VIETNAM GENERAL CONFEDERATION OF LABOR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY

NGUYEN TRUNG THANG
HO MINH CHI TAN

DEVELOPING QUESTION-ANSWERING SYSTEMS INTEGRATED WITH REINFORCEMENT LEARNING FOR ROAD TRAFFIC LAW

INFORMATION TECHNOLOGY PROJECT

SOFTWARE ENGINEERING

Supervisor
Ph.D. TRAN LUONG QUOC DAI

HO CHI MINH CITY, 2024

VIETNAM GENERAL CONFEDERATION OF LABOR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY

NGUYEN TRUNG THANG – 519H0231
HO MINH CHI TAN – 519H0044

DEVELOPING QUESTION-ANSWERING SYSTEMS INTEGRATED WITH REINFORCEMENT LEARNING FOR ROAD TRAFFIC LAW

INFORMATION TECHNOLOGY PROJECT

SOFTWARE ENGINEERING

Supervisor
Ph.D. TRAN LUONG QUOC DAI

HO CHI MINH CITY, 2024

ACKNOWLEDGMENT

First of all, I would like to sincerely thank the teachers of Ton Duc Thang University for their dedication to teaching and helping me complete my study program. The teachers are like silent ferrymen who impart to us valuable knowledge and especially practical experiences so that we can confidently walk on our path. Studying at this school is our luck and happiness. I am extremely grateful for this luck and wish the teachers good health to continue these silent ferry trips.

In particular, I would like to sincerely thank my teacher, Dr. Tran Luong Quoc Dai, for his dedicated guidance and instruction throughout the implementation process.

Due to my limited understanding, there were misunderstandings incorrect thoughts, and incorrect implementation, leading to many errors in the project. Therefore, I look forward to receiving sincere comments from the teachers and the defense council so that I can update, edit, and improve.

I sincerely thank you!

. Ho Chi Minh City, August 20th , 2024

Author

(Signature and full name)

PROJECT COMPLETED

AT TON DUC THANG UNIVERSITY

I hereby declare that this is my own research work and was scientifically guided by Dr. Luong Tran Quoc Dai. The research contents and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments and evaluation were collected by the author himself from various sources clearly stated in the reference section.

In addition, the Project also uses a number of comments, price assessments and data from other authors and organizations, all of which are cited and noted.

If any fraud is detected, I will take full responsibility for the content of my Project. Ton Duc Thang University is not related to any copyright violations caused by me during the implementation process (if any).

Ho Chi Minh City, August 20th , 2024

Author

(Signature and full name)

Nguyen Trung Thang

Ho Minh Chi Tan

DEVELOPING QUESTION-ANSWERING SYSTEMS INTEGRATED WITH REINFORCEMENT LEARNING FOR ROAD TRAFFIC LAW

ABSTRACT

This thesis focuses on developing a specialized chatbot in the field of road traffic law, applying the Reinforcement Learning from Human Feedback (RLHF) method to optimize the efficiency and accuracy of responses. The chatbot model is trained using advanced language models, such as Llama, combined with model compression techniques to reduce computational resources while maintaining high performance. The training process includes three main steps: collect and classify data from trusted legal sources, deploy a machine learning model with RLHF mechanism to adjust and improve the response based on human feedback Use and apply Sequence Classification technique to evaluate and improve response quality. Research results show that the chatbot not only provides accurate information about traffic laws but also continuously improves based on actual feedback from users, thereby confirming the effectiveness and potential application of the method. RLHF in developing smart public service systems.

TABLE OF CONTENT

TABLE OF CONTENT ..... **v**

LIST OF FIGURES ..... **viii**

LIST OF TABLES ..... **ix**

ABBREVIATIONS ..... **x**

CHAPTER 1. INTRODUCTION AND TOPIC OVERVIEW **2**

1.1 Reasons for choosing the topic 2

1.2 Document implementation objectives 3

1.3 Solution orientation 4

CHAPTER 2. THEORETICAL BASIS **5**

2.1 Overview of artificial intelligence, machine learning, deep learning 5

2.1.1 Concept *5*

2.1.2 Types of problems *6*

2.1.3 Overfitting and Underfitting *7*

2.1.4. Machine learning model formation process *8*

2.1.5 Challenges and benefits of AI *9*

2.2. Overview of RLHF (Reinforcement Learning with Human Feedback) 10

2.2.1. Procedure of RLHF (Reinforcement Learning with Human Feedback) *11*

2.2.2. Benefit of RLHF (Reinforcement Learning with Human Feedback) *11*

2.2.3. Introduction to reinforcement learning (RL) *11*

2.2.4. LLM (Large Langue Model) *12*

2.3. Overview of chatbot system 14

2.3.1. Concept *14*

2.3.2. Methods of building chatbot *15*

2.3.3. Question and answer system *16*

2.4. Platform Development 17

2.4.1. Pytorch *18*

2.4.2. TRL(Transformers Reinforcement Learning) *19*

2.4.3. Gradio *20*

2.4.4. Unsloth *21*

CHAPTER 3. PROPOSED MODEL 22

3.1. Functional analysis 22

3.2. Overall architecture of the chatbot RLHF 23

3.3. Construct dataset and model 25

3.3.1. Experimental data *25*

3.3.2. Pretraining Language Models (SeaLLM v3 1.5B) *27*

3.3.3. Reward model (Qwen2-0.5B) *33*

3.3.4. PPO model (Proximal Policy Optimization) *38*

3.4. Interface Design 43

CHAPTER 4. EXPERIMENTAL MODEL 44

4.1. Evaluate SFT model 44

4.2. Evaluate Reward model 46

4.3. Evaluate PPO model 48

CHAPTER 5. CONCLUSION 58

5.1. Result 58

5.2. Development 58

5.3. Limitations.....58

5.4. Summary.....59

REFERENCE 60

LIST OF FIGURES

Figure 2.1. Types of data joints that can be present in the model 7

Figure 2.2. Overview build chatbot 8

Figure 2.3. PyTorch libary.....18

Figure 3.1. Functions of chatbot.....22

Figure 3.2. Overall architecture chatbot using RL 24

Figure 3.3. Dataset 4,000 sample 26

Figure 3.4. Dataset with 1,900 sample 27

Figure 3.5. Base LLM to SFT model process 27

Figure 3.6. Comparison evaluate Sea-bench in 2 mode.....29

Figure 3.7. Lora Structure (Low-Rank Adaptation) 32

Figure 3.8. Reward Model training process.....34

Figure 3.9. Context length and search precision of model Qwen2 36

Figure 3.10. Process of training a Reinforcement Learning model 40

Figure 3.11. Interface design of chatbot using Gradient 43

LIST OF TABLES

Table 3.1. Model evaluation table in 5 languages 30

Table 3.2. Distribution of training and testing data 31

Table 3.3. Evaluation results of language models 35

Table 4.1. Experimental configuration for the SFT model 45

Table 4.2. Compare loss on the training and validation sets of the SFT model 45

Table 4.3. Testing results of the SFT model 46

Table 4.4. Experimental configuration for the Reward model 47

Table 4.5. Comparison table between BLEU score and Winner Rate for two models SFT and PPO 49

Table 4.6. Manual assessment table of RLHF 50

ABBREVIATIONS

AI	Artificial Intelligence
BLEU	Bilingual Evaluation Understudy
LoRA	Low-Rank Adaptation
LLM	Large Language Model
KL	Kullback-Leibler Divergence
MMLU	Massive Multitask Language Understanding
NLP	Natural Language Processing
PPO	Proximal Policy Optimization
RLHF	Reinforcement Learning with Human Feedback
SFT	Supervised Fine-tuned

CHAPTER 1. INTRODUCTION AND TOPIC OVERVIEW

This chapter will provide a general introduction to the project, including the issues, scope and goals that the document aims at. From there, We will propose directional solutions that I will implement in the process of building this question and answer system.

1.1 Reasons for choosing the topic

Nowadays, with the 4.0 industrial revolution, we have witnessed the remarkable development of science and technology. It has brought great benefits to our lives, along with it, convenience and opening up many opportunities to bring a more comfortable life. Artificial intelligence (AI) is one of the most popular technologies today, in fact, this technology has been researched since the mid-20th century, but at that time it was still considered an algorithm that only served simple problems or simplified problems. But with the present, through experience through continuous research and improvement, AI brings us from one surprise to another, almost difficult problems such as light memorial cannot replace humans, but AI has proven that nothing is impossible. The evidence is machine translation, recognition warning, or even virtualization is supported. Virtual assistants, also known as chatbots or automatic answering systems, are becoming more and more popular and familiar as they appear everywhere, at all times. Their nature can be understood as computer programs, a system that can interact with users in natural language, like a story between people. Users can command or request it to perform some simple tasks, such as "Turn on the alarm at 6am" or "What's the weather like today", .. This convenience has gradually replaced the old clubs that operate, are not effective. With such capabilities, chatbots can be widely applied in different jobs or tasks such as: Customer care, support providing service information, automating schedules. Recently, thanks to the breakthrough development of algorithms related to Deep learning along with the field of Natural Language Processing (NLP), chatbot systems are now able to perform more complex human requests (e.g. ChatGPT, PaLM, Bard,

...). Therefore, you can hope that it is possible to develop chatbots to assist humans in more complex task tools.

In the field of road traffic law, providing accurate and timely information to road users is crucial to ensure safety and legal education. A chatbot system can be developed to assist in this task. Chatbots can provide information on traffic regulations, warnings and laws quickly. Users can ask the chatbot about specific rules such as speed limits, safety equipment requirements or priority rules, can provide information on legal measures to take in case of an accident or traffic incident, think of ways to contact the authorities or basic first aid instructions, and more basic legal questions related to traffic violations, fines, and road users' rights. This helps users better understand their responsibilities and rights when participating in traffic, in addition it is also used to promote traffic safety, provide Tips and advice for safe driving, and remind traffic participants of the importance of discipline, more specifically it can provide multilingual support and operate 24/7, helping traffic participants access necessary information anytime, anywhere.

1.2 Document implementation objectives

As in the above section, we see that developing virtual assistant systems helps people learn about traffic laws, while helping to reduce the work of propaganda about road traffic for officials and authorities. However, we will have to carefully consider the directions to be able to build effectively. Building virtual assistant systems will always come with 2 main issues or aspects: technology used and development methods. Although technology is important, it is not a sufficient condition to create a good enough chatbot. If we focus on technology issues, but the chatbot is not well designed, it will produce a bad chatbot, and vice versa, if the chatbot is well designed but the technology does not meet the requirements, the chatbot cannot be used. Therefore, we need to research in the direction of combining both of these issues in the process of researching and building chatbots. Currently, NLP AI technologies have been strongly developed, typically the RLHF (Reinforcement Learning from Human Feedback) models were born, along with a series of research on algorithms

to enhance the quality of the Q&A system. Therefore, in terms of technology, it can be said that it has been partly solved, but the remaining aspect of application development methods has not been solved. This problem is how to apply and select existing technologies to build a highly applicable chatbot system to support people or officials in answering questions and difficulties when learning about traffic laws.

1.3 Solution orientation

To build a good chatbot, we must ensure that the chatbot is technologically integrated with the system to optimize it. With the diagram "Building an integrated response system with reinforcement learning for traffic laws". Therefore, with this method, the system chatbot can only respond in a dynamic form, meaning that because it can ask the necessary information from the user, in the form of a question and answer system, it will default to giving a response every time the user asks a question, without having to care whether the user's question has enough information to support or not. However, for our topic, defining documents to handle the above cases is almost impossible. Along with that, there is very little or no real-world data to train AI models to identify which text a user's question belongs to.

In addition to using the above-mentioned scenarios, recently researchers have been able to build chatbot systems in the form of conversations (conversations) without the need for scenarios, based on rules, such as LLM models like ChatGPT, etc. However, the disadvantage of this method is that it requires large enough LM models, along with the input length (input limit of the language language model, usually calculated by the number of tokens), to be able to develop applications in practice, it requires a very large amount of resources to be able to serve, so in terms of optimization and design, it is completely unsuitable.

So, what we need in a virtual assistant system to suit our problem is the ability to optimize, be compact but still ensure accuracy.

CHAPTER 2. THEORETICAL BASIS

2.1 Overview of artificial intelligence, machine learning, deep learning

2.1.1 Concept

Artificial intelligence (AI) is a field in computer science that studies and develops systems that can perform cognitive tasks typically associated with human intelligence, such as learning, creativity, and image recognition. Modern organizations collect large amounts of data from a variety of sources, including smart sensors, human-generated content, monitoring tools, and system logs. The goal of AI is to develop self-learning systems that can analyze and derive meaning from data, then apply this knowledge to solve new problems in a human-like manner. For example, AI technology can engage in natural conversations with humans, generate original images and text, and make decisions based on real-time input data. Organizations can integrate AI capabilities into their applications to optimize business processes, enhance customer experiences, and drive innovation.

Machine learning is a field of science that focuses on developing algorithms and statistical models that computer systems use to perform tasks based on pattern recognition and inference, without specific instructions. These systems apply machine learning algorithms to analyze large volumes of historical data and find patterns in the data. This allows the systems to make more accurate predictions based on given input data. For example, data scientists might develop a medical application to diagnose cancer from X-ray images by using millions of scans along with their corresponding diagnoses to train the model.

Deep learning is a branch of artificial intelligence (AI) that focuses on teaching computers to process data in a way that mimics the way the human brain works. This process is inspired by the way neurons in the brain work together to process and analyze information. Deep neural networks use artificial neurons, or nodes, to process information. Each artificial neuron performs mathematical calculations to analyze

data and solve complex problems. This deep learning approach has the potential to solve problems or automate tasks that would normally require human intelligence.

2.1.2 Types of problems

Currently, for general machine learning problems, there are two main types of problems that are often encountered: supervised and unsupervised learning. In addition, there are also semi-supervised learning and reinforcement learning.

Supervised learning has achieved many successes in practical applications. This method, also known as inductive learning in machine learning, is similar to the way humans learn from past experiences to improve their skills and solve real-world tasks. However, since machines do not have the ability to "experience", they must learn from data collected in the past. This data is used to train computers, helping them grasp patterns and apply these insights to solve real-world problems.

Unsupervised learning is a branch of machine learning that uses unlabeled data to analyze and identify data features. Unlike supervised learning, unsupervised learning does not focus on predicting specific outputs. Instead, the goal is to discover relationships in data and group data points based on input data alone. Supervised learning involves using labeled data to make predictions, while unsupervised learning does not require labeled data. Instead, it aims to analyze data to discover important features, often uncovering hidden subgroups or patterns in a data set that humans might not recognize.

Semi-supervised learning is an approach that falls between supervised learning (with all labeled data) and unsupervised learning (with unlabeled data). In semi-supervised learning, both labeled and unlabeled data are used, with the goal of exploiting the unlabeled data to better understand the overall structure of the data. This is useful in machine learning and data mining, especially when labeled data is scarce or expensive to collect. This method typically uses a small set of labeled data combined with a larger set of unlabeled data. The goal is to improve the ability to predict future unknown data, outperforming labeled data alone.

Reinforcement Learning (RL) is a field in machine learning that focuses on training agents to perform actions in an environment to maximize rewards. Unlike supervised and unsupervised learning, reinforcement learning does not rely on labeled data or specific data patterns. Instead, it uses the process of learning from interactions with the environment, in which the agent learns from the feedback and rewards it receives to improve its decisions over time.

2.1.3 Overfitting and Underfitting

When performing machine learning tasks, there are two common problems: overfitting and underfitting. These problems can lead to models that do not perform as expected or are even unusable in real-world applications. Overfitting occurs when a model fits the training data too well, resulting in poor performance on unseen data. Conversely, underfitting occurs when a model does not learn well enough from the data, resulting in poor performance on both training and test data. Causes of these problems can include the quality and quantity of the data, as well as the structure and complexity of model.

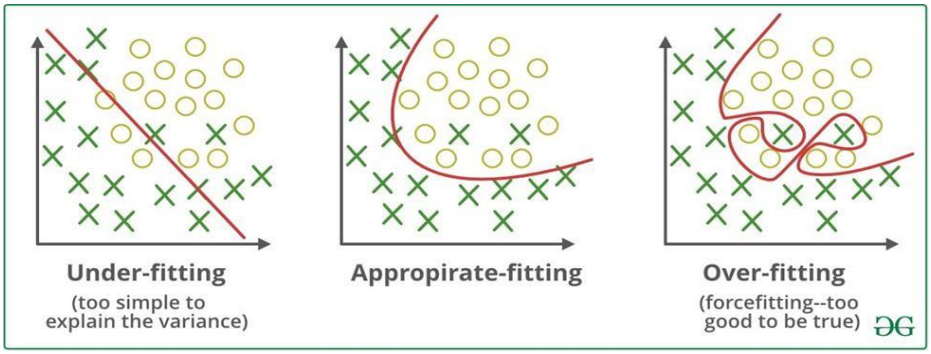

Figure 2.1. Types of data joints that can be present in the model

According *Figure 2.1* illustrates three common situations in machine learning: underfitting, appropriate fitting và overfitting.

Underfitting: This is a situation where the model is not complex enough to capture the underlying structure of the data, resulting in inaccurate predictions. In the figure, underfitting is represented by the red curve with the predicted value lacking accuracy.

Appropriate Fitting: This is the situation where the model fits the training data well and is able to predict accurately on new data. In the figure, appropriate fitting is shown by the red curve, with the predicted values being close to correct on new data.

Overfitting: This is a condition where the model overfits the training data, resulting in the model performing very well on the training data but poorly on new data. In this case, the model learns too much detail from the training data, reducing its ability to generalize. This usually happens when the model is too complex and tends to learn random details of the training data without being able to generalize to new data.

Conversely, underfitting occurs when the model is too simple to understand the structure of the data, resulting in poor performance and failing to account for data variability.

2.1.4. Machine learning model formation process

Developing an AI system is a complex and sometimes difficult process:

- **Data preparation:** Collect and process the data needed for model training.
- **Model building**: Design and structure the AI model to suit the project goals
- **Model training:** Use prepared data to train the model, helping it learn and improve its performance.
- **Product deyloyment**: Put the model into practice, deploy it to solve specific problems or integrate it into real-world applications.

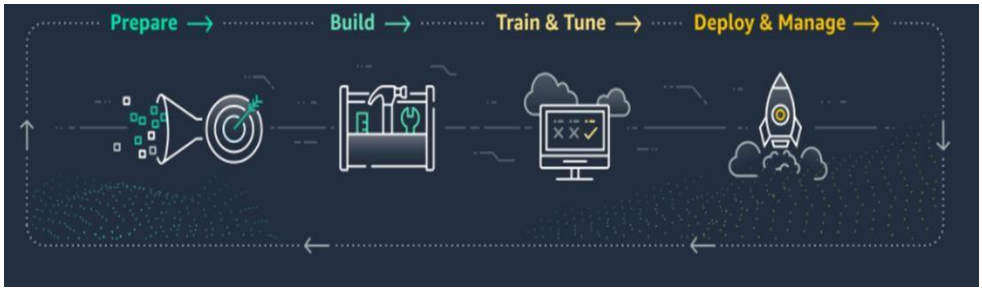

Figure 2.2. Overview build chatbot

The steps of developing an AI system will be repeated many times until the model achieves the desired performance. Only when the model is good enough can the product be deployed. However, as the application is widely used, more data and feedback from users will be generated, requiring continued improvement to maintain customer satisfaction. Therefore, the complexity of the development process also increases when the AI system includes many modules or sub-models. In addition, the development of an AI system depends on many factors such as hardware resources, project scale, and available data. These factors can affect the efficiency and progress of the development process.

2.1.5 Challenges and benefits of AI

The fourth industrial revolution has made technology smarter, smaller, lighter and more cost-effective. AI technologies include hardware such as physical robots, drones and autonomous vehicles, along with components such as processors, sensors, cameras and chips. In addition, there are software and source code such as data analytics, speech processing, biometrics, virtual reality, augmented reality, cloud technology, mobile technology, geotagging, low-code platforms, robotic process automation (RPA) and machine learning. AI is changing industries in many areas of services and automation as follows:

- In human resource management: AI helps optimize labor processes, improve productivity and work efficiency. According to a recent report, many large organizations have prioritized AI applications, with 70% of executives considering it a top priority. 2020 has seen an increase in the use of employee monitoring software. For example, IBM's "Maximo Worker" solution uses AI to analyze data from sources such as cameras, Bluetooth signals, mobile phones, IoT-connected wearables and environmental sensors, helping managers monitor more effectively and predict problems before they occur**.**

- In the enterprise: According to Alibaba, AI chatbots have helped reduce customer queries by up to 90% and serve more than 3.5 million users per day. A recent study of 1,500 companies across 12 industries found that performance improved significantly when humans and machines worked together.

- In healthcare: Regarding cancer detection, Wang et al. (2016) found that the combination of humans and AI outperformed humans or AI alone. The error rate dropped to 0.5% when both AI and humans were used in decision making, a reduction of at least 85% compared to methods using only humans or AI.

- In service and automation industries: Service robots are capable of analyzing large volumes of data, integrating information from multiple sources, and recognizing patterns related to customer profiles. They can propose appropriate solutions and make recommendations in a short time. Human-robot hybrid teams are increasingly taking on tasks that require high cognitive and emotional skills.

- In banking and finance: AI helps monitor data in real time, detect abnormal behaviors for investigation, and reduce payment fraud. Modern anti-fraud methods often struggle with sophisticated fraud, but AI applications enable rapid fraud detection and timely alert information**.**

2.2. Overview of RLHF (Reinforcement Learning with Human Feedback)

Reinforcement Learning with Human Feedback (RLHF) is a technique in machine learning that combines reinforcement learning (RL) with human feedback to optimize the behavior of AI models. Here is an overview of the RLHF model.

2.2.1. Procedure of RLHF (Reinforcement Learning with Human Feedback)

The RLHF process typically involves the following steps:

- Initial Training: The RL model is trained using traditional reinforcement learning methods. During this phase, the model learns from automatic rewards from the environment.
- Collecting Human Feedback: After initial training, feedback from humans is collected. This feedback can be incorporated into the training process to improve the model.
- Model Fine-Tuning: The model is fine-tuned based on human feedback. This feedback is used to adjust the agent's actions to achieve better results.
- Evaluation and Adjustment: The model is then evaluated based on actual performance and ongoing feedback from the user or the environment. Adjustments can be made to optimize the model.

2.2.2. Benefit of RLHF (Reinforcement Learning with Human Feedback)

- Improved quality: Human feedback helps improve the quality of agent performance, especially in problems where automatic rewards are insufficient.
- Error Reduction: Can help reduce errors and guide the agent to perform actions closer to the user's wishes.
- Better Generalization: Fine-tuning based on human feedback helps the model better generalize problems that were not found in the data trainer.

2.2.3. Introduction to reinforcement learning (RL)

Reinforcement learning is a field of machine learning in which an agent (agent) learns to act in an environment to maximize cumulative rewards over time. The learning process goes like this:

- Agent: An entity that takes actions in the environment to achieve a goal, the agent learns to maximize its cumulative reward through experimentation and experience.
- The environment responds with a reward or punishment based on the agent's actions.
- Actions: Decisions or steps that the agent takes in the environment. Actions can be discrete (such as specific choices) or continuous (such as changes in a range of numbers).
- Rewards: Signals from the environment that respond to the agent's actions. Rewards can be positive (encouraging behavior) or negative (discouraging avoidance of behavior).

2.2.4. LLM (Large Langue Model)

Large language models are a class of language models trained using deep learning techniques on vast text datasets. These models are capable of generating human-like text and performing various natural language processing tasks.

Language models can vary in complexity, ranging from simple n-gram models to highly complex neural network-based models that simulate human cognitive processes. However, the term "Large language model" typically refers to models that utilize deep learning techniques and possess a large number of parameters, often in the billions or even trillions. Such models can discern complex patterns in language and produce text that closely mimics human writing.

Large language models (LLMs) are advanced computational models designed using transformer architectures, which allow for the efficient processing and generation of text based on extensive datasets. Key aspects and characteristics of LLMs include:

- Scale: LLMs are distinguished by their substantial size, both in terms of the number of parameters, which can range from billions to trillions, and the volume of training data they are exposed to. This large scale enables them to capture and represent a wide range of linguistic features and knowledge.
- Pre-training and Fine-tuning: Typically, LLMs undergo a pre-training phase where they learn general language patterns from a broad and diverse corpus of text. Following this, they can be fine-tuned on specific datasets tailored to particular tasks, such as translation, summarization, or question-answering, allowing the models to specialize and improve performance in these areas.
- Transformer Architecture: The core architecture of most LLMs is the transformer, which utilizes mechanisms such as self-attention to efficiently handle large amounts of data and capture complex relationships within the text. This architecture facilitates the generation of coherent and contextually appropriate language outputs.

Applications: LLMs are versatile tools with applications across various fields, including:

- Natural Language Processing (NLP): For tasks such as machine translation, sentiment analysis, text classification, and named entity recognition.
- Natural Language Generation (NLG): To produce human-like text, which is useful in chatbots, automated writing systems, and creative writing tools.
- Conversational AI: In the development of interactive and intelligent virtual assistants and customer support systems.
- Information Retrieval and Summarization: For extracting and condensing relevant information from large volumes of text.

Ethical and Practical Considerations: The deployment of LLMs involves significant ethical and practical challenges, such as addressing issues related to bias, misinformation, and the environmental impact of training large models. Additionally, the implementation of these models requires substantial computational resources and careful consideration of privacy and security.

Prominent examples of LLMs include OpenAI's GPT (Generative Pre-trained Transformer) series, Google's BERT (Bidirectional Encoder Representations from Transformers), and the T5 (Text-To-Text Transfer Transformer) model. These models represent cutting-edge advancements in natural language processing and have become integral to modern AI research and applications.

2.3. Overview of chatbot system

2.3.1. Concept

Language is a primary tool for communication and information exchange in our daily lives. Whether we are working, conversing, or engaging in any form of communication, we use language. These exchanges are referred to as dialogues. A Dialogue System, or Conversational Agent, is a system capable of interacting with users through natural language. These systems provide an interface that allows users to interact using natural language, either in text or spoken form.

For text-based interaction, dialogue systems offer an interface similar to messaging applications (such as Facebook Messenger, Zalo, or Telegram), where users can send text messages and receive responses from the system. For spoken language, the system enables voice communication, capturing the user's spoken input, processing it, and responding accordingly. These systems, known as Spoken Dialogue Systems, include examples like Amazon Echo smart speakers, the Siri virtual assistant, and the robot Sophia.

Dialogue systems are generally categorized into two main types: Task-Oriented Dialogue Agents and Chatbots. Task-oriented dialogue systems are designed to help users accomplish specific tasks through conversation, such as finding directions, making phone calls, setting alarms, or checking the weather via virtual assistants like Siri, Alexa, Google Assistant, and Cortana. Additionally, these systems can answer user queries about a business or provide customer service support.

In contrast, chatbots are designed to engage in natural, human-like conversations, typically without a fixed structure and without the need for pre-defined rules. While chatbots are often used for entertainment purposes, they can also be integrated into task-oriented systems to make interactions more natural and user-friendly.

Today, the term “chatbot” is often used interchangeably with “dialogue system”, reflecting the broader range of tasks that chatbots can support and engage in natural conversations with users. For example, in the context of systems that advise citizens on traffic laws, the term “chatbot” is similarly used to describe a system that can support specific requests and converse in a more natural manner.

This overview has introduced the basic concepts and types of dialogue systems, highlighting that chatbots are a specific form of dialogue system. Subsequent sections will explore the methods used to build these systems, including chatbots, to understand their construction and functionality.

2.3.2. Methods of building chatbot

There are various methods for building chatbot systems, but they can generally be categorized into three main approaches:

- Rule-Based Chatbots: These chatbots rely on a set of predefined rules to match user inputs with appropriate responses. The rules are typically based on patterns or keywords, enabling the chatbot to provide consistent and predictable responses.

- Corpus-Based Chatbots: This approach utilizes a corpus of human-to-human conversations. Corpus-based chatbots use information retrieval techniques to find similar past interactions and replicate those responses, or they employ encoder-decoder models to generate new responses. This method relies heavily on the quality and diversity of the training data to produce natural and relevant conversations.
- Dialogue Systems: These systems are more advanced and are designed based on a dialogue state architecture. Dialogue systems track the context of the conversation, maintaining a coherent flow and adjusting responses based on the ongoing interaction. This approach often incorporates elements of machine learning and natural language processing to dynamically adapt to user inputs.

2.3.3. Question and answer system

Rule-based chatbots and dialogue systems are two prevalent approaches in chatbot development. Rule-based chatbots, though straightforward to implement, often struggle with efficiency and scalability due to their reliance on predefined rules. This approach can be limiting as it may not handle complex tasks or conversations that require multiple interactions to complete, such as booking train tickets, where the system needs to gather user information, trip details, booking preferences, and payment.

In contrast, dialogue systems are better suited for managing complex tasks that require iterative dialogue to gather all necessary information. However, they come with significant drawbacks, such as the complexity of implementation. Developing such systems requires extensive scenario-based scripting and data preparation, making them challenging to scale for real-world applications that need to offer a seamless user experience. Additionally, dialogue systems demand the definition of numerous potential scenarios, which can be labor-intensive and time-consuming.

Given the context of my project, which involves creating an automated question-answering system primarily for information retrieval, employing a full-fledged dialogue system is unnecessary. If such a system were used, it would require scripting for each specific question and topic, which is not practical. Instead, the project aligns more closely with the architecture of a corpus-based chatbot. This approach involves designing and refining modules such as question answering and information retrieval, which are adaptable to various systems by simply replacing the underlying dataset and associated modules.

However, corpus-based systems still lack the natural interaction capabilities that dialogue systems can offer, which limits the conversational fluidity. While advancements in large language models have mitigated some of these limitations, the issue persists unless extremely large models are used.

The architecture of question-answering systems typically includes key components like a question-answering module, a related data retrieval module, and a database. Some systems also incorporate additional preprocessing modules. There are also end-to-end systems, where a question is processed by a deep learning model to directly produce an answer without the need for intermediate steps.

2.4. Platform Development

Below are the main frameworks and libraries that I used to complete my project. While there are many smaller libraries and frameworks that could be utilized, I will focus on introducing the key ones

2.4.1. Pytorch

Figure 2.3. PyTorch libary

PyTorch is an open-source library widely used for building and training neural networks in the fields of Machine Learning and Deep Learning. Developed by Facebook's AI Research lab (FAIR), PyTorch offers a flexible and powerful approach for creating machine learning models, from simple to complex. Here are some key features of PyTorch:

- Dynamic Computational Graphs: PyTorch is known for its use of dynamic computational graphs, which allow for more flexibility in model creation and adjustment compared to other libraries. The graph is built on-the-fly as the Python code runs, making it easier to debug and modify models.
- Strong GPU Support: PyTorch provides robust support for GPU acceleration, significantly speeding up the training of neural networks and the processing of large datasets.
- Ease of Learning and Use: PyTorch features a Pythonic syntax that is intuitive and easy to understand, making it accessible to newcomers in Deep Learning. A large user community and extensive documentation also aid in learning and using PyTorch.
- TorchScript Module: TorchScript enables the conversion of PyTorch models into an intermediate representation that can be run in non-Python environments, facilitating deployment in various production settings.

- Large Community and Continuous Development: PyTorch benefits from a large user base and ongoing support and updates from Facebook and the open-source community. This continuous development helps improve the library, add new features, and address issues promptly.
- Support for Various Applications: While primarily used for Deep Learning, PyTorch also supports a wide range of applications, including Computer Vision, Natural Language Processing (NLP), Reinforcement Learning, and more.

PyTorch is a popular and powerful deep learning library, widely chosen by the community for its flexibility, performance, and ease of use.

2.4.2. TRL(Transformers Reinforcement Learning)

TRL (Transformers Reinforcement Learning) is a concept in machine learning that integrates two foundational technologies: Transformers and Reinforcement Learning (RL).

- Transformers: Transformers represent a highly effective neural network architecture, widely utilized in natural language processing (NLP). This architecture leverages an attention mechanism that enables the model to focus on the most relevant parts of the input data. Renowned models such as GPT and BERT are built upon the Transformer architecture.
- Reinforcement Learning (RL): Reinforcement Learning is a machine learning paradigm where an agent interacts with its environment to learn how to maximize a cumulative reward. Through a process of trial and error, the agent incrementally refines its actions to achieve the most favorable outcomes.
- TRL (Transformers Reinforcement Learning): TRL is the amalgamation of these two technologies. In this framework, a Transformer model can be fine-tuned or trained further using feedback from its environment via reinforcement learning techniques. This approach allows the model to go

beyond static data reliance, enabling it to adapt and improve continuously based on ongoing feedback from the environment.

- Applications of TRL:

  Fine-tuning Language Models: TRL can be applied to refine large language models to produce outputs that align more closely with specific requirements.

  Intelligent Dialogue Systems: TRL can be employed in the development of chatbots or dialogue systems that respond more effectively and adapt to novel situations.

  Output Optimization: TRL enables models to learn how to optimize their outputs based on criteria such as creativity, relevance, or accuracy.

2.4.3. Gradio

Gradio is an open-source library in Python designed to help users build interactive user interfaces (UI) for machine learning (ML) models quickly and easily. The library provides the tools needed to create intuitive and interactive applications for ML models, without requiring users to have in-depth knowledge of front-end development..

Here are some of the highlights of Gradio:

- Create interactive user interfaces for ML models: Gradio allows users to design simple interactive interfaces to interact with machine learning (ML) models with just a few lines of code. Users can create widgets to import data and display results directly from the model.
- Support for a wide range of ML models: Gradio is compatible with a wide range of ML models, including deep learning, traditional machine learning, and models in the fields of natural language processing (NLP), image processing, and tabular data.

- Compatible with many frameworks: Gradio works well with popular frameworks such as TensorFlow, PyTorch, Keras, Scikit-learn, and many others in the ML field.
- Easy integration and flexible customization: Gradio provides a simple API for integration and customization of the user interface. Users can customize the structure, input and output data types, and display settings to suit their specific needs.
- Gradio has become a useful tool for creating interactive, simple, and intuitive user interfaces for ML models, making these models more accessible and usable for users without in-depth knowledge of Machine Learning.

2.4.4. Unsloth

The uSloth library on Hugging Face is part of the Hugging Face ecosystem, focused on providing optimized models for machine learning applications. Here are some key details:

- Integration with Hugging Face: The uSloth library integrates with the Hugging Face platform, allowing users to easily download, deploy, and test machine learning models. It leverages Hugging Face tools and APIs to provide easy model access and deployment.
- Multiple Model Support: uSloth not only provides optimized models, but also allows testing with different versions of machine learning models, including versions of the Llama model.
- Supported Features: The library provides tools and resources to help users deploy models in real-world applications, including optimizing performance and reducing computational resource requirements.

CHAPTER 3. PROPOSED MODEL

3.1. Functional analysis

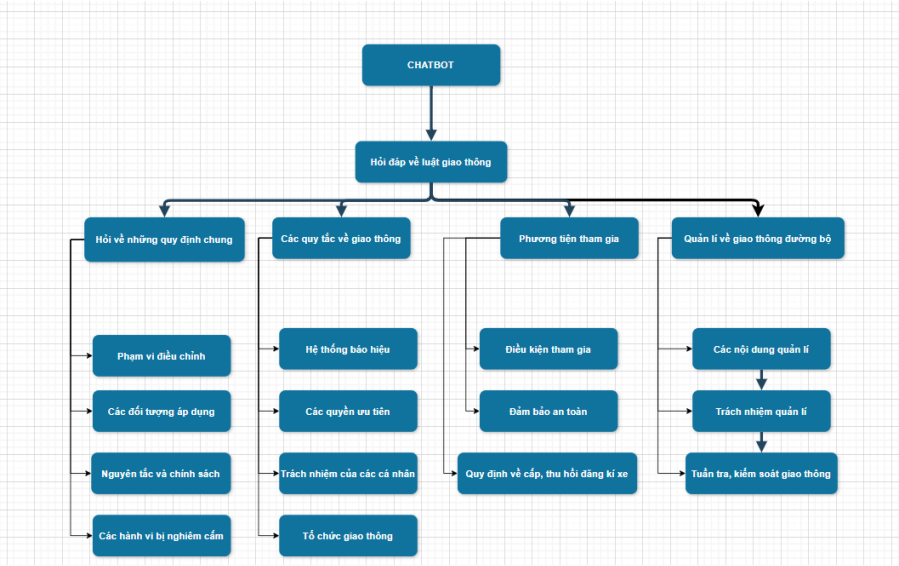

Figure 3.1. Functions of chatbot

Chatbots or automated response systems can greatly enhance user experience by addressing frequently asked questions about traffic laws. These questions often repeat across various users, making it essential to organize the inquiries into four main functional categories: general regulations, traffic rules, participating vehicles, and road traffic management.

General Regulations: In this category, the system provides users with essential information on the scope of traffic laws, the different types of entities to which these laws apply, as well as the fundamental principles and policies governing all participants. Additionally, it covers actions that are strictly prohibited under the law.

Traffic Rules: The second category focuses on assisting users in understanding the traffic signaling system, including traffic lights and road signs. It also clarifies the rules regarding the right of way for different participants and offers valuable insights into the responsibilities of individuals and organizations involved in traffic, detailing who is responsible and how.

Participating Vehicles: This section addresses inquiries about vehicle requirements for participating in traffic, ensuring compliance with road safety standards, and provides information on the issuance and revocation of vehicle registration certificates.

Road Traffic Management: Lastly, the system offers comprehensive information on road traffic management, outlining the responsibilities of various departments and divisions. It also provides details on traffic patrols and traffic control measures.

By categorizing the inquiries and providing targeted responses, the system effectively supports users in navigating the complexities of traffic laws and ensures a smoother, more informed traffic experience for everyone.

3.2. Overall architecture of the chatbot RLHF

Implementing RLHF goes beyond leveraging available data to expand the ability to learn directly from user feedback. This allows the system to flexibly adapt to diverse requirements and situations without having to rely on pre-programmed rules or scenarios, which often cause rigidity and ineffectiveness when scale and complexity arise.

The integration of RLHF enables continuous improvement through direct optimization and feedback mechanisms. While traditional rule-based chatbots often face limitations in scaling and adapting to changing user needs, systems powered by RLHF can automatically adapt responses based on reward signals from users. This not only helps improve the accuracy of answers, but also makes the system more natural and interactive, going beyond the "one-way" model in which the bot simply answers questions without actually doing anything perform any other response actions.

From a practical deployment perspective, RLHF offers significant benefits when the system must handle large volumes of users with complex and diverse requirements. The ability to learn from real-life feedback helps the system maintain high flexibility and adaptability, avoiding generalization problems that traditional

question-and-answer systems often encounter. Therefore, the adoption of RLHF is not only a step forward in improving the quality of user interaction but also plays a decisive role in ensuring the long-term success of the Q&A system.

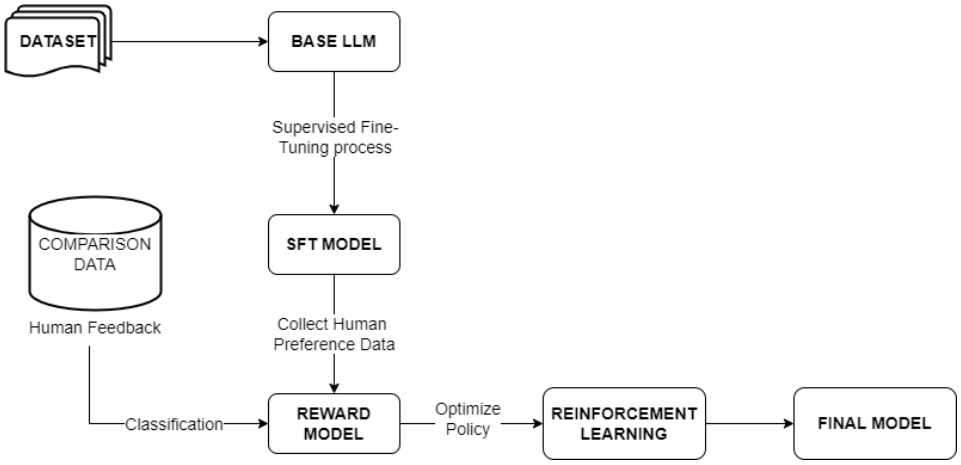

Figure 3.2. Overall architecture chatbot using RL

Data (Dataset):

- The data is initially collected, including real-world examples or human-generated data, and used to train the underlying language model.

Basic Language Model (Base LLM):

- The basic language model is trained from the initial data set. This is the first step towards creating a model capable of understanding and generating natural language.

Supervised Fine-Tuning Process:

- Additional training data from humans is used to fine-tune the underlying language model, creating a supervised fine-tuning model (SFT Model). This process improves the accuracy and effectiveness of the model based on specific feedback from humans.

Comparison Data:

- Response data from humans, where results are compared and evaluated, helps the model understand human preferences.

Reward Model:

- The reward model acts as a classifier, evaluating and ranking outputs based on feedback from humans. It is trained to classify which outputs are better according to human criteria.

Reinforcement Learning:

- The SFT model is then policy-optimized based on the reward model using Reinforcement Learning, to enhance the model's performance in generating results that are more consistent with the user's preferences. human.

Final Model:

- The end result is a fully optimized model, capable of providing more accurate and relevant responses according to human requirements and preferences.

3.3. Construct dataset and model

Traffic law data includes information such as an overview of road traffic laws, specific regulations and provisions, information on safety measures when participating in traffic, and rules of conduct. on the road, and penalties for traffic violations. In addition, the data also includes detailed information about traffic signs, speed regulations, lanes, rights of way, and special regulations related to different types of vehicles. (like motorbikes, cars, bicycles, trucks, etc.).

3.3.1. Experimental data

To build a chatbot specializing in road traffic laws, we collected and created a data set including 4000 questions and corresponding answers. This data is researched and compiled by hand based on content from the "Law Library" website. Each question and answer pair is stored in the following format:

Prompt = Question:\n{}\n ### Answer:\n{}

In the process of preparing data for fine-tuning the RLHF (Reinforcement Learning from Human Feedback) chatbot model using supervised fine-tuning (SFT) technique, our team collected and processed a data set using includes 4,000 samples

from the law library on "Luật giao thông đường bộ (2008)". This process requires great care and precision, as each data sample is selected and processed by hand to ensure representativeness and suitability for model training goals.

Figure 3.3. Dataset 4,000 sample

This data is collected from many reliable sources, including official websites of traffic agencies, legal documents issued in doc, txt, pdf format, and practical guidance documents. enforce laws related to road traffic. However, some pdf documents cannot be processed using the OCR method due to image quality or inappropriate formatting, leading to the removal of some Q&A topics related to administrative procedures or submission processes punish.

In the process of building and evaluating the RLHF (Reinforcement Learning from Human Feedback) model for chatbots, especially in the field of Vietnamese road traffic law, creating different types of answers is an important part. so that the model can learn and distinguish between correct answers (chosen) and inappropriate answers (reject). To do this, the research team used ChatGPT's API to generate non-strict answers according to Vietnamese road traffic laws, with the goal of creating "reject answers" for the assessment reward model.

ChatGPT's API, with the ability to synthesize and generate documents from diverse information sources, has been applied to collect data from law libraries, PDF documents, and online legal forums. Because ChatGPT does not fully comply with

the strict regulations of Vietnam's road traffic laws, the answers generated by this model are used as a form of "reject answer", helping the model learn to distinguish and Remove inaccurate or inappropriate information.

Figure 3.4. Dataset with 1,900 sample

3.3.2. Pretraining Language Models (SeaLLM v3 1.5B)

An SFT (Supervised Fine-Tuning) model, in the context of machine learning and natural language processing, refers to a model that has been further trained on a specific task or dataset after initial pre-training. This technique is often applied to large pre-trained models, such as language models, to specialize them for particular tasks using labeled data. The process helps adapt the model's general language understanding to the specific requirements of the target application.

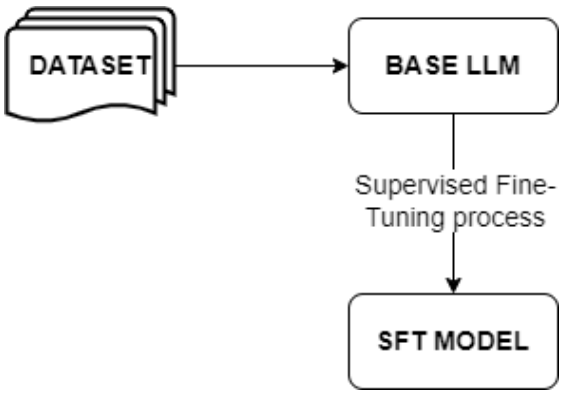

Figure 3.5. Base LLM to SFT model process

Key Concepts in SFT Models:

- Pre-training: The model is first trained on a large and diverse corpus of text data to learn general language patterns. This phase is typically unsupervised, allowing the model to develop a broad understanding of language without labeled data.
- Fine-Tuning: The pre-trained model is then fine-tuned on a smaller, task-specific dataset that includes labeled examples. This phase involves supervised learning, where the model learns to predict labels or outputs based on the input data. Fine-tuning adjusts the model's parameters to improve its performance on the specific task, such as text classification, sentiment analysis, question answering, or translation.
- Supervised Learning: During fine-tuning, the model is trained using labeled data, meaning it learns from pairs of inputs and their corresponding outputs (or labels). The objective is to minimize the discrepancy between the model's predictions and the actual labels in the training set.
- Transfer Learning: SFT models capitalize on transfer learning, where knowledge acquired during pre-training on a broad dataset is transferred and applied to a specific task. This method is efficient, as it enables the model to leverage a vast amount of pre-training data, enhancing its performance even with a smaller fine-tuning dataset.

Applications and Examples:

- Text Classification: Fine-tuning pre-trained models to categorize text into specific categories, such as spam detection or topic classification.
- Sentiment Analysis: Adapting models to detect the sentiment expressed in text, such as positive, negative, or neutral tones.
- Named Entity Recognition (NER): Training models to identify and classify named entities like people, organizations, or locations within text.
- Machine Translation: Fine-tuning models to improve accuracy in translating text between languages.

- Prominent examples of SFT models include GPT-3, BERT, and T5, which are frequently fine-tuned for various applications, showcasing the versatility and efficacy of this approach in enhancing model performance across different NLP tasks.

This stage will basically still train an LM as usual (using available data, available architectures for each task, available optimization methods, available labels). SeaLLM 1.5B was developed by VinBigdata, a technology company belonging to Vingroup in Vietnam. SeaLLM (Southeast Asia Large Language Model) is a large language model designed to process and understand natural language in the context of Southeast Asia, with the ability to support many regional languages, including Vietnamese.

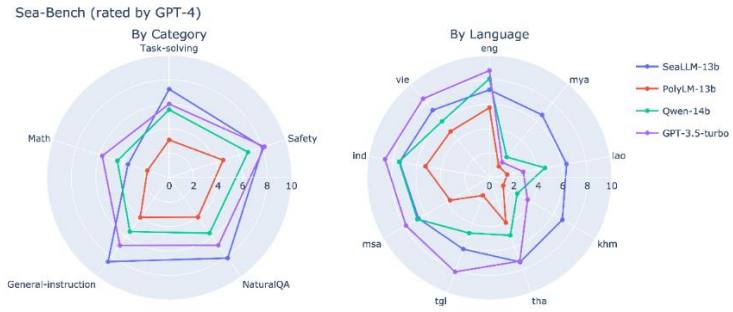

Figure 3.6. Comparison evaluate Sea-bench in 2 mode

The SeaLLM-13b large language model, designed and optimized to support languages in the Southeast Asian region, offers many unique benefits in natural language processing (NLP) for languages. Vietnamese. Different from international models such as the GPT-3.5-turbo, the SeaLLM-13b not only demonstrates superior performance in tasks involving native languages but is also more compact and suitable for demanding applications. High performance without consuming too much computational resources. In particular, with strong support for Vietnamese, SeaLLM-13b helps significantly improve quality and accuracy in chatbot applications, machine translation, and other automated support systems.

Table 3.1. Model evaluation table in 5 languages

Model	**En**	**Zh**	**Id**	**Th**	**Vi**	**Avg_Sea**
Gemma-2B	0.411	0.267	0.296	0.283	0.313	0.297
Sailor-1.8B	0.270	0.239	0.250	0.261	0.260	0.257
Sailor-4B	0.387	0.295	0.275	0.296	0.311	0.294
SeaLLMs-v3-1.5B	0.635	0.745	0.424	0.371	0.465	0.420

The models below are based on sets of exam questions from different countries (M3Exam), designed to assess world knowledge of language models (e.g. language subjects or subjects social studies) and their reasoning abilities.

The SeaLLMs-v3-1.5B-Chat model has the highest score in the Southeast Asia average (42.0), indicating its overall performance and is particularly strong in Southeast Asian languages.

With only 1.5B parameters, SeaLLMs-v3-1.5B-Chat shows a balance between performance and resource efficiency. Compared with larger models such as Sailor-4B-Chat, it can still achieve or surpass the same or even higher scores, which demonstrates the effective refinement and optimization of the model.

These reseach, use 2 prepared data sets including 4,000 samples and 1,900 samples to prepare for pre-trained. The reason here is that mixing 4000 samples with 1900 samples aims to create a more diverse and rich data set, helping to improve model performance when fine-tuning, especially in training the SFT model.

- Avoid Overfitting: When using only correct or relevant answer data, the model can be susceptible to overfitting, which means the model overlearns the samples in the training set and underperforms when faced with new data. By adding reject samples, the model is encouraged to learn to better distinguish between cases, which reduces the risk of overfitting.

- Improves model generality: Adding reject answer helps the model develop better generalization capabilities, as the model must learn to handle a wide range of situations ranging from correct answers to the answer was rejected.

Table 3.2. Distribution of training and testing data

Dataset	**Purpose**	**Size (proportion)**
Train	Model training	85%
Test	Evaluate model	15%

Depending on the classification, regression, and clustering problems, different data division ratios may be required. A ratio of 15% typically provides a reasonable balance between training a model accurately enough to evaluate performance.

Tokenizer is an important component in the process of preprocessing text data before feeding it into the model. It ensures that the data is prepared in the right format so that the model can understand and process it effectively. Tokenizer divides the input text (prompt) into smaller units called tokens. These tokens can be words, characters or subwords depending on the type of tokenizer used. In short, Tokenizer in this stage plays the role of converting text into a number (token) so the model can understand and process.

Besides, fine tuning LLMs requires powerful resources depending on the size of the model, and here we use a method called LoRA (Low - Rank Adaptation). Instead of when fine-tuning or training the model, all parameters will have to be updated, LoRA will add a small amount of parameters to the model, usually very small, in which the original parameters of the model are frozen so it will only update the weights for LoRA parameters.

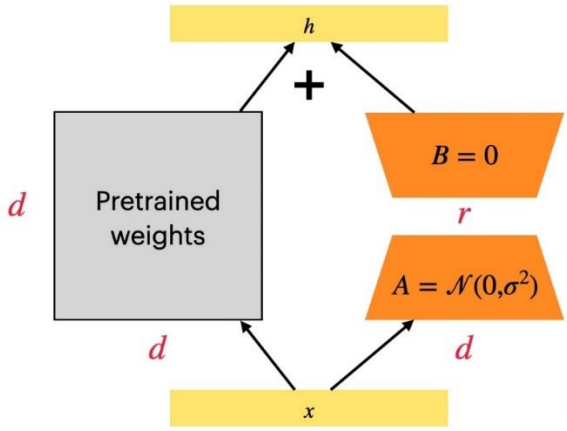

Figure 3.7. Lora Structure (Low-Rank Adaptation)

For fine-tuning, we use the LoRA adapter for fine-tuning to be more effective and cost-effective. However, because the size of each sample is very large, for this test we set max length= 2048, instead because the numbers are smaller to avoid loss of information.

- per_device_train_batch_size = 8
- gradient_accumulation_steps = 4
- num_train_epochs = 5
- metric_for_best_model='loss' (Use the loss function to determine the best model)
- learning_rate: 1e-4
- weight_decay=0.01
- max_length= 2048

All parameters mentioned above, including batch size per device, number of gradient accumulation steps, number of training epochs, best model selection criteria, learning rate, weight reduction factor, and accuracy maximum length of the input sequence, which we will present and explain in more detail in Chapter 4, where we focus on describing the model as well as analyzing the obtained experimental results.

3.3.3. Reward model (Qwen2-0.5B)

A reward model is a key concept in reinforcement learning (RL) and human-in-the-loop learning systems, used to guide the behavior of an agent or system by providing feedback in the form of rewards or penalties. This feedback is based on the quality of the agent's actions or outputs, helping to shape its learning process and performance.

Key Concepts of Reward Models:

- Reinforcement Learning (RL): In RL, an agent learns to make decisions by interacting with an environment and receiving rewards or penalties based on its actions. The primary goal is to develop a policy that maximizes cumulative rewards over time. The reward model is crucial in this process, as it defines the reward signal that informs the agent about the desirability of its actions.
- Reward Signal: The reward model generates a scalar value, known as the reward, which reflects the value or desirability of a particular action in a given state or context. This signal is essential for guiding the learning process, encouraging actions that yield higher rewards and discouraging those that result in lower rewards.
- Human Feedback: In certain systems, particularly those involving language models or conversational agents, human feedback plays a crucial role in constructing or refining the reward model. For example, human evaluators may rate the quality of responses generated by a model, and these ratings are used to adjust the model's behavior.
- Application in Language Models: Reward models are instrumental in aligning large language models with human preferences, ethical standards, or specific performance criteria. This alignment process often involves training a reward model using human-labeled data, followed by the application of reinforcement learning techniques, such as Proximal Policy

Optimization (PPO), to fine-tune the language model based on the reward model's guidance.

- Alignment and Safety: Reward models are vital in aligning AI systems with human values and safety considerations. By defining precise and accurate reward signals, developers can help ensure that AI behaviors are predictable and align with intended outcomes.

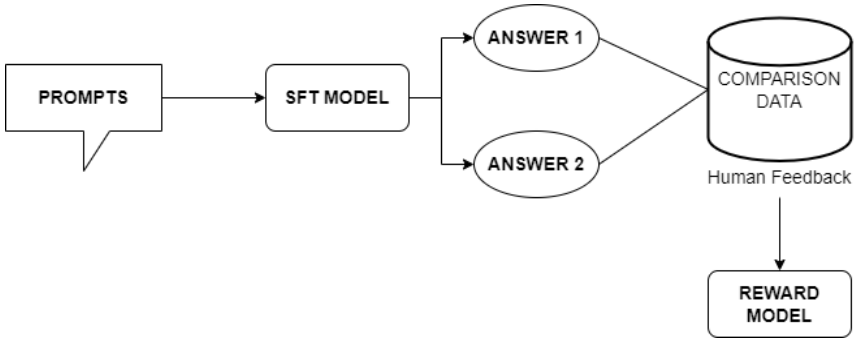

Figure 3.8. Reward Model training process

Component of Reward model:

- Prompts (Questions): These are the inputs or questions given to the model. They are the starting point for the model to generate responses.
- SFT Model (Supervised Fine-Tuned Model): "SFT" stands for Supervised Fine-Tuned, meaning the model has been fine-tuned on a large data set with human-labeled responses. When receiving a question, this model will generate answers to help humans feedback.
- Answer 1 and Answer 2: The SFT model will generate two different answers to the same question. These answers will then be compared. Humans then evaluate and rank these answers in order from best to worst. This process is to label the answers in order of priority based on their quality. For example, if "Answer 1" is considered better than "Answer 2," it will be ranked higher. This ranking order will be used as data to train the Reward Model

- Comparison Data: Human responses are used to compare two answers generated by the SFT model. Humans will rate responses based on quality, accuracy, or other relevant criteria.
- Reward Model: Comparative data, reflecting human preferences, is used to train the reward model. The model learns to predict which answer is better based on the feedback it has received. Over time, the bonus model will help optimize the initial SFT model to generate higher quality answers.

Integrating the chosen and rejected context into the loss function of the Qwen2-0.5B-Instruct model helps improve the performance of chatbots. By identifying and prioritizing relevant contexts and eliminating noise factors, the model can provide more accurate and relevant answers to each specific situation.

Table 3.3. Evaluation results of language models

Datasets	**Gemma-2B**	**Qwen1.5-1.8B**	**Qwen2-0.5B**	**Qwen2-1.5B**
MMLU	42.3	46.8	45.4	56.5
HumanEval	22.0	20.1	22.0	31.1
MATH	11.8	10.1	10.7	21.7
IFEval (Prompt Strict-Acc)	25.9	16.3	16.8	29.0
TruthfulQA	33.1	39.4	39.7	45.9

- MMLU: Measures the model's ability to handle questions on many topics (Massive Multitask Language Understanding).
- HumanEval: Measuring model performance in solving simple programming problems.
- MATH: Measures model performance in solving mathematical problems
- IFEval (Prompt Strict-Acc.): Measures the model's accuracy in answering the correct question based on the request prompt

- TruthfulQA: Assessing the accuracy and fidelity of large language models

These results, Qwen2-1.5B demonstrates outstanding ability to understand and process diverse contexts with a score of 56.5% on MMLU. Additionally, with 31.1% on HumanEval, this model shows the ability to reason and solve problems, something very few models can do well at the same level.

When considering the application of Qwen2-1.5B in the Reward Model, this model can provide high quality responses thanks to better scores compared to other models from the TruthfulQA dataset (45.9%) shows that Qwen2-1.5B can provide accurate assessments of the correctness and appropriateness of the responses generated by the model.

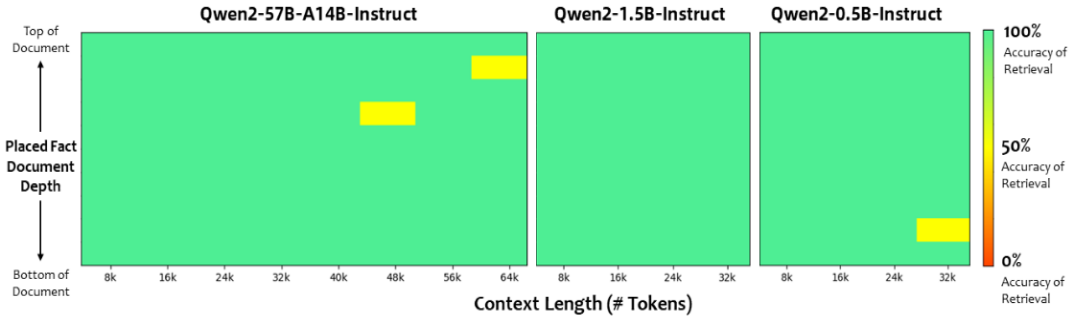

Figure 3.9. Context length and search precision of model Qwen2

These finding, Qwen2-1.5B-Instruct achieves a good balance between performance and required resources. Although not as powerful as the Qwen2-57B-A14B-Instruct in handling very long contexts, it is still capable of handling contexts up to 32k tokens with quite good accuracy. Because in this study, the context of the sample data set related to road traffic laws is not a paragraph too long to extract all the terms or chapters, but will focus on analyzing questions and making recommendations. correct answer. Qwen2-1.5B-Instruct provides a good balance between performance and resource cost because this model consumes less resources in terms of memory and processing power.

Below is the formula for calculating the loss function:

$$\text{loss(re)} = -E(a, y_0, y_1, i, c_{chosen}, c_{reject}) \sim D \left[\log\left(o\left(ro(x, y, c_{chosen}, c_{reject}) - ro(x, y_1 - 1, c_{chosen}, c_{reject})\right)\right)\right]$$

where:

c_{chosen}: The set of chosen contexts.

c_{reject}: The set of rejected contexts.

$ro(x, y, c_{chosen}, c_{reject})$: The reward function now depends not only on the input (x) and output (y) but also on the chosen and rejected contexts.

To optimize the chatbot model, we apply LoRA (Low-Rank Adaptation) technique for the fine-tuning process. LoRA helps reduce computational load and resources without sacrificing model performance. Configuration parameters used in this process include:

- r: Rank of the matrix, set to 16 to balance performance and resources.
- target_modules: Apply LoRA to important modules such as q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj in Transformer.
- lora_alpha: Matrix adjustment factor, set to 16.
- lora_dropout: Set to 0 to optimize the training process.
- bias: Set to "none" to minimize unnecessary training of parameters.
- use_gradient_checkpointing: Set to "unsloth" to reduce memory requirements and increase batch size.

In this model, we will use a dataset consisting of about 1,900 questions and answers related to traffic laws to improve the processing and response capabilities of a chatbot based on the RLHF (Reinforcement) model. Learning with Human Feedback). To optimize the model's ability to learn, we will exploit ChatGPT's API to generate inappropriate or incorrect responses, called reject_answer, in the reward model.

The *preprocess function* plays an important role in preparing data for the fine-tuning process of the machine learning model. This function converts raw text data into input formats that can be processed by the model, including sequences of numeric tokens and attention masks. The input to the function is a set of text pairs, classified into two types: "**chosen**" text and "**rejected**" text.

For each pair of text "chosen" and "rejected", the function uses a tokenizer to convert the text into sequences of numeric tokens. The tokenization process includes:

- Truncation: Text is truncated if it exceeds the maximum length, to match the model's requirements.
- Padding: Text is padded if it is shorter than the maximum length, ensuring that all text is uniform in length.
- Encoding: The text is encoded into input_ids, and an attention_mask is created to specify the parts of the text that the model should pay attention to.

3.3.4. PPO model (Proximal Policy Optimization)

Proximal Policy Optimization (PPO) is a widely used reinforcement learning (RL) algorithm known for its efficiency and stability. Developed by OpenAI, PPO has proven effective in training policies for a variety of complex tasks, including robotics, game playing, and conversational agents, particularly in environments with continuous and discrete action spaces.

Key Concepts of Proximal Policy Optimization (PPO):

- Policy-Based Method: PPO is a policy gradient method, focusing on optimizing the policy—the agent's strategy for selecting actions based on states. The policy is usually represented by a neural network, which learns to make decisions that maximize cumulative rewards.
- Clipped Objective: A distinctive feature of PPO is its clipped objective function. This function limits the extent of policy changes during training, preventing large updates that could lead to instability. It achieves this by

clipping the probability ratio between the new and old policies' actions, ensuring updates remain "proximal" to the existing policy.

- Surrogate Objective Function: PPO uses a surrogate objective function that approximates the actual expected reward, simplifying the optimization process. This function includes a penalty for significant deviations from the current policy, balancing the trade-off between exploration (trying new actions) and exploitation (choosing actions known to yield good results).
- Trust Region: The clipping mechanism in PPO effectively establishes a trust region around the current policy. This concept ensures that updates to the policy are sufficiently small to maintain the reliability of the approximations used in the surrogate objective function.

Applications of PPO:

- Robotics: PPO is used to train robotic agents for tasks like locomotion, manipulation, and navigation, where stable and smooth policy updates are crucial for performance.
- Game Playing: The algorithm is applied in training agents for complex games, including both video games and board games, capable of handling high-dimensional state and action spaces.
- Conversational AI: In fine-tuning language models, PPO can be utilized to refine a model's responses based on a reward model that reflects human preferences or other desired attributes.
- Simulations: PPO is also employed in simulation environments where agents need to learn and adapt to various scenarios, such as economic simulations or environmental modeling.
- PPO's balance of performance and user-friendliness has established it as a leading RL algorithm in both research and practical applications.

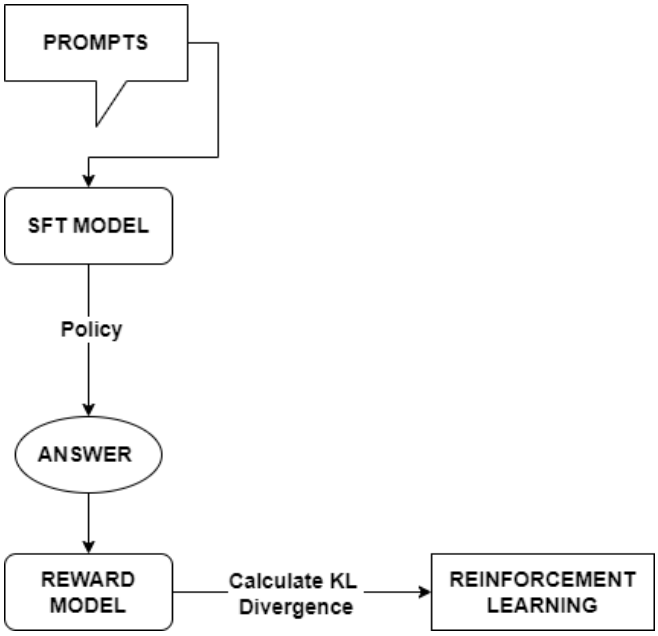

Figure 3.10. Process of training a Reinforcement Learning model

Prompts:

- These are the questions or inputs that the model will process. The user issues a prompt so the model starts generating answers.

SFT Model (Supervised Fine-tuning Model):

- The SFT model is a previously fine-tuned model based on supervised data. This model will receive input from the prompt and apply a policy to generate a response.

Answer:

- Based on the policy of the SFT model, an answer will be generated. This answer is the model's response to the initial prompt.

Reward Model:

- The answer from the SFT model is fed into the Reward model. The Reward model evaluates the quality of this answer and provides a corresponding reward.

Calculate KL Divergence:

- After receiving the reward, the system will calculate KL Divergence between the probability distribution of the SFT model and the optimized

model during the Reinforcement Learning process. KL Divergence measures the difference between these two distributions.

Based on the calculated reward and KL Divergence, the Reinforcement Learning model will be optimized to learn to generate better answers in the future. The goal is to reduce KL Divergence (so that the PPO model is not too different from SFT) while still maximizing rewards.

In addition, Kullback-Leibler Divergence is the crucial component in PPO model that used to measure the difference between the probability distribution of the SFT model and the PPO model after the PPO is updated.

If the KL Divergence is too large, it may indicate that the PPO model has varied too much relative to the SFT model, possibly leading to unstable learning or unexpected responses. In this case, KL Divergence will be used as a penalty in the objective function of PPO, to limit too large variation.

The goal is to keep KL Divergence at an acceptable level, ensuring that the PPO model can learn from the rewards without losing the stability and quality of the original model.

Below is the formula of KL Divergence:

$$R_\theta(x, y) = r(x, y) - \beta \log \left[\frac{\pi^{RL}(y|x)}{\pi^{SFT}(y|x)} \right]$$

Where:

$r(x, y)$ is output of RM

β is a hyper-parameter

$\pi^{RL}(y|x)$ is the policy that will be optimized by PPO

$\pi^{SFT}(y|x)$ is the policy from the SFT model

In this RL training step, we still apply the Lora technique to help optimize the model refinement process to minimize the requirement for computing and memory resources. LoRA works by reducing the number of parameters that need to be updated during the fine-tuning process, by fixing the majority of the parameters of the original model and fine-tuning only a few low-rank weight matrices. This not only helps

reduce computational costs but also increases training efficiency, especially in complex models like PPO, where optimizing policies often requires many update steps.

This fine-tuning task is formulated as a reinforcement learning (RL) problem, where the policy is a language model that takes a prompt and returns a text string. The model's action space includes all tokens in the vocabulary, and the observation space is the distribution of input token sequences, both of which are very large compared to traditional RL applications.

The reward function is where the system combines all the models we discussed into one RLHF process. Given the prompt, x from the dataset, two texts, y_1, y_2, are generated – one from the original language model and one from the current iteration of the refined policy. The text from the current policy is passed to the priority model, which returns a scalar concept of "priority", r_θ

Finally, the updated rule is the parameter updated from the PPO that maximizes the reward value in the current batch of data (PPO is on-policy, this means the parameters are only updated with current batch).

3.4. Interface Design

To be able to easily interact with the virtual assistant system, these reseach use the gradient library to easily create an interface for testing chatbot and virtual assistant models without spending too much time on design and interface stability.

Gradio provides easily customizable components, helping to design chatbot interfaces according to specific requirements. Thanks to this library, it is possible to create dialog boxes, input areas, and even other complex interactive components to enhance the user experience.

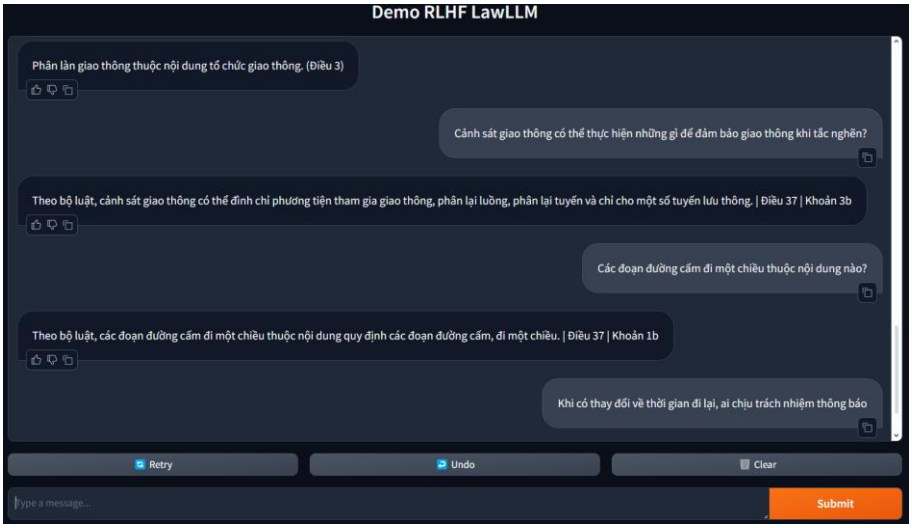

Figure 3.11. Interface design of chatbot using Gradient

CHAPTER 4. EXPERIMENTAL MODEL

4.1. Evaluate SFT model

According to the research analyzed previously in subsection ***3.3.2***, Our research sets up an environment to fine-tune a language model using the transformers library. Here, in this study, a fine-tune technique called LoRA (Low-Rank Adaptation) is applied to reduce the number of parameters that need to be trained.

The configuration supports many optimizations such as using "unsloth" to test gradients, which helps manage memory usage during training. An important parameter in training and using natural language models is max_seq_length, which we set to the value '2048' instead of '1024' to determine the maximum length of the sequence. that the model can handle. Below is the reason to explain the above problem

- The value '2048' allows the model to handle longer text strings, up to '2048' tokens (including words, punctuation, spaces,...). This is especially useful in tasks that require the model to understand long contexts, such as parsing long documents, answering questions from long texts, or handling lengthy conversations in chatbots.
- With shorter string lengths, like '1024', the model may have to truncate or ignore the end of the text if it exceeds this maximum length. This can lead to the loss of important information and reduce the quality of predictions.

To handle hardware resource capacity to avoid GPU ram overflow, if the hardware is not powerful enough, you may encounter problems with slow processing time or running out of memory (Out Of Memory - OOM). Below we test the model on RTX 4090 24GB:

Table 4.1. Experimental configuration for the SFT model

Num GPUs	1
Num examples	5,039
Num Epochs	5
Batch size per device	8
Gradient Accumulation steps	4
Total batch size	32
Total steps	785
Number of trainable parameters	18,464,768
Time out	35 minutes 50 seconds

In the above model, we used only one GPU for training with Number of samples (for example) in data trainer as '5039' samples. The entire training process was completed in 785 steps and took 35 minutes and 50 seconds. Training ends at Epoch 5/5, indicating that the process completes the full 5 epochs.

Table 4.2. Compare loss on the training and validation sets of the SFT model

Epoch	**Training Loss**	**Validation Loss**
0	1.022700	1.078682
2	0.833800	0.878178
4	0.738300	0.846794

In the first epoch, Training Loss and Validation Loss are relatively close to each other, showing that the model is not overfit. At epoch 2, both Training Loss and Validation Loss decreased, and they remained close to each other. This shows that the model is learning well without obvious signs of overfitting. By epoch 4, Training Loss continued to decrease, but Validation Loss decreased more slowly. However, the difference between Training Loss and Validation Loss is still not too large, showing that the model is not seriously overfit, but may be approaching that threshold.

- Good progress: The model is making good progress, both Training Loss and Validation Loss are decreasing steadily in the last ‘epoch’.
- There are no signs of overfitting: The gap between Training Loss and Validation Loss is not too large, showing that the model is not seriously overfitting.

However, to achieve higher performance, it is necessary to combine many factors such as data quality, model architecture, hyperparameters, deep learning techniques and performance evaluation.

Evaluate:

Table 4.3. Testing results of the SFT model

eval_loss	0.8467943668365479
eval_runtime	23.549
eval_samples_per_second	37.794
eval_steps_per_second	4.756
epoch	4.984126984126984

With important "Loss" parameters on the evaluation data set close to the final “Validation Loss” from the training process, it shows that the model operates stably and is not overfit. This value is close to the final “Validation Loss” from training '0.846794'. This shows that the model performs consistently between the training set and the evaluation set, without a large difference, meaning the model is not overfitting and has good generalization ability on unseen data.

In addition, the total time to perform the evaluation process '23,549'. This short time shows that the evaluation process was carried out effectively. The evaluation has almost completed the 5th Epoch, indicating that the model has been evaluated almost all the data in the last epoch of training. This also demonstrates that the assessment covers the entire range of data in the last epoch.

4.2. Evaluate Reward model

Based on the results from the training and evaluation of the reward model in the RLHF chatbot system on road traffic laws, we can draw some important observations from the parameters provided.

Even in the first epoch, the model showed effectiveness with Training Loss and Validation Loss values close to each other, and achieved an impressive accuracy of 99.88%. This proves that the model is effectively learning features from the data without overfitting problems. By the 2nd Epoch, both Training Loss and Validation Loss continued to decrease, and the accuracy reached a maximum of 100%, showing that the model had learned well the necessary features for accurate classification.

Achieving 100% accuracy in Epoch 2 is a clear sign that the model has captured all the important factors in the training data, helping to make accurate predictions without errors. More importantly, even though the model achieves absolute accuracy, the difference between Training Loss and Validation Loss is still very small.

Overall, this result reflects a very successful training process, where the model not only learned well but also maintained stability and generalization ability.

Table 4.4. Experimental configuration for the Reward model

Global Step	471
Train Runtime	11 minutes 17 seconds
Train Samples per Second	22.319 seconds
Train Steps per Second	0.695
Epoch	2.9904761904761905

Model training completed in 677.3 seconds (~11 minutes 17 seconds), demonstrating good training performance in a reasonable amount of time. With a processing speed of 22,319 samples per second and nearly 0.695 steps per second, the model demonstrates the ability to process data quickly and efficiently. The training results show that the model achieved good performance with very low Training Loss, and the training process took place in a reasonable amount of time.

The data processing speed and training steps also show that the model works effectively. With the number of Epochs almost completed at 3, the model is on the verge of completion and achieving impressive results.

4.3. Evaluate PPO model

BLEU (Bilingual Evaluation Understudy) is a popular index used to evaluate the quality of machine translation or automatic text generation models, by comparing the model output with one or more reference sentences) written by humans.

After fine-tuning the RLHF model, the BLEU score can be used to compare the generated responses to standard reference responses. Higher BLEU scores indicate that the generated answers are closer to the reference sentences, indicating better output quality.

In these research, we use "Winner rate" in this context to be understood as the percentage of times the PPO model produces an answer that is rated better than the SFT model. It reflects the extent to which the PPO model is superior in generating responses compared to the SFT model, based on a reward model for comparison. In this study, we use a list of "Like []" to compare scores:

- If the score of the PPO model is higher (score response ppo > score response sft), then we consider that the PPO model has won in this case. Therefore, '1' is added to the "Like []" list.
- On the contrary, if the score of the SFT model is higher or equal (score response sft >= score response ppo), then we consider that the SFT model has won or the two models have equivalent quality. In this case, '0' is added to the "Like []" list.

Table 4.5. Comparison table between BLEU score and Winner Rate for two models SFT and PPO

Metrics	**SFT model**	**PPO model**
BLEU Score	0.56	0.8
Winner Rate	28%	72%

BLEU score range:

- 0 - 0.3: The quality of the generated sentence is poor, not similar to the reference sentence.
- 0.3 - 0.5: The generated sentence has some similarities with the reference sentence, but there are still many differences.
- 0.5 - 0.7: The quality is quite good, the generated sentence has the majority of n-grams similar to the reference sentence.
- 0.7 - 1.0: High quality, the generated sentence is almost identical to the reference sentence.

With a BLEU score of 0.56, the SFT model shows that the generated answers have quite good quality. They have a significant degree of similarity with the referenced answers, but there may still be some errors or minor differences (coming from context, content or spelling).

The PPO model achieved a BLEU score of 0.8, indicating outstanding quality. The generated responses almost perfectly matched the reference sentences, demonstrating that the PPO model had learned to generate more accurate and appropriate responses.

Currently, there is no specific and comprehensive way to evaluate the model to evaluate whether it is suitable or not, especially it is necessary to evaluate whether it is okay when experimenting in the context of road traffic laws. So instead of using automatic evaluation metrics, we will evaluate manually using 20 questions and random context extracted from dataset data. The evaluation results below are manually compared directly with the answer and context, with the criteria that the

answer can be generalized, or almost enough for the entire context, and must include the correct format. of sample, however, if the answer contains an incorrect element, the answer will be considered wrong, for example asking about the Road Signaling System but the answer is related to the responsibilities of the traffic police.

Table 4.6. Manual assessment table of RLHF

Question	**Context**	**Answer**	**Check**
Cảnh sát giao thông thực hiện nhiệm vụ gì khi có tình huống gây ách tắc?	Theo bộ luật, cảnh sát giao thông thực hiện nhiệm vụ phân lại luồng, phân lại tuyến khi có tình huống gây ách tắc. | Điều 37 | Khoản 3b	Chào bạn, cảnh sát giao thông thực hiện nhiệm vụ bảo đảm giao thông và phân luồng giao thông khi có tình huống gây ách tắc. | Điều 37 | Khoản 3b	PASS
Cảnh sát giao thông có nhiệm vụ chỉ huy giao thông khi nào?	Tại Điều 9 Thông tư 65/2020/TT-BCA, Điều 9 Thông tư 65/2020/TT-BCA quy định cảnh sát giao thông thực hiện nhiệm vụ và phối hợp với các lực lượng trong đảm bảo trật tự an toàn giao thông đường bộ như sau: Chỉ huy, điều tiết giao thông khi có tình huống đặc biệt phức tạp, nguy hiểm xảy ra hoặc được chỉ đạo để kiểm	Cảnh sát giao thông có nhiệm vụ chỉ huy giao thông tại nơi giao nhau nhiều đường, cầu đường bộ và các đường không do doanh nghiệp kinh doanh. (Điều 9)	PASS

	soát, xử lý các sự cố gây ách tắc giao thông, tình huống trật tự, an toàn giao thông trên các tuyến đường chính, các đường cao tốc, đường cao tốc trên vùng biển, đường bộ nước ngoài, các tuyến đường chuyên dùng giao thông (áp dụng những tuyến đường bộ có chức năng liên quan trực tiếp đến quốc phòng, an ninh quốc gia và kết nối với quốc lộ). Bảo đảm trật tự, an toàn giao thông trong các hoạt động chuyên môn của lực lượng quản lý đường bộ theo quy định của pháp luật.		
Việc cấm dừng đỗ xe thuộc nội dung nào?	Theo bộ luật, việc cấm dừng đỗ xe thuộc nội dung quy định các đoạn đường cấm đi, đường một chiều, nơi cấm dừng, cấm đỗ, cấm quay đầu xe. | Điều 37 | Khoản 1b	Theo bộ luật, việc cấm dừng đỗ xe thuộc nội dung quy định các đoạn đường cấm đi, đường một chiều, nơi cấm đi, cấm đỗ, cấm quay đầu xe. | Điều 37 | Khoản 1b	PASS

Những nội dung nào thuộc tổ chức giao thông?	Theo bộ luật, tổ chức giao thông bao gồm phân làn, phân luồng, phân tuyến và quy định thời gian đi lại. | Điều 37 | Khoản 1a	Theo bộ luật, tổ chức giao thông bao gồm phân làn, phân luồng và phân tuyến. | Điều 37 | Khoản 1a	PASS
Khi có sự cố giao thông, ai chịu trách nhiệm ứng cứu?	Theo bộ luật, cơ quan tổ chức giao thông chịu trách nhiệm ứng cứu khi có sự cố giao thông. | Điều 37 | Khoản 1c	Theo bộ luật, cơ quan tổ chức giao thông chịu trách nhiệm ứng cứu khi có sự cố giao thông. | Điều 37 | Khoản 1c	PASS
Bộ Giao thông vận tải có nhiệm vụ gì trong tổ chức giao thông?	Theo bộ luật, Bộ Giao thông vận tải chịu trách nhiệm tổ chức giao thông trên hệ thống quốc lộ. | Điều 37 | Khoản 2a	Theo bộ luật, Bộ trưởng Bộ Giao thông vận tải phụ trách tổ chức giao thông trên hệ thống quốc lộ. | Điều 37 | Khoản 2b	FAIL
Những thay đổi về phân luồng giao thông cần được ai thông báo?	Theo bộ luật, những thay đổi về phân luồng giao thông cần được cơ quan tổ chức giao thông thông báo. | Điều 37 | Khoản 1c	Theo bộ luật, những thay đổi về phân luồng giao thông cần được cơ quan tổ chức giao thông thông báo ngay. | Điều 37 | Khoản 1c	PASS

Phân làn giao thông có thuộc nội dung của tổ chức giao thông không?	Theo bộ luật, phân làn giao thông thuộc nội dung của tổ chức giao thông. | Điều 37 | Khoản 1a	Theo bộ luật, phân làn giao thông là một phần của tổ chức giao thông. | Điều 37 | Khoản 1a	PASS
Quy định về các đoạn đường cấm dừng, đỗ xe nằm trong nội dung nào?	Theo bộ luật, quy định này thuộc nội dung quy định các đoạn đường cấm, nơi cấm dừng, đỗ xe. | Điều 37 | Khoản 1b	Theo bộ luật, các đoạn đường cấm dừng, đỗ xe nằm trong nội dung quy định về các đoạn đường cấm, đi một chiều, nơi cấm dừng, đỗ xe. | Điều 37 | Khoản 1b	PASS
Trách nhiệm thông báo khi có thay đổi về phân tuyến giao thông thuộc về ai?	Theo bộ luật, cơ quan tổ chức giao thông chịu trách nhiệm thông báo khi có thay đổi về phân tuyến giao thông. | Điều 37 | Khoản 1c	Theo bộ luật, cơ quan tổ chức giao thông chịu trách nhiệm thông báo khi có thay đổi về phân tuyến giao thông. | Điều 37 | Khoản 1c	PASS
Cảnh sát giao thông	Theo bộ luật, cảnh sát giao thông có thể tạm thời đình	Theo bộ luật, cảnh sát giao thông có thể đình chỉ đi lại	PASS

có thể thực hiện những gì để đảm bảo giao thông khi tắc nghẽn?	chỉ đi lại và phân lại luồng, tuyến giao thông. | Điều 37 | Khoản 3b	và phân lại luồng. | Điều 37 | Khoản 3b	
Các đoạn đường cấm đi một chiều thuộc nội dung nào?	Theo bộ luật, các đoạn đường cấm đi một chiều thuộc nội dung quy định các đoạn đường cấm, nơi cấm dừng, đỗ xe. | Điều 37 | Khoản 1b	Theo bộ luật, các đoạn đường cấm đi một chiều thuộc nội dung quy định các đoạn đường cấm, nơi cấm dừng, đỗ xe. | Điều 37 | Khoản 1b	PASS
Phân tuyến giao thông là nội dung của tổ chức giao thông như thế nào?	Theo bộ luật, phân tuyến giao thông là một phần của tổ chức giao thông bao gồm phân làn, phân luồng, phân tuyến và quy định thời gian đi lại. | Điều 37 | Khoản 1a	Theo bộ luật, phân tuyến giao thông là nội dung tổ chức giao thông trên đường một chiều và tuyến đường cụ thể. | Điều 37 | Khoản 1a	PASS
Khi có thay đổi về thời gian đi lại, ai chịu trách	Theo bộ luật, cơ quan tổ chức giao thông chịu trách nhiệm thông báo khi có thay đổi về thời gian đi lại. | Điều 37 | Khoản 1c	Chào bạn, người quản lý đường bộ chịu trách nhiệm thông báo khi có thay đổi về thời gian đi lại. | Điều 35 | Khoản 1c	FAIL

nhiệm thông báo?			
Cảnh sát giao thông có thể đình chỉ đi lại trong trường hợp nào?	Theo bộ luật, cảnh sát giao thông có thể đình chỉ đi lại khi có tình huống ách tắc hoặc yêu cầu cần thiết về an ninh, trật tự. | Điều 37 | Khoản 3b	Theo bộ luật, cảnh sát giao thông có thể đình chỉ đi lại khi có tình huống ách tắc, tình trạng bão, mưa lũ, lũ làn hoặc khác có yêu cầu cần đình chỉ đi lại. | Điều 37 | Khoản 4b	FAIL
Khi gặp ách tắc giao thông, cảnh sát giao thông có thể làm gì?	Theo bộ luật, cảnh sát giao thông có thể tạm thời đình chỉ đi lại và phân lại luồng giao thông. | Điều 37 | Khoản 3b	Khi có ách tắc giao thông, cảnh sát giao thông có thể dừng lại và chỉ cho phương tiện tham gia giao thông theo tuyến đường này. | Điều 37 | Khoản 3b	PASS
Cảnh sát giao thông có vai trò gì trong việc điều khiển giao thông?	Theo bộ luật, cảnh sát giao thông chỉ huy và điều khiển giao thông trên đường. | Điều 37 | Khoản 3a	Theo bộ luật, cảnh sát giao thông có vai trò điều khiển giao thông theo quy tắc giao thông và tổ chức giao thông cho người và phương tiện tham gia giao thông đường bộ. | Điều 37 | Khoản 3a	PASS
Khi có sự thay đổi về	Theo bộ luật, cơ quan tổ chức giao thông sẽ thông	Theo bộ luật, cơ quan tổ chức giao thông sẽ thực hiện	PASS

thời gian đi lại tạm thời, ai sẽ thực hiện thông báo?	báo khi có thay đổi về thời gian đi lại tạm thời. | Điều 37 | Khoản 1c	thông báo khi có sự thay đổi về thời gian đi lại tạm thời. | Điều 37 | Khoản 1c	
Cảnh sát giao thông có thể làm gì để bảo đảm giao thông trong tình huống ách tắc?	Theo bộ luật, cảnh sát giao thông có thể đình chỉ tạm thời đi lại ở một số đoạn đường, phân lại luồng, tuyến và nơi tạm dừng, đỗ xe. | Điều 37 | Khoản 3b	Theo bộ luật, cảnh sát giao thông có thể thực hiện các biện pháp bảo đảm giao thông hoặc đình chỉ hoạt động tại một số đoạn đường hoặc tạm thời thu hồi phần đường để tổ chức giao thông an toàn. | Điều 37 | Khoản 3b	PASS
Các quy định về đường cấm quay đầu xe nằm trong nội dung nào của tổ chức giao thông?	Theo bộ luật, các quy định này thuộc nội dung quy định các đoạn đường cấm, đi một chiều, nơi cấm dừng, đỗ, quay đầu xe. | Điều 37 | Khoản 1b	Theo bộ luật, quy định các đường cấm quay đầu xe là thuộc nội dung quy định về dừng, đỗ xe. | Điều 37 | Khoản 1b	PASS
Cảnh sát giao thông có quyền	Theo bộ luật, cảnh sát giao thông có quyền tạm thời đình chỉ đi lại khi có tình	Theo bộ luật, cảnh sát giao thông có quyền tạm thời đình chỉ đi lại trong các	PASS

tạm thời đình chỉ đi lại trong trường hợp nào?	huống ách tắc hoặc yêu cầu cần thiết về bảo đảm an ninh, trật tự. | Điều 37 | Khoản 3b	trường hợp bảo đảm an ninh, trật tự hoặc bảo vệ sinh hazard của phương tiện giao thông đường bộ. | Điều 37 | Khoản 3b	
Phân làn giao thông thuộc nội dung nào của tổ chức giao thông?	Theo bộ luật, phân làn giao thông thuộc nội dung phân làn, phân luồng, phân tuyến và quy định thời gian đi lại cho người và phương tiện. | Điều 37 | Khoản 1a	Theo bộ luật, phân làn giao thông thuộc nội dung phân làn, phân luồng, phân tuyến. | Điều 37 | Khoản 1a	PASS
Hệ thống báo hiệu đường bộ gồm những gì?	Hệ thống báo hiệu đường bộ gồm hiệu lệnh của người điều khiển giao thông; tín hiệu đèn giao thông, biển báo hiệu, vạch kẻ đường, cọc tiêu hoặc tường bảo vệ, rào chắn. (Điều 10)	Chào bạn, gồm súng đèn, đèn tín hiệu, biển báo, vạch kẻ đường và người điều khiển. (Điều 10)	PASS

CHAPTER 5. CONCLUSION

5.1. Result

The automatic response system for traffic laws was successfully built using the RLHF model, with main functions including: questions and answers about regulations, questions and answers about issues related to concepts. This system is capable of answering users questions relatively accurately, understanding users' intentions and interacting naturally with them.

5.2. Development

To improve the quality of the system, the following development directions can be implemented:

- Expand the training dataset: The current training dataset is limited, it is necessary to expand this dataset so that the system can answer questions more accurately.
- Enhance the ability to handle open questions: The current system can handle closed questions well, it is necessary to enhance the ability to handle open questions so that the system can answer more diverse questions.
- Enhance the ability to interact: The current system can interact with users naturally, however, this ability to interact can be enhanced so that the system can create a better experience for users.

5.3. Limitations

The system still has some limitations as follows:

- Understanding ability: The system's understanding ability is not yet perfect, it needs to be improved so that the system can understand the user's meaning more accurately.
- Interactivity: The system's interaction ability is not yet completely natural, it needs to be improved so that the system can create a better experience for the user.

- Accuracy: The system's accuracy is not yet 100%, it needs to be further improved so that the system can answer questions most accurately.

5.4. Summary

The project has achieved the best results, however, there are still some modes that need to be solved. With the published oriented developments, the system will be increasingly perfect and better meet the needs of users.

REFERENCE

Luật Giao thông đường bộ 2008 (Luật số 23/2008/QH12) ngày 13/11/2008.

Loc, D. (2023). RLHF và cách ChatGPT hoạt động.

website: https://viblo.asia/p/rlhf-va-cach-chatgpt-hoat-dong-3RlL5AEmLbB.

Phuc, P (2023). ChatGPT series 4: RLHF & DPO: Kỹ thuật mới đơn giản hơn, tăng cường khả năng Fine-tuning cho Large language models.

Chip, H. (2023). RLHF: Reinforcement Learning from Human Feedback.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T., Radford, A., Amodei, D., Christiano, P. F., & Irving, G. (2019). Fine-Tuning Language Models from Human Preferences. arXiv preprint, arXiv:1909.08593.

Amirloo, E., Samadi, S., Saeedi, P., & Furlanello, T. (2024). Understanding Alignment in Multimodal LLMs: A Comprehensive Study. arXiv preprint, arXiv:2407.02477.

Zhong, Y., & Zhou, Y. (2024). Low-Rank Interconnected Adaptation Across Layers. arXiv preprint, arXiv:2407.09946.

Wang, Y., Lu, L., Wang, M., & Xiong, X. (2024). Reinforcement Learning from Human Feedback for Lane Changing of Autonomous Vehicles in Mixed Traffic. arXiv preprint, arXiv:2408.04447.

Nguyen, X.-P., Dang, Q.-T., Tran, T.-A., & Le, V.-T. (2023). SeaLLMs: Large Language Models for Southeast Asia. arXiv preprint, arXiv:2312.00738.

Dai, J., Pan, X., Sun, R., Ji, J., Xu, X., Liu, M., ... & Yang, Y. (2023). Safe RLHF: Safe reinforcement learning from human feedback. arXiv preprint, arXiv:2310.12773.

Liu, G. K. M. (2023). Transforming Human Interactions With AI Via Reinforcement Learning With Human Feedback (RLHF). Massachusetts Institute of Technology.

Sanghi, N. (2024). Proximal Policy Optimization (PPO) and RLHF. In Deep Reinforcement Learning with Python: RLHF for Chatbots and Large Language Models (pp. 461-522). Berkeley, CA: Apress.

Dong, H., Xiong, W., Pang, B., Wang, H., Zhao, H., Zhou, Y., & Zhang, T. (2024). Rlhf workflow: From reward modeling to online rlhf. arXiv preprint, arXiv:2405.07863.

Zhang, T., Wu, F., Katiyar, A., Weinberger, K. Q., & Artzi, Y. (2020). Revisiting few-sample BERT fine-tuning. arXiv preprint, arXiv:2006.05987.

Sutton, R. S., Machado, M. C., Holland, G. Z., Szepesvari, D., Timbers, F., Tanner, B., & White, A. (2023). Reward-respecting subtasks for model-based reinforcement learning. Artificial Intelligence, 324, 104001.

Kwa, T., Thomas, D., & Garriga-Alonso, A. (2024). Catastrophic Goodhart: regularizing RLHF with KL divergence does not mitigate heavy-tailed reward misspecification. arXiv preprint, arXiv:2407.14503.

Xu, N., Zhao, J., Zu, C., Gui, T., Zhang, Q., & Huang, X. (2024). Advancing Translation Preference Modeling with RLHF: A Step Towards Cost-Effective Solution. arXiv preprint, arXiv:2402.11525.

Wang, B., Zheng, R., Chen, L., Liu, Y., Dou, S., Huang, C., ... & Jiang, Y. G. (2024). Secrets of rlhf in large language models part ii: Reward modeling. arXiv preprint, arXiv:2401.06080.

Siththaranjan, A., Laidlaw, C., & Hadfield-Menell, D. (2023). Understanding hidden context in preference learning: Consequences for rlhf. In Socially Responsible Language Modelling Research.

Hou, Z., Niu, Y., Du, Z., Zhang, X., Liu, X., Zeng, A., & Dong, Y. (2024). ChatGLM-RLHF: Practices of Aligning Large Language Models with Human Feedback. arXiv preprint, arXiv:2404.00934.

Hou, Z., Niu, Y., Du, Z., Zhang, X., Liu, X., Zeng, A., ... & Dong, Y. (2024). ChatGLM-RLHF: Practices of Aligning Large Language Models with Human Feedback. arXiv preprint, arXiv:2404.00934.

Latif, E., & Zhai, X. (2024). Fine-tuning ChatGPT for automatic scoring. Computers and Education: Artificial Intelligence, 6, 100210.