	Size	Description
Llama 3.1 8B	7.62 Billion	Advanced open-source language model developed by Meta, designed to excel in multilingual dialogue, reasoning, and text generation tasks.
Qwen2.5 14B	14.8 Billion	Significantly more knowledge and greatly improved capabilities in coding and mathematics, especially in JSON data.
Llama 8B distilled from DeepSeek R1	7.62 Billion	Distilled from the larger DeepSeek-R1, which was trained using reinforcement learning to enhance reasoning capabilities. Strong performance in reasoning tasks, offering a balance between computational efficiency and capability.

tweaking on these layers can change the model's behavior more directly.

- **Information selection**: by adjusting layer Q, it is possible to influence how the model selects information (i.e., what information it pays attention to), and by adjusting layer V, it is possible to influence how the model makes use of certain information once it has been selected. layer K influences primarily how the information is matched, and in many cases adjustments to both Q and V will be enough to direct the model's attention to more useful information.
- **Computational efficiency**: while LoRA aims to improve parameter efficiency through low-rank updates, applying such updates on all layers still adds an additional computational burden. Therefore, selecting the layers to tune that have the greatest impact on the final performance can yield the greatest performance gains while adding the least computational cost.

Additionally, r and α are two important variables for LoRA. The r parameter refers to the rank in LoRA, which determines the size of the low-rank matrix. Typical value range for rank is 4 to 64. Lower ranks (e.g., 4, 8) are more efficient, often with minimal performance drop. Higher ranks can improve performance but increase memory and compute. The α parameter defines the learning rate scaling factor for LoRA adaptation. This parameter affects the update rate of the low-rank matrix. It is always range from 8 to 64, setting such that $\alpha/r \approx 1 - 8$ to scales the LoRA output. To prevent catastrophic forgetting, we test different combinations of rank and alpha and evaluate corresponding models' performance on the MMLU dataset.

$$\widetilde{W} = W_0 + \Delta W, \tag{4.2a}$$

$$\Delta W = \frac{\alpha}{r} B\,A, \tag{4.2b}$$

$$y = \widetilde{W}x = W_0 x + \frac{\alpha}{r} B\,A\,x, \tag{4.2c}$$

$$B \in \mathbb{R}^{d\times r}, \quad A \in \mathbb{R}^{r\times k}, \qquad r \ll \min(d,k). \tag{4.2d}$$

After fine-tuning, we used two generalized datasets, MMLU [28] and HellaSwag [29], to evaluate the models' capabilities. Among them, MMLU focuses on verifying the logical reasoning ability of the model. It includes a series of multiple-choice questions on STEM, social science, humanities, etc. We hope that our fine-tuned model can achieve the highest possible score, which represents the model's ability to reason with complex logic. HellaSwag focuses on verifying the model's ability to understand context. It is also a multi-choice question dataset, and we hope that the model can correctly find out the appropriate context. The model needs to capture important information hidden in the problem, such as numbers, and correctly understanding this information can have a significant impact on the final decision.

4.4 Experimental Design and Evaluation

This thesis work employs a dynamic highway traffic simulation environment developed using the Simulation of Urban MObility (SUMO) platform, aimed at the comprehensive evaluation of fine-tuned models in the domain of autonomous truck driving, comprehensively including foundational mathematical reasoning, logical inference, and the capacity for making context-aware, real-time driving decisions. One objective is to assess the models' ability to operate safely and efficiently within complex and dynamic traffic scenarios. In addition, this research also investigates the models ability to produce clear and reasonable justifications for their decisions, with the aim of enhancing the transparency and trustworthiness of AI-driven autonomous vehicle technologies.

4.4.1 Simulation Settings

The road network models a straight highway segment consisting of three 3.2 meters wide lanes in the primary direction of travel, spanning a total length of 5200 meters. The lane number is encoded as 0, 1, 2, with 0 meaning the right-most lane, 1 meaning the middle lane, and 2 meaning the left-most lane. Traffic flow is governed by a density of 0.02 vehicles/meter, with 80% trucks and 20% cars, representing a challenging, truck-dominated scenario. Furthermore, the ego vehicle is modeled as a semi-trailer truck with a length of 16 meters and a width of 2.55 meters, and the driver's behavior is deterministic. The Surrounding vehicles, modeled as passenger cars, are governed by the Krauss car-following model [30] and LC2013 [31] lane-changing model, with speed explicitly sampled from the uniform distributions with

an interval from 20 m/s to 30 m/s. In addition, the sensor range is set to 200 meters both in the front and rear of the ego truck. Every other vehicle that is currently driving within this range will be detected as a surrounding vehicle. A visual illustration of the simulation setting is provided in Fig 4.2. We design two sets of experiments, one with up to three surrounding vehicles, and another representing a denser traffic with up to seven surrounding vehicles.

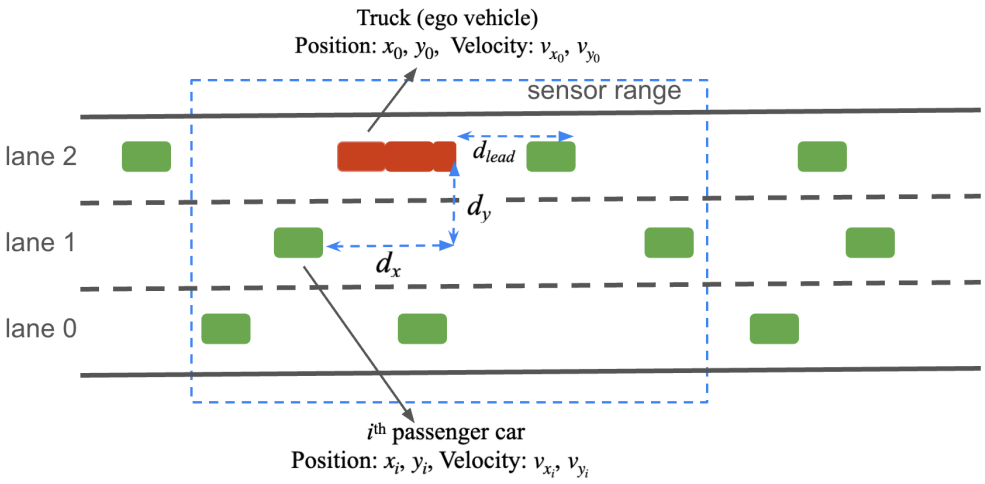

Figure 4.2: Schematic diagram of highway simulation setup

The first 250 meters are the "warm-up" area of the experiment. Therefore, each simulation episode officially begins from a longitudinal distance of 250 meters from the start of the highway segment. To ensure a dynamic and non-deterministic traffic context, the initial positions and velocity profiles of surrounding vehicles are procedurally randomized at the onset of each episode.

A simulation episode is recorded as successful if the ego vehicle navigates the entire 5200-meter highway segment and reaches the designated terminal point. Conversely, an episode is prematurely terminated and classified as a failure if a critical safety violation occurs. Such events include, but are not limited to, collisions with other vehicles or road infrastructure, or any deviation of the ego vehicle from the defined drivable roadway (e.g., driving off-road).

Each episode is limited to a maximum of 500 simulation steps. At each step, the ego vehicles current driving state, along with information of the surrounding vehicles within the sensor range, is recoded, including position, speed, and lane assignment. This information is used to construct an encoded textual representation of the driving environment, which is then provided as input to the Large Language Model (LLM), which serves as the high-level tactical planner.

The LLM analyzes the current environment context and determines an appropriate high-level tactical action (e.g., maintain speed, initiate lane change to the left, increase the time gap to 3 seconds). This decision is then passed to a low-level executor that is responsible for executing the maneuver through concrete control inputs.

The low-level controller operates at a fine-grained temporal resolution of 0.1 seconds per simulation step. It translates high-level actions into precise longitudinal and lateral control commands:

- Longitudinal speed adjustments are executed smoothly over a period of 10 simulation steps (1 second).
- Lateral lane changes are completed over 40 simulation steps (4 seconds), ensuring stable and safe transitions.

Figure 4.3 illustrates the overall simulation framework and control architecture adapted from paper [9].

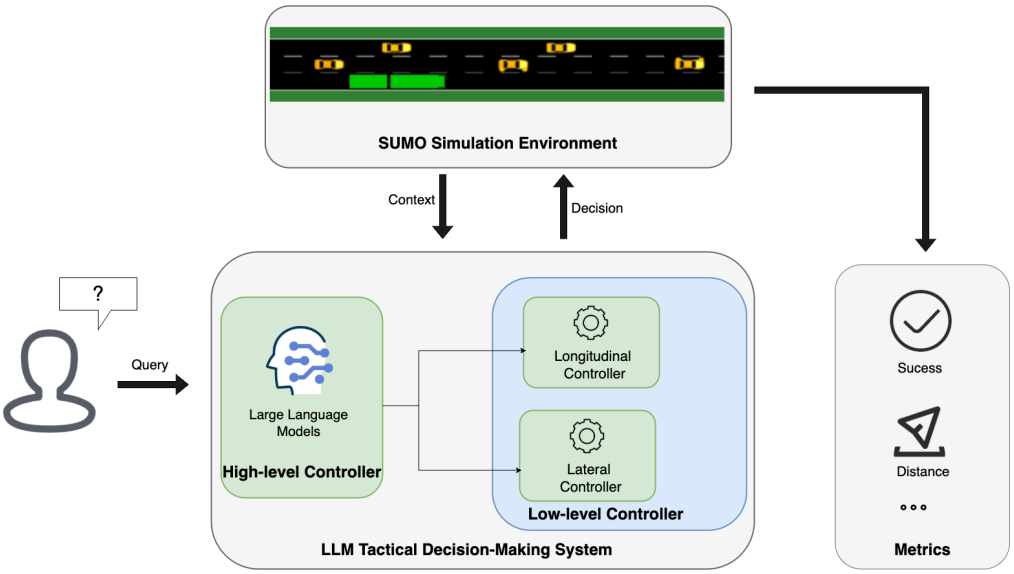

Figure 4.3: Simulation control architecture

4.4.2 Observational Space (Input to the LLM)

The LLM-based driving agent perceives the environment through a structured set of observations, categorized into ego vehicle state and surrounding vehicle states.

The intrinsic state of the ego vehicle (the autonomous semi-truck) is captured by the following parameters:

1. **Current Longitudinal Speed:** The forward velocity of the ego vehicle, measured in meters per second (m/s).
2. **Current Lane Index:** An integer representing the ego vehicle's current lane occupancy. For this three-lane highway segment, this value can be 0 (the rightmost lane), 1 (the middle lane), or 2 (the leftmost lane).
3. **Current Set ACC Time Gap:** The prevailing target time gap setting, in seconds (s), for the ego vehicle's Adaptive Cruise Control (ACC) system. This represents the desired time headway the ACC is currently configured to maintain with a preceding vehicle, and serves as an input observation for the LLM's subsequent decision on potentially adjusting this setting.

Every vehicle that is within the ego truck's simulated sensor range will be defined as a surrounding vehicle, and the following information is recorded:

1. **Vehicle Identifier (ID):** A unique identifier for each surrounding vehicle, enabling temporal tracking and consistent reference.
2. **Current Longitudinal Speed:** The forward velocity of the observed vehicle, measured in meters per second (m/s).
3. **Current Lane Index:** The current lane occupied by the observed vehicle, using the same indexing scheme as the ego vehicle (i.e., 0, 1, or 2).
4. **Relative Lane Position:** A categorical description of the observed vehicle's lane in relation to the ego vehicle (i.e., "same lane," "left lane," or "right lane").
5. **Relative Longitudinal Distance:** The forward or backward distance, in meters (m), from the ego vehicle to the observed vehicle. This is typically measured from the rear bumper of a leading vehicle to the front bumper of the ego vehicle, or vice-versa for a trailing vehicle.
6. **Relative Positional:** A categorical representation indicating whether the observed vehicle is positioned ahead or behind the ego vehicle. Specifically, it is defined as 'front' or 'rear' based on longitudinal distance.

4.4.3 Action Space (LLM Decision Outputs)

Based on the processed observational input, the LLM determines the ego vehicle's tactical maneuvers. The decision outputs primarily cover three key aspects of driving behavior:

1. **Adaptive Cruise Control (ACC) Target Speed**: The desired cruising speed, in m/s, for the ACC system to maintain, subject to prevailing traffic conditions and safety constraints.
2. **ACC Target Time Gap**: The desired time gap, in seconds (s), for the ACC system to maintain with a preceding vehicle.
3. **Lane Change Maneuver:** A categorical decision regarding lateral movement:
    - Maintain current lane.
    - Initiate a lane change to the left.
    - Initiate a lane change to the right.

4.4.4 Episode Termination Conditions

Each simulation run, or episode, concludes based on one of the following conditions:

1. **Successful Target Achievement:** The episode ends successfully if the ego vehicle safely reaches the "exit" segment, which is the pre-defined destination that the last 200 m of the highway segment is marked as the "exit" segment.

Numerically, to successfully reach the destination, the traveled distance of the ego-truck should be larger than 4700, as the whole highway segment is of length 5200 m, and the ego-truck starts from 250 m.

2. **Safety Violation (Hazardous Event):** The episode is terminated prematurely if a critical safety event occurs, including:
   - **Collision:** Any physical impact between the ego vehicle and another vehicle or a static road element.
   - **Road Departure:** The ego vehicle deviates from the designated drivable roadway boundaries.
3. **Failure to Reach Target (Timeout):** The episode is terminated if the ego vehicle fails to reach the target destination within a predefined maximum number of simulation steps. This condition typically indicates overly cautious or inefficient driving behavior (e.g., excessively low driving speed).

4.4.5 Performance Evaluation Metrics

The comprehensive abilities of the fine-tuned LLMs are assessed using a suite of quantitative metrics, typically averaged over a statistically significant number of simulation episodes:

1. **Average Speed:** The mean longitudinal speed (m/s) of the ego vehicle over the duration of an episode. This metric reflects travel efficiency.
2. **Average Distance Traveled:** The mean distance (m) traveled by the ego vehicle per episode. This is particularly relevant for episodes that terminate prematurely.
3. **Average Executed Steps:** The mean number of simulation decision cycles completed by the ego vehicle per episode, indicating the efficiency in decision-making.
4. **Success Rate:** The proportion of episodes in which the ego vehicle successfully reaches the target destination, expressed as a percentage.

$$\text{Success Rate} = \left(\frac{\text{Number of Success Episodes}}{\text{Total Number of Episodes}} \right) \tag{4.3}$$

5. **Collision Rate:** The proportion of episodes that end due to a collision, defined as:

$$\text{Collision Rate} = \left(\frac{\text{Number of Collisions}}{\text{Total Number of Episodes}} \right) \tag{4.4}$$

6. **Off-Road Rate:** The proportion of episodes that terminate due to the ego vehicle leaving the drivable area, calculated as:

$$\text{Off-Road Rate} = \left(\frac{\text{Number of Off-Road Incidents}}{\text{Total Number of Episodes}} \right), \tag{4.5}$$

7. **Timeout Rate:** The proportion of episodes that exceed the maximum allowed number of steps (set to 500), calculated as:

$$\text{Timeout Rate} = \left(\frac{\text{Number of Timeouts}}{\text{Total Number of Episodes}} \right) \tag{4.6}$$

8. **Average TCOP (Total Cost of Operation):** A composite metric comprising energy and driver costs, representing the operational expenditure in euros [9]. The cost at each step is computed as:

$$c(t) = C_{el}\, e_t + C_{dr}\, \Delta t \tag{4.7}$$

Where C_{el} is the electricity cost rate, e_t is the energy consumed at time step t, C_{dr} is the driver cost rate, and Δt is the duration of a time step. The energy consumed (e_t) is given by,

$$e_t = f_t\, v_t\, \Delta t \tag{4.8}$$

with force f_t at time step t defined as,

$$f_t = m\, a_t + \tfrac{1}{2}\, C_d A_f \rho_{\text{air}}\, v_t^2 + mgC_r + mg\, \sin\Big(\arctan\Big(\tfrac{\text{slope}}{100}\Big)\Big) \tag{4.9}$$

where:

- m: vehicle mass
- a_t: vehicle acceleration at time t
- C_d: aerodynamic drag coefficient
- A_f: frontal area
- ρ_{air}: air density
- C_r: rolling resistance coefficient
- g: gravitational acceleration

In this study, the slope is assumed to be zero. The detailed parameter values are provided in Table 4.4.

Table 4.4: Parameters Used in TCOP Calculation

Parameters	**Values**
m	40000 kg
C_d	0.36
A_f	10 m^2
ρ_{air}	1.225 kg/m^3
C_r	0.005
g	9.81 m/s^2
C_{el}	0.5 euro per kwh
C_{dr}	50 euro per hour
Δt	1 s

5

Results

In the results chapter, we illustrate the outcome of the conducted experiments. Beginning with the model performances, we compare the reasoning and generalization capabilities of the three models before and after the fine-tuning to determine whether the fine-tuning resulted in a modification of the model's capabilities. Afterwards, we evaluate our fine-tuned models in the SUMO simulation environment, as quantified by custom metrics specifically crafted for this study. Finally, we show the actual experiment scenarios to demonstrate the model's output.

5.1 Evaluation of Generalization Abilities

This section shows how the generalization ability of the three models changes before and after fine-tuning. The models will be evaluated on two generic datasets. The datasets include MMLU and HellaSwag, which evaluate the models' reasoning ability and common-sense contextual understanding, respectively.

The evaluation accuracy scores for each LLM are presented in Figure 5.1. We let each of the three models answer the questions in the two datasets. What is clearly seen is that Qwen2.5 14B exceed the other two models by a cliff on both datasets, especially on HellaSwag. From this we can tentatively infer that the Qwen series of models may be more suitable for making correct and rational decisions in our SUMO environment, as the decision to drive an automated truck in a SUMO environment is a complex

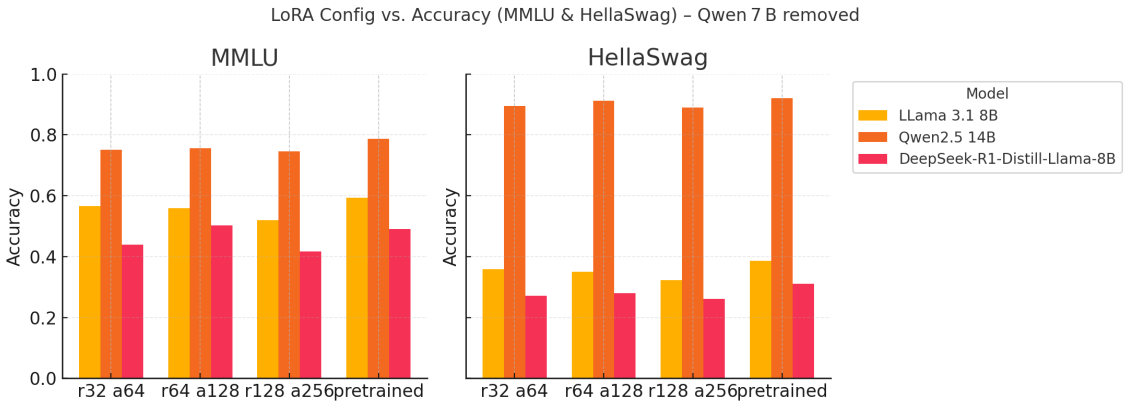

Figure 5.1: Accuracy of MMLU and HellaSwag dataset for each LLM.

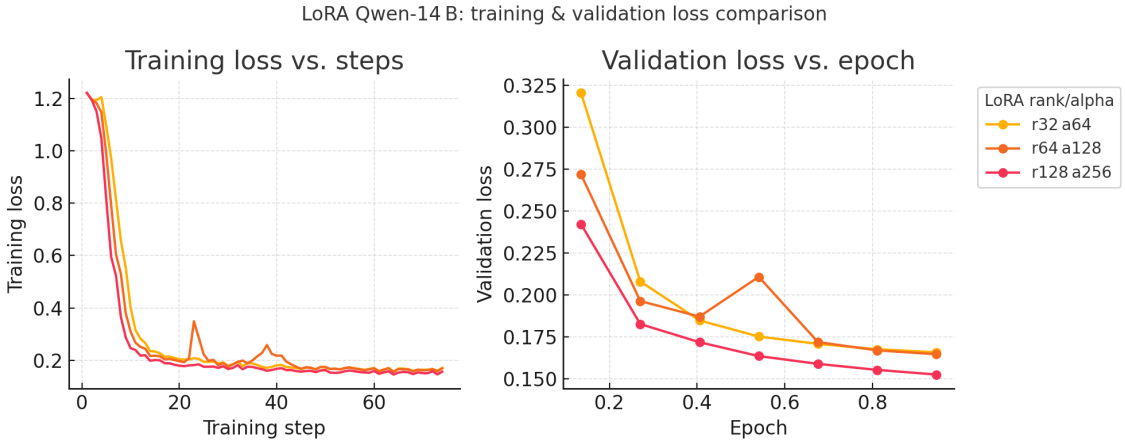

Figure 5.2: Loss change comparison of Qwen2.5 14B

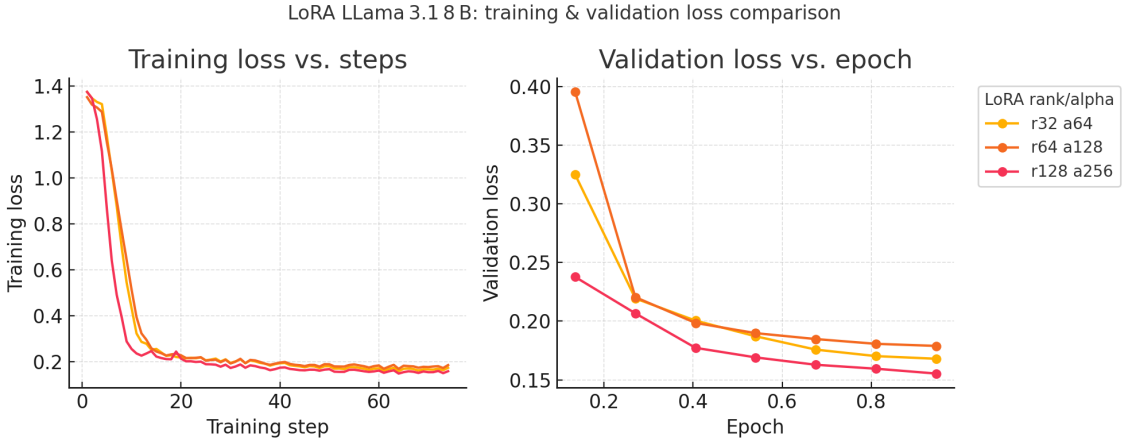

Figure 5.3: Loss change comparison of Llama 3.1 8B

and continuous decision-making process. We can also see that almost all models have their performance on both datasets more or less degraded after fine-tuning. This is explainable because the two datasets are more generalized datasets, whereas the fine-tuned datasets are synthetic, with strong specialized domain knowledge. While we have frozen most of the model parameters as a way to mitigate the problem of catastrophic model forgetting, the model still suffers above its ability to generalize. Among all three models and two datasets, the worst relative drop occurs on LLama 3.1 8B evaluated on HellaSwag after fine-tuning with rank 128 and α 256, where accuracy falls by approximately 16 % compared with its pre-trained model.

Looking at the changes in the fine-tuning hyperparameters, the scores of most of the models on these two datasets tend to increase and then decrease as the values of α and r increase. One explanation for this is that the increase in parameters makes the model learn the logic in the training set, but as the number of parameters continues to increase, the amount of information contained in the dataset is all but exhausted, and the model spends its attention on the training set, leading to catastrophic forgetting of the general knowledge.

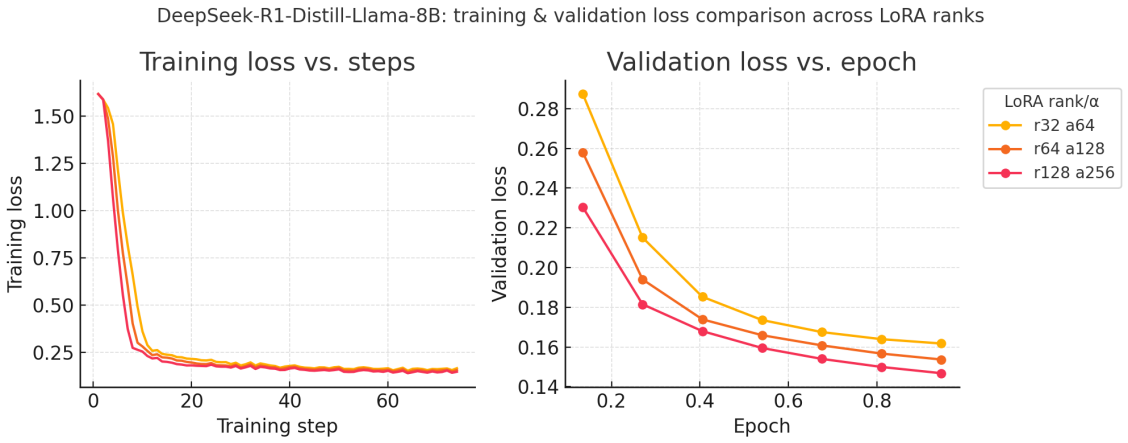

Figure 5.4: Loss change comparison in training and validation

Figure 5.2 shows the loss changes during the training and validation process for Qwen2.5 14B. We can see that as the rank and alpha parameters increase, the model converges faster during training and eventually reaches lower loss values. This is easy to understand because the model's fitting ability gets stronger with more parameters, and we validate their loss values on the validation set at the same time to prevent overfitting. Figure 5.3 and 5.4 shows the loss curves for the Llama and DeepSeek-R1-Distill-Llama models during training, again demonstrating the same pattern as the Qwen2.5 14 model, further validating the previous observation.

5.2 Evaluation of Tactical Decision Making Abilities

This section analyzes the performance of the proposed three large language models for decision making with the SUMO simulation environment, both before and after fine-tuning. To evaluate the performance of the model more comprehensively, including dealing with common driving situations (the common case of three surrounding vehicles in the training set), as well as the more intensive case of seven surrounding vehicles, we set up two separate sets of experiments for this, and calculate the corresponding metrics. In addition, this section also analyzes the reasonableness of the model's decisions in specific driving scenarios through case studies.

5.2.1 Metrics Evaluation

The calculated metrics provide the assessment of models' performance in terms of safety, efficiency, and operational cost.

Evaluation metrics, shown in Tables 5.1 (pretrained Models) and 5.2 (fine-tuned Models) are from experiments with three surrounding vehicles. And the results for the seven surrounding vehicles experiments are shown as Tables 5.3 (pretrained Models) and 5.4 (fine-tuned Models).

Metric	Llama3.1-8B	Qwen2.5-14B	Distill-Llama-8B
Success Rate	0.88	0.97	0.02
Failure Rate	0.12	0.03	0.98
Max Steps Rate	0.00	0.00	0.00
Invalid Decision Rate	0.09	0.00	0.00
Average Distance (m)	4538.82	4613.82	1100.65
Average Speed (m/s)	19.62	23.65	21.61
Average Steps	235.38	195.87	50.42
Energy Cost	2.87	3.56	1.63
Driver Cost	2.99	2.72	0.73
TCOP	5.86	6.28	2.36
TCPO/km	1.42	1.49	3.44

Table 5.1: Performance metrics of pretrained models in three surrounding cars experiments

Metric	Llama3.1-8B	Qwen2.5-14B	Distill-Llama-8B
Success Rate	1.00	1.00	0.66
Failure Rate	0.00	0.00	0.34
Max Steps Rate	0.00	0.00	0.00
Invalid Decision Rate	0.00	0.00	0.00
Average Distance (m)	4759.40	4766.59	4001.87
Average Speed (m/s)	21.08	24.00	22.28
Average Steps	229.70	198.87	178.28
Energy Cost	3.06	3.71	2.88
Driver Cost	3.19	2.76	2.51
TCOP	6.25	6.47	5.39
TCPO/km	1.31	1.36	1.38

Table 5.2: Performance metrics of fine-tuned models in three surrounding cars experiments

In general, fine-tuning has improved the success rate of all models. Shown by the results, the models are capable of generating required format responses simply by prompt engineering, even though LLama3.1-8b has some cases of invalid decisions in the pretrained version, it can be eliminated by fine-tuning, and the model will generate 100% responses that align with the template in the prompt. The elimination of invalid decisions for Meta-Llama-3.1-8B, in particular, underscores improved logical consistency and decision safety, which is a key prerequisite for real-world deployment. In addition, we investigate the abnormal responses, and they are all of long redundant text in the "reason" part. Thus they exceed the maximum token requirement and become invalid. This is common in small models that simply keep repeating meaningless words to meet the maximum token requirement.

In three surrounding vehicles experiments, especially for poor-performanced model Deepseek-distill-llama-8B, the success rate has been increased to 66% compared to 2% for its pretrained version. While fine-tuning has improved Llama-8B's success

Metric	Llama3.1-8B	Qwen2.5-14B	Distill-Llama-8B
Success Rate	0.66	0.90	0.48
Failure Rate	0.34	0.10	0.52
Max Steps Rate	0.00	0.00	0.00
Invalid Decision Rate	0.01	0.00	0.00
Average Distance (m)	3796.73	4585.20	3267.25
Average Speed (m/s)	22.27	21.95	23.75
Average Steps	172.93	212.18	137.09
Energy Cost	3.02	3.16	2.92
Driver Cost	2.36	2.95	1.92
TCOP	5.39	6.11	4.85
TCPO/km	1.86	1.34	2.76

Table 5.3: Performance metrics of pretrained models in seven surrounding cars experiments

Metric	Llama3.1-8B	Qwen2.5-14B	Distill-Llama-8B
Success Rate	1.00	1.00	0.42
Failure Rate	0.00	0.00	0.58
Max Steps Rate	0.00	0.00	0.00
Invalid Decision Rate	0.00	0.00	0.00
Average Distance (m)	4762.96	4760.08	3344.73
Average Speed (m/s)	23.01	21.57	23.53
Average Steps	209.18	225.46	141.10
Energy Cost	3.48	3.14	2.91
Driver Cost	2.91	3.13	1.98
TCOP	6.39	6.27	4.89
TCPO/km	1.34	1.32	1.89

Table 5.4: Performance metrics of fine-tuned models in seven surrounding cars experiments

rate both in three and seven vehicle experiments, Qwen2.5-14B consistently maintained a high success rate, demonstrating high robustness both before and after fine-tuning in all experiments. This is also reflected in the average experiment steps that are taken to carry out the experiment. Note that they are not the lower the better, because if the model fails the experiment and leads to an early termination, the steps will naturally be fewer than those ones taken to accomplish the experiment successfully. To provide a standard benchmark, it takes 189 steps to make the ego-truck reach the "exit" segment with the constant maximum speed of 25 m/s.

Note that although overall the results show a relatively high success rate after fine-tuning, much of this performance gain can be attributed to the architectural design of the controller system. In this setup, the LLM functions primarily as a high-level tactical decision-maker, responsible for understanding the driving context and issuing appropriate commands, while the low-level executor handles the actual vehicle control tasks such as lane changes, speed adjustments, and maintaining safe follow-

ing distances. The improved success rate thus reflects the benefits of this hierarchical control design. But still, it also indicates that the fine-tuned LLM has learned to understand a more complex and dynamic driving environment. It demonstrates enhanced decision-making capabilities, such as avoiding the risk of collisions associated with abrupt lane change maneuvers. Most importantly, the LLM has learned to avoid critical safety violations like driving out of the road, further emphasizing its improved contextual awareness and planning ability.

To assess efficiency, we also record and calculate the average driving speed (m/s) and Total Cost of Operation (TCOP) throughout all experiments. Due to the same concerns we mentioned before, a lower TCOP does not directly indicate that the model provides decisions that make driving more efficient. Therefore, we also record the Total Cost per kilometer (TCOP/km), which offers a normalized mark for the efficiency assessment. For example, in the three surrounding vehicle experiments, while Meta-Llama-3.1-8Bs TCPO/km decreases from 1.42 to 1.31, and Qwen2.5-14B drops from 1.49 to 1.36. Notably, deepseek-distill-llama's TCPO/km declined dramatically from 3.44 to 1.38, indicating that although its total costs increased (from 2.36 to 5.39), its operational efficiency per kilometer greatly improved. The same patterns are observed in the seven-vehicles experiments. Energy and driver costs followed the same trend as TCOP. And these increases are expected, given the models greater success in completing longer and more complex driving tasks.

Average speed increased modestly across all models throughout both three-surrounding-vehicle and seven-surrounding-vehicle experiments. These gains, considered alongside changes in average distance and average steps, reflect improvements in navigation efficiency and suggest more streamlined paths. Notably, although Qwen2.5-14B shows a relatively higher average speed in the three-surrounding-vehicle experiment, it adopts a more conservative strategy when driving conditions are more intensive. In contrast, Llama-8b demonstrates significant progress, with both average distance and average driving speed all increased in these two sets of experiments, which is consistent with its improved success rate after fine-tuning.

In conclusion, fine-tuning has enhanced the performance of all three models to some extent, especially in terms of task efficiency and decision reliability. Although overall operational costs increase, largely due to higher task completion rates, the cost efficiency per kilometer improves or remains stable. Qwen2.5-14B shows the most robust performance, with a stable experiment success rate and relatively low TCOP/km. These results indicate the potential for fine-tuning in adapting large language models for complex, real-world autonomous driving. Future work could explore further optimization strategies to enhance the balance between task success and cost-efficiency.

5.2.2 Case Studies

To further explore the reasoning capabilities and explainability of the models' decisions, this section presents detailed case studies observed during the experiments. These examples highlight the models' responses to specific driving scenarios both before and after the fine-tuning process, offering a qualitative understanding of the

changes in their decision-making behavior. It indicates the potential of models to analyze text-based driving situations and decision-making ability, as well as the ability to provide human-like language explainability.

5.2.3 Case Study 1: Deepseek-distill-llama's Lane Change Behavior

This case study focuses on the lane change decisions made by the Deepseek-distill-llama model in a specific traffic scenario before and after fine-tuning.

5.2.3.1 Pretrained Deepseek-distill-llama

The current driving scenario is shown as Table 5.5, and the corresponding decision made by the model is shown in Table 5.6.

Table 5.5: Encoded Current Driving Environment Observation

Ego-truck Status		
Current Speed (m/s) 24.0	Current Lane Number 0	Current Time Gap (s) 3.0
Surrounding Vehicles Information		
Car 1	Current Speed: 20.9 m/s Current Lane Number: 2 Relative Distance to Ego Vehicle: 130.9 m Position to Ego Vehicle: Front Lane Relation: Left lane	
Car 2	Current Speed: 20.1 m/s Current Lane Number: 2 Relative Distance to Ego Vehicle: 181.0 m Position to Ego Vehicle: Front Lane Relation: Left lane	
Car 3	Current Speed: 22.1 m/s Current Lane Number: 0 Relative Distance to Ego Vehicle: -45.3 m Position to Ego Vehicle: Rear Lane Relation: Same lane	

Table 5.6: Pretrained Deepseek-Distill-Llama Decision Based on Observed Context

Decision Component	**Value**
Desired Speed (m/s)	25
Desired Time Gap (s)	3
Lane Change Intent	Right
Reason	Adjusting to maintain safety and efficiency.

In the pretrained state, deepseek-distill-llama, despite the ego vehicle traveling at 24.0 m/s in lane 0 only with a following vehicle ('car3') in the same lane, it decided to change the lane to the right. The stated reason, "Adjusting to maintain safety and efficiency," is quite general and lacks specific justification based on the observed environment. There is no immediate indication of a safety threat or an obvious efficiency gain from moving to the right, especially considering the presence of vehicles in the left lanes. Given the ego vehicle is currently driving in the right-most lane, this decision leads to "out of road" and fails the experiment. Coupled with the previously reported low success rate, it suggests a potential lack of nuanced understanding of the driving context in the pretrained model.

5.2.3.2 Fine-tuned Deepseek-distill-llama

The current driving scenario is shown as Table 5.7, and the corresponding decision made by the model is shown in Table 5.8.

Table 5.7: Encoded Current Driving Environment Observation

Ego-truck Status		
Current Speed (m/s) 21.3	Current Lane Number 1	Current Time Gap (s) 3.0
Surrounding Vehicles Information		
Car 1	Current Speed: 18.6 m/s Current Lane Number: 2 Relative Distance to Ego Vehicle: -129.9 m Position to Ego Vehicle: Rear Lane Relation: Left lane	
Car 2	Current Speed: 19.2 m/s Current Lane Number: 2 Relative Distance to Ego Vehicle: 185.0 m Position to Ego Vehicle: Front Lane Relation: Left lane	
Car 3	Current Speed: 21.7 m/s Current Lane Number: 1 Relative Distance to Ego Vehicle: 99.9 m Position to Ego Vehicle: Front Lane Relation: Same lane	

Table 5.8: Finue-tuned Deepseel-Distill-Llama Decision Based on Observed Context

Decision Component	**Value**
Desired Speed (m/s)	25
Desired Time Gap (s)	3
Lane Change Intent	Right
Reason	Move to right lane to maintain faster speed in clear lane with no immediate obstructions ahead.

In this case, the ego truck is in lane 1, driving at 21.3 m/s. There is a slower vehicle ('car1' at 18.6 m/s) behind in the left lane (lane 2) and a slightly faster vehicle ('car3' at 21.7 m/s) ahead in the same lane. The model identifies the right lane (lane 0, implicitly) as a "clear lane with no immediate obstructions ahead" and aims to "maintain faster speed." This reasoning aligns with typical driving strategies for efficiency and maintaining desired speed. The improved explainability suggests that fine-tuning has enhanced the model's ability to perceive the environment and provide more contextually relevant justifications for its decisions.

5.2.4 Case Study 2: Qwen2.5-14B's Safety Maintenance Behavior

This case study focuses on the lane change decisions made by the Qwen2.5-14B model in a specific traffic scenario before and after fine-tuning.

The current driving state of the ego-truck and the surrounding vehicles information are shown as Table 5.9.

Table 5.9: Encoded Current Driving Environment Observation

Ego-truck Status		
Current Speed (m/s)	Current Lane Number	Current Time Gap (s)
25.0	2	2.0
Surrounding Vehicles Information		
Car 1	Current Speed: 20.2 m/s Current Lane Number: 2 Relative Distance to Ego Vehicle: 20.0 m Position to Ego Vehicle: Front Lane Relation: Same lane	
Car 2	Current Speed: 24.8 m/s Current Lane Number: 1 Relative Distance to Ego Vehicle: 57.1 m Position to Ego Vehicle: Front Lane Relation: Right lane	
Car 3	Current Speed: 25.5 m/s Current Lane Number: 1 Relative Distance to Ego Vehicle: -5.7 m Position to Ego Vehicle: Rear Lane Relation: Right lane	
Car 4	Current Speed: 25.5 m/s Current Lane Number: 0 Relative Distance to Ego Vehicle: -56.7 m Position to Ego Vehicle: Rear Lane Relation: Right lane	
Car 5	Current Speed: 23.5 m/s Current Lane Number: 1 Relative Distance to Ego Vehicle: 84.7 m Position to Ego Vehicle: Front Lane Relation: Right lane	
Car 6	Current Speed: 25.5 m/s Current Lane Number: 2 Relative Distance to Ego Vehicle: -40.5 m Position to Ego Vehicle: Rear Lane Relation: Same lane	

5.2.4.1 Pretrained Qwen-14B

The corresponding decision made by the pretrained model is shown in Table 5.10.

Table 5.10: Pretrained Qwen-14B Decision Based on Observed Context

Decision Component	Value
Desired Speed (m/s)	20.2
Desired Time Gap (s)	2.0
Lane Change Intent	Right
Reason	Reduce speed to maintain safe distance from front vehicle and change lane to avoid rear vehicle.

The pretrained Qwen-14B model is able to recognize the speed difference with the lead vehicle ('car1' at 20.2 m/s), and it detects the potential threat from the rear vehicle ('car6' at 25.5 m/s). In response, it attempts to handle this situation by both slowing down and changing lanes to the right, presumably to avoid the pressure from Car 6. However, this decision reflects an incomplete understanding of the surrounding traffic, and the model fails to capture the threat that comes from Car 3 (with speed 25.5 m/s) in the right lane (lane 1) with a very short distance, and this could lead to a potential collision, which is very dangerous.

5.2.4.2 Fine-tuned Qwen14B

The corresponding decision made by the fine-tuned model is shown in Table 5.11.

Table 5.11: Fine-tuned Qwen-14B Decision Based on Observed Context

Decision Component	Value
Desired Speed (m/s)	20.2
Desired Time Gap (s)	2.0
Lane Change Intent	None
Reason	Maintain speed to match slower front vehicle in same lane while avoiding rear collision risk from veh6.

After fine-tuning, the model demonstrates a more conservative and context-aware behavior. As it identifies that the leading vehicle ('car1' at 20.2 m/s) is driving with a slow speed and it is aware of the potential danger comes from the rear fast-speed car ('car6' at 25.5 m/s). More importantly, the model now captures the critical risk posed by Car 3 (with speed 25.5 m/s) on the right (lane 1). It shows that the model takes into account the dynamic traffic in the right lane, and it chooses to maintain a safe, slower speed that matches the leading vehicle, and stay in its current lane instead of making a potentially risky lane change to the right. This suggests improved safety maintenance behavior and a better understanding of surrounding traffic dynamics.

6

Conclusion

This thesis investigates the application of LLMs in the domain of tactical decision-making for autonomous driving trucks. Traditional decision-making approaches, typically rule-based, often fall short when faced with the complexity, uncertainty, and socially nuanced nature of real-world traffic scenarios. In contrast, LLMs can offer a promising alternative due to their contextual reasoning and flexibility.

This work focuses on using small, pretrained LLMs, given practical considerations such as computational efficiency and the need for deployment on resource-limited systems. Through fine-tuning on domain-specific datasets, these models exhibit the ability to generate contextually appropriate tactical maneuvers, including lane changes, efficient driving speed adjustments, together with reasonable justification on the basis of analyzing the current driving environment.

The fine-tuning process allows the LLM to learn from a diverse set of traffic contexts, encompassing interaction dynamics and traffic densities. One of the key contributions of this research is that it demonstrates LLMs' ability to effectively process text-based representations of complex driving environments and generate maneuver decisions that are both operationally feasible and safety-aligned, showing the potential of LLMs as high-level tactical planners in autonomous driving systems. In addition, this thesis proposes and implements a modular control architecture that separates high-level planning from low-level execution. Within this framework, the LLM operates as a strategic decision-maker, while a dedicated low-level controller handles actual acceleration/deceleration calculation. Thus the ego truck can maintain a safe distance from the leading vehicle.

Beyond decision-making capabilities, another significant benefit is that LLMs can generate human-like natural language explanations. This opens up possibilities for improving the transparency and explainability of the autonomous driving system in the future. By articulating the reasoning behind decisions, LLMs support interpretability, build user trust, and help meet the increasing regulatory requirements for explainable AI in safety-critical applications.

6.1 Future Work

While this thesis demonstrates the potential of LLMs in tactical decision-making, several limitations and research opportunities remain.

A primary challenge is hallucination, where LLMs generate plausible but incorrect or unsafe outputs. It is worth mentioning that, based on the experiments' observations, the models are not always able to make intelligent and accurate decisions. Future research must prioritize strategies to detect and mitigate hallucinations, potentially through uncertainty estimation, rule-based filters, or external sensory validation.

Another observation is that when exposed to sequential driving scenarios, especially in the highway driving environment, where there are many similar consecutive scenarios, while LLMs tend to make repeated and redundant decisions, in some specific cases, LLMs can produce consistent and smooth decisions with small changes between time steps, showing an implicit capability to maintain short-term temporal coherence. Future work may explore mechanisms that make LLMs retain relevant information from context over time while avoiding meaningless redundancy in the generated content.

Another promising direction is to integrate LLMs with other methods, such as reinforcement learning (RL) or the rule-based system, to form a hybrid decision-making system. In such a framework, the LLM can serve multiple roles:

- High-Level Goal Generation: The LLM generates abstract sub-goals (e.g., prepare for merge or maintain safe following distance) that guide the RL agents exploration and policy learning.
- Reward Shaping and Policy Guidance: The LLM provides structured language-based feedback or safety constraints, which can be translated into reward signals or training biases to guide the RL agent toward safe and efficient behaviors.
- Decision Filtering and Validation: Benefiting from its commonsense reasoning and understanding of natural language safety rules, the LLM can act as a reasoning-based filter, evaluating decisions proposed by itself or other modules, rejecting those that are unsafe, unreasonable, or non-compliant with traffic norms.

This hybrid approach improves both safety and explainability while also making the decision-making process more efficient and trustworthy.

Bibliography

[1] J. Hu, Y. Wang, S. Cheng, *et al.*, "A survey of decisionmaking and planning methods for selfdriving vehicles," *Frontiers in Neurorobotics*, vol. 19, p. 1 451 923, 2025. DOI: 10.3389/fnbot.2025.1451923. [Online]. Available: https://doi.org/10.3389/fnbot.2025.1451923.

[2] C. Zhao, L. Li, X. Pei, Z. Li, F.-Y. Wang, and X. Wu, "A comparative study of state-of-the-art driving strategies for autonomous vehicles," *Accident Analysis & Prevention*, vol. 150, p. 105 937, 2021.

[3] M. Pellkofer and E. Dickmanns, "Behavior decision in autonomous vehicles," in *Intelligent Vehicle Symposium, 2002. IEEE*, IEEE, vol. 2, 2002, pp. 495–500.

[4] S. Noh and K. An, "Decision-making framework for automated driving in highway environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 58–71, 2017.

[5] F. Bouchard, S. Sedwards, and K. Czarnecki, "A rule-based behaviour planner for autonomous driving," in *International Joint Conference on Rules and Reasoning*, Springer, 2022, pp. 263–279.

[6] J. Chen, *Decision-making in highway autonomous driving combined with prediction algorithms*, 2022.

[7] M. Bojarski, D. Del Testa, D. Dworakowski, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[8] J. Kim, T. Misu, Y.-T. Chen, A. Tawari, and J. Canny, "Grounding human-to-vehicle advice for self-driving vehicles," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 10 591–10 599.

[9] D. Pathare, L. Laine, and M. H. Chehreghani, "Improved tactical decision making and control architecture for autonomous truck in sumo using reinforcement learning," in *2023 IEEE International Conference on Big Data (BigData)*, IEEE, 2023, pp. 5321–5329.

[10] A. Rizehvandi, S. Azadi, and A. Eichberger, "Decision-making policy for autonomous vehicles on highways using deep reinforcement learning (drl) method," *Automation*, vol. 5, no. 4, p. 564, 2024.

[11] L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, pp. 681–694, 2020.

[12] Z. Yang, X. Jia, H. Li, and J. Yan, "Llm4drive: A survey of large language models for autonomous driving," *arXiv preprint arXiv:2311.01043*, 2023.

[13] H. Sha, Y. Mu, Y. Jiang, *et al.*, "Large language models as decision makers for autonomous driving," 2023.

[14] W. Wang, J. Xie, C. Hu, *et al.*, "Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving," *arXiv preprint arXiv:2312.09245*, 2023.

[15] T. Cai, Y. Liu, Z. Zhou, *et al.*, "Driving with regulation: Interpretable decision-making for autonomous vehicles with retrieval-augmented reasoning via llm," *arXiv preprint arXiv:2410.04759*, 2024.

[16] J. Wei, X. Wang, D. Schuurmans, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[17] Z. Xu, Y. Zhang, E. Xie, *et al.*, "Drivegpt4: Interpretable end-to-end autonomous driving via large language model," *IEEE Robotics and Automation Letters*, 2024.

[18] OpenAI, *GPT4o: Large language model*, `https://chat.openai.com`.

[19] Anthropic, *Claudeă3ăSonnet: Large language model*, `https://claude.ai`.

[20] K. He, R. Mao, Q. Lin, *et al.*, "A survey of large language models for healthcare: From data, technology, and applications to accountability and ethics," *Information Fusion*, p. 102 963, 2025.

[21] S. Petrus, *Base models are more based: Base models vs instruct models explained*, `https://sebastian-petrus.medium.com/base-models-are-more-based-base-models-vs-instruct-models-53609730f3e6`, Blog post, Sep. 2024. (visited on 05/07/2025).

[22] L. Huang, W. Yu, W. Ma, *et al.*, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, 2023," *arXiv preprint arXiv:2311.05232*, 2023.

[23] E. J. Hu, Y. Shen, P. Wallis, *et al.*, "Lora: Low-rank adaptation of large language models.," *ICLR*, vol. 1, no. 2, p. 3, 2022.

[24] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.

[25] J. Erdmann, "Sumo's lane-changing model," in *Modeling Mobility with Open Data*, ser. Lecture Notes in Computer Science, vol. 8594, Springer, 2015, pp. 105–123. DOI: `10.1007/978-3-319-15024-6_7`. [Online]. Available: `https://link.springer.com/chapter/10.1007/978-3-319-15024-6_7`.

[26] A. Liu, B. Feng, B. Xue, *et al.*, "Deepseek-v3 technical report," *arXiv preprint arXiv:2412.19437*, 2024.

[27] DeepSeek-AI, *Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning*, 2025. arXiv: `2501.12948 [cs.CL]`. [Online]. Available: `https://arxiv.org/abs/2501.12948`.

[28] D. Hendrycks, C. Burns, S. Basart, *et al.*, "Measuring massive multitask language understanding," *arXiv preprint arXiv:2009.03300*, 2020.

[29] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "Hellaswag: Can a machine really finish your sentence?" *arXiv preprint arXiv:1905.07830*, 2019.

[30] S. KrauSS, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Physical Review E*, vol. 55, no. 5, p. 5597, 1997.

[31] J. Erdmann, "Lane-changing model in sumo," *Proceedings of the SUMO2014 modeling mobility with open data*, vol. 24, pp. 77–88, 2014.

A

Appendix 1 - System Prompt

A.1 Supervised Fine-Tuning Prompt

You are a tactical decision-making system for a semi-truck,

ego truck with the following parameters:
- Maximum speed: {} m/s
- Maximum acceleration: {} m/sš
- Maximum deceleration: {} m/sš
- Adaptive Cruise Control (ACC) time gap range: {} - {} seconds

Current situation:
- Current speed: {} m/s
- Current lane: {} (0=right, 1=middle, 2=left on a 3-lane highway)
- Current ACC time gap setting: {} seconds

Surrounding vehicles:
{ {
"Vehicle id": {}
"Current speed": {}
"Current lane": {}
"Relative lane relation to the ego vehicle": {}
"Relative distance to the ego vehicle": {}
"Relative position to the ego vehicle": {}
}}

Based on the situation, make a tactical decision with ONLY these outputs:
1. ACC set speed (m/s)
2. ACC time gap (s)
3. Lane change request (none, left, right)
4. Brief reason for decision (one sentence)

Format your response as a JSON object:
{{
"acc_set_speed": X,
"time_gap": Y,
"lane_change": "none/left/right",
"reason": "brief explanation"
}}

Do not include any other text outside of this JSON object.

B

Appendix 2 - Dataset

B.1 Example Scenario-Decision Pair

Scenario:

```
{
"ego_speed": 22.1,
"ego_lane": 1,
"current_time_gap": 3.1,
"surrounding_vehicles": [
{
"id": "veh2",
"distance": 42.2,
"rel_position": "rear",
"lane_relation": "same_lane",
"speed": 21.0,
"lane": 1
},
{
"id": "veh5",
"distance": 104.6,
"rel_position": "front",
"lane_relation": "right_lane",
"speed": 22.6,
"lane": 0
}
]
}
```

Decision:

```
{
"acc_set_speed": 22.6,
"time_gap": 3.8,
"lane_change": "right",
"reason": "Return to right lane to follow keep-right rule;
gap ahead is clear and safe."
}
```