Empowering LLMs in Task-Oriented Dialogues: A Domain-Independent Multi-Agent Framework and Fine-Tuning Strategy

Anonymous ACL submission

Abstract

Task-oriented dialogue systems based on Large Language Models (LLMs) have gained increasing attention across various industries and achieved significant results. Current approaches condense complex procedural workflows into a single agent to achieve satisfactory performance on large-scale LLMs. However, these approaches face challenges to achieve comparable performance on fine-tuned lightweight LLMs, due to their limited capabilities in handling multiple complex logic. In this work, we design a Domain-Independent Multi-Agent Framework (DIMF), which contains Intent Classification Agent, Slot Filling Agent and Response Agent. This approach simplifies the learning complexity and enhances the generalization ability by separating the tasks into domain-independent components. In this framework, we enhance the capabilities in contextual understanding using the Direct Preference Optimisation (DPO) method, and propose a simple and effective Data Distribution Adaptation (DDA) method to mitigate degradation issues during DPO training. Experiments conducted on the MultiWOZ datasets show that our proposed method achieves a better average performance among all the baselines. Extensive analysis also demonstrates that our proposed framework exhibits excellent generalizability and zero-shot capability.

1 Introduction

Task-oriented dialogue (TOD) systems play a significant role in both academic research and industry.(Peng et al., 2022; Xu et al., 2024). Researchers have divided the traditional TOD systems into the following several key components (Zhang et al., 2020): 1) Natural Language Understanding (NLU) (Karanikolas et al., 2023; Cambria, 2024). 2) Dialogue State Tracking (DST) (Feng et al., 2023; Heck et al., 2023). 3) Dialogue Policy. 4) Natural Language Generation (NLG) (Li et al., 2020). With the development of the Large Language Model (LLM), recent research has mainly focused on leveraging the strong capabilities and generalization of LLMs to solve the complex task of TOD (Qin et al., 2023a; Algherairy and Ahmed, 2024; Chung et al., 2023). The LLM-based multi-agent approach has been proven to be effective in multi-domain TOD systems (Gupta et al., 2024).

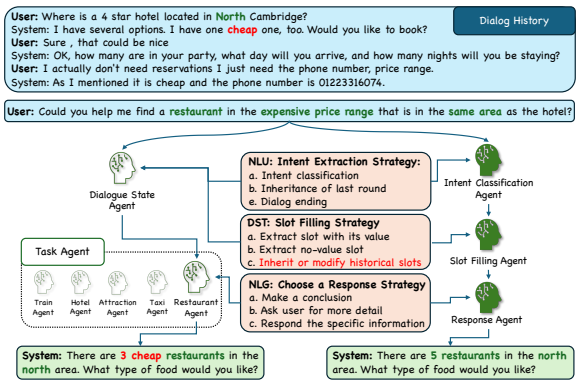

Figure 1: Different architectures of our proposed system and other LLM-based systems. The left part is other LLM-based systems and the right is ours. The information in the orange box indicates the strategies in different sub-tasks that the agent needs to follow.

Existing methodologies often attempt to condense complex procedural workflows of TOD systems into a single LLM-based agent. However, such implementations typically rely on exceptionally large-scale models, such as GPT-4 (Achiam et al., 2023) and Claude, to achieve satisfactory performance (Xu et al., 2024; Gupta et al., 2024). In contrast, smaller open-source models, even when fine-tuned for specific tasks, struggle to attain comparable completion quality (Xu et al., 2024; Gupta et al., 2024). This discrepancy contrasts sharply with their competitive performance in other NLP tasks (e.g., LLama (Touvron et al., 2023) or Qwen (Yang et al., 2024) models post-fine-tuning), suggesting that the inherent complexity of TOD ne-

cessitates specialized approaches. We posit that effective modeling of multi-step procedural logic and developing targeted learning strategies are critical to bridging this performance gap.

To address this challenge, we propose a Domain-Independent Multi-Agent Framework (DIMF), which contains Intent Classification Agent, Slot Filling Agent and Response Agent. Unlike the current methods, which obtain the dialogue state by a single agent, DIMF decouples the workflow into several components. As illustrated in Figure 1, both phases require contextual reasoning and policy-guided decision-making capabilities, easily conflated in monolithic agent architectures. The task separation design stems from our observation of domain relevance and challenges in slot integration from history dialogue during slot filling process. This approach guarantees that the agent considers the slot that matches the current specific domain. Furthermore, this modular decomposition facilitates the enhancement of targeted capability through reinforcement learning techniques (e.g., DPO/PPO (Rafailov et al., 2023; Schulman et al., 2017)), enabling specialized optimization while maintaining domain adaptability. We therefore propose a Data Distribution Adaptation (DDA) method designed to mitigate the degradation of DPO training attributable to the diversity of domain types.

The experimental results indicate that the framework and training methodology significantly enhance the performance of the fine-tuned models. Additionally, it was observed that the domain-independent design exhibits a robust zero-shot capability. In conclusion, this paper offers the following contributions:

- We design a novel Domain-Independent Multi-Agent Framework for TOD systems based on LLMs. Our approach separates the complex task into three sub-tasks which better leverages the generalization capabilities of LLMs.
- We utilize DPO during the training process, and innovatively propose a Data Distribution Adaptation method to alleviate the DPO's training degradation problem during the DPO training process.
- Our new framework and training strategy for the TOD system have enhanced the system's scalability and zero-shot capabilities, allowing the system to maintain good performance even on domains it has not seen before.

2 Background

2.1 Large Language Models as Agents

Recently, many efforts have been made to build systems through LLMs acting as agents for planning, decision-making, and acting tasks between various specialized APIs, dialogue, or other simpler tools to perform complex tasks (Liu et al., 2023; Liang et al., 2023; Deng et al., 2024). ReAct (Yao et al., 2023) method is a prompt framework that has been widely used for fine-tuning the LLMs with the ability of reasoning and action based on text. Various tasks such as logical reasoning (Du et al., 2023; Tang et al., 2023), societal simulations (Zhou et al., 2023), tool learning (Qin et al., 2023b; Shen et al., 2024) have achieved significant improvement in performance using LLMs as agents.

However, most research focuses on task-specific scenarios with poor scalability. The challenge of LLMs working as agents that can generalize better and adapt to different tasks needs more research.

2.2 Direct Preference Optimisation (DPO)

Direct Preference Optimisation (DPO) (Rafailov et al., 2024) is a popular method for learning from human-preference data, and it has been widely leveraged to improve the performance of pre-trained LLMs on downstream tasks (Wang et al., 2023; Tunstall et al., 2023). DPO directly uses pairwise preference data for model optimization. In this way, we can directly train the language model through the reward learning pipeline, eliminating the need for the reinforcement learning stage.

Although the DPO method facilitates model training, experiments demonstrate that the DPO loss has flaws: Compared to learning to generate responses preferred by humans, the DPO loss function demonstrates a tendency for LLMs to readily learn to avoid generating responses that humans disprefer (Feng et al., 2024). Based on this conclusion, DPO exhibits significant degradation issues on data where the Levenshtein Distance between positive and negative examples is small. The reason is that with highly similar positive and negative examples, the DPO process tends to reject the negative examples, which in turn reduces the generation probability for the corresponding positive examples (Pal et al., 2024). Thus, the DPO process can lead to a simultaneous decrease in the reward functions for both positive and negative examples, which leads to degradation.

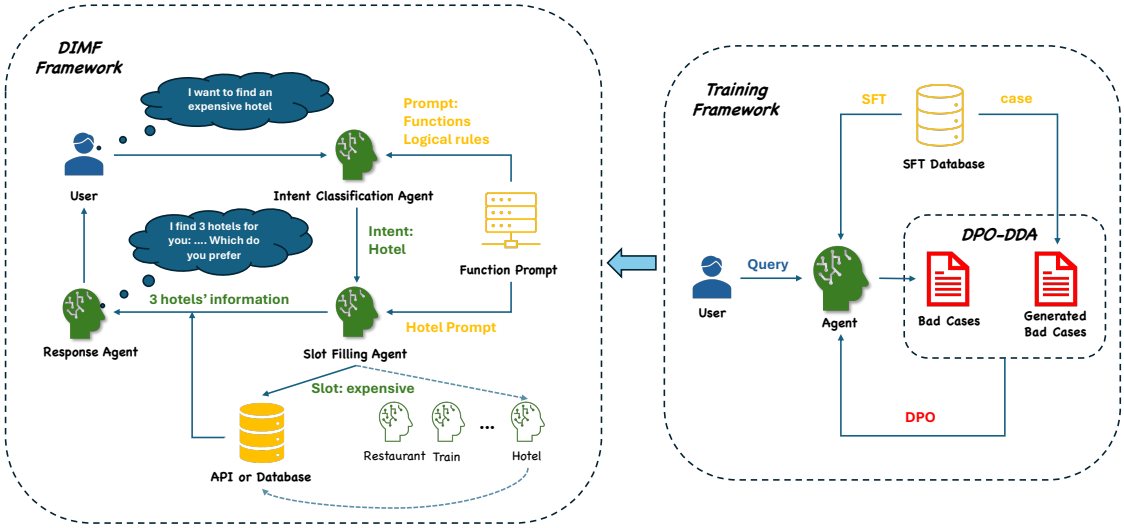

Figure 2: The main framework of our proposed method. Our method contains two parts. The left part is the framework of our proposed DIMF. We train three agents to collaboratively solve users' questions and provide responses. Each agent can fulfill different user needs through different prompts, instead of training domain-specific agents (as indicated by the agents in the left part such as "Restaurant"). The right part is the framework of our training process for each agent. We first fine-tune the model with the training set, and then leverage the validation dataset to complete the DPO process.

3 Domain-Independent Multi-Agent Framework

In this section, we introduce our proposed Domain-Independent Multi-Agent Framework (DIMF) for the TOD task. We give an introduction to the Intent Classification Agent , Slot Filling Agent and Response Agent separately. We will provide a detailed introduction to the division of labor between each agent.

3.1 Intent Classification Agent

The Intent Classification Agent aims to extract the intent of the user's question and serves as the foundation for the subsequent agents. Specifically, this agent is provided with the user's question and the descriptions of each domain, then outputs in the ReACT format. Besides, this task involves the user's follow-up questions regarding historical dialogue. Therefore, we have designed a logic module in the prompt that provides the logical rules in the current round of dialogue based on the intent of the last round. Moreover, we design an "other" domain to implement the dialogue-ending intent . The details of the prompt are appended in Appendix A.1.

3.2 Slot Filling Agent

After obtaining the intent of the user's question from the Intent Classification Agent, we train a Slot Filling Agent to extract slots for the specific domain from the query, which is required for extracting information from the database. This agent can be adapted to various domains through conducting domain-specific prompts. In this way, we can obtain a generalized Slot Filling Agent instead of training different models for different domains.

For the user's questions, there are two different types of slots: 1) The slot with its corresponding value, such as *I need train reservations from Norwich to Cambridge.* which contains the name of the departure and destination. 2) The slot without value, such as *I would also like to know the travel time, price, and departure time please.* which needs to respond the value to the user. We design two modules to respond to these two types of information separately, and provide a logical rules module in the prompt to distinguish between them.

Besides, to address the issue of slot inheritance based on dialogue history, we have also designed a module for the Slot Filling Agent in the prompt that includes historical dialogue slots, allowing the agent to better implement this capability by integrating this information with the dialogue history. Later, according to the generated slot information by the Slot Filling Agent, we can extract the entries in the database that match the user's query. In this work, we use a rule-based approach for extrac-

tion. The detail of the prompt is attached in the Appendix A.2.

3.3 Response Agent

Different dialogue histories and states dictate various strategies, such as asking the user to fill in the required slots, allowing the user to refine results, letting the user confirm or cancel, and so on. The Response Agent aims to respond to the user based on the dialogue history and states. Since the database's results of each query vary, we develop the following strategies for the Response Agent to assist the user in obtaining information about the outcome during conversations.

After calling database, the response strategy depends on the number of database results that meet the user's question. If there is only one option, the agent should respond to the information of a specific item that the user asks directly. Otherwise, the response's content should contain the following information: 1) The total number of available options. 2) The conclusion of all options. 3) The question asking users for more specific information to narrow the range of available options (we have provided these selectable slots in the prompt). The detail of the prompt is attached in the Appendix A.3.

4 Improving DPO Training by Data Distribution Adaptation Method

Since multiple sub-tasks of TOD are executed under limited states, we conducted DPO training after SFT which is more conducive to leveraging the advantages of DPO. However, due to the uncertainty in the distribution of domains in the bad cases, we encountered the degradation issue of DPO mentioned in Section 2.2. We propose a Data Distribution Adaptation (DDA) method to improve the issue simply and effectively.

For the first two agents, their results for one real question are all on a specific domain in formatted structures. Therefore, the DPO method is well-suited to leverage its strengths in this scenario. Besides, both of the agents in our method need to complete the complex logical instructions in the prompt, which faces challenges on lightweight LLMs. The DPO method can further improve the weaknesses in training on these instructions during the SFT phase.

When we directly leverage the DPO method to train on the bad cases in the validation set, we also encountered the issue of model degradation after DPO training, which is mentioned in Section 2.2. We analyze the bad cases and find that, compared to the SFT training data, the rejected data used by DPO had a very uneven distribution in terms of domains. Based on the conclusion that "*the DPO loss function demonstrates a tendency for LLMs to readily learn to avoid generating responses that humans disprefer*" (Feng et al., 2024), we believe that if the category of the rejected data in the DPO phase is concentrated in a certain category, it will significantly reduce the generation probability for that category after training, which leads to model degradation in that category. Therefore, we generate bad cases for other categories to match the distribution of rejected data across all categories with the data from the SFT phase. In this way, we have effectively alleviated the degradation problem caused by DPO.

5 Experimental Setup

5.1 Dataset & Evaluation Metrics

We evaluate our proposed method on the MultiWOZ 2.2 dataset (Zang et al., 2020). The dataset is a large-scale multi-domain TOD dataset which contains 10437 conversations and is divided into training, validation, and test sets. The dataset comprises 7 domains and contains a database for querying the information of a specific domain.

We leverage the traditional evaluation method of the MultiWOZ 2.2 dataset, Inform, Success, and BLEU scores, to evaluate our proposed method. The **Inform** rate is to check whether the system finds the right entity for the user. The **Success** rate is to check whether the system provides all the required entity attributes for the user. The **BLEU** measures the fluency compared to the references, which are delexicalized. Finally, the **Combine** score is a comprehensive metric to indicate the overall performance, which is formulated as: $Combine = \frac{Inform+Success}{2} + BLEU$. Besides, we leverage the Conditional Bigram Entropy (CBE), #unique words and #unique 3-grams to evaluate the richness of the response.

5.2 Baselines

We compare our proposed method with the traditional system and the LLM-based system. We choose several strong baselines fine-tuned on the traditional language models, including GALAXY (He et al., 2022), TOATOD (Bang et al., 2023),

Model	BLEU	Inform	Success	Combined	CBE	#uniq. words	#uniq. 3-grams
Traditional model:							
GALAXY (He et al., 2022)	19.6	85.4	75.7	100.2	1.75	295	2275
TOATOD (Bang et al., 2023)	17.0	90.0	79.8	101.9	-	-	-
Mars-G (Sun et al., 2023)	**19.9**	88.9	78.0	103.4	1.65	288	2264
KRLS (Yu et al., 2023)	19.0	89.2	80.3	103.8	1.90	494	3884
DiactTOD (Wu et al., 2023)	17.5	89.5	84.2	104.4	2.00	418	4477
Large Language Model (LLM):							
Mistral-7B DARD (Gupta et al., 2024)	15.2	78.8	61.2	85.2	2.79	993	13317
Qwen2.5-7B DARD	14.9	80.1	61.5	85.7	2.14	902	12974
SGP-TOD-GPT3.5 (Zhang et al., 2023)	9.2	82.0	72.5	86.5	-	-	-
Claude Sonnet 3.0 DARD (Gupta et al., 2024)	9.5	**95.6**	**88.0**	101.3	2.37	1197	13742
Ours:							
Qwen2.5-7B DIMF w/o DPO	14.8	90.3	75.4	97.7	2.73	1139	14305
Qwen2.5-7B DIMF	18.7	92.4	82.8	**106.3**	**2.81**	**1231**	**14328**

Table 1: End-to-end response generation evaluation results on MultiWOZ 2.2 dataset. All results of traditional models are cited from the official leaderboard. We execute the publicly accessible results of the LLM-based model. The "**bold**" indicates the best score among all the systems of each language pair.

Mars-G (Sun et al., 2023), KRLS (Yu et al., 2023), DiactTOD (Wu et al., 2023). For the LLM-based system, we evaluate the SGP-TOD (Zhang et al., 2023) method which leverages the symbolic knowledge to build a TOD system with GPT3.5. Besides, we compare our method with the state-of-the-art LLM-based method, DARD (Gupta et al., 2024). Since the code was not provided of DARD, we independently replicate the results of the DARD method on the Qwen2.5-7B model.

5.3 Setup

We select Qwen2.5-7B-Instruct[1] (Yang et al., 2024), a representative and common open source LLM as our foundation model. We generate the training dataset tailored to each agent. All the agents are fully fine-tuned and conducted on 8 A100 GPUs with 40GB of RAM for 2 epochs.

6 Experiments

6.1 Main Results

We present the results of our proposed DIMF and other baselines in Table 1. Specifically, each agent in DIMF is first fine-tuned on the entire training set under supervision and then trained using the DPO method on the validation set. The results show that our proposed method achieves the best Combined score among all the baselines.

Compared with the traditional models, DIMF has become more powerful in slot extraction which corresponds to the scores of Inform and Success.

[1] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

Model	BLEU	Inform	Success	Combined
Qwen2.5-7B DIMF	18.7	92.4	82.8	106.3
w/o R. DPO	16.8	91.2	81.3	103.1
w/o R. & S. DPO	14.6	91.2	76.8	98.6
w/o R. & S. & I. DPO	14.8	90.3	75.4	97.7

Table 2: Ablation studies results on our proposed DDA-based DPO method. The R., S. and I. represent Response Agent, Slot Filling Agent and Intent Classification Agent separately. Each row in the table is based on the last row with the DPO method removed.

This also demonstrates that the method of separating the complex tasks in our DIMF can effectively enhance the system's capability. As for the Large Language Model, our model has outperformed the same size model on all evaluation metrics. The results of the DARD method on the Qwen model prove the advancement of our method. Besides, compared to the large-scale LLMs, our method has a significant improvement on the BLEU. Moreover, unlike the DARD method, we use a single model for all domains which demonstrates a better generalization of our method.

The last three metrics evaluate the textual richness of the model response. The results show that our method significantly outperformed other models. This also demonstrates the advantages of LLMs compared to the traditional models: the diversity of responses can provide users with a better interactive experience in real-world scenarios.

6.2 The Impact of DPO Method

In this section, we will evaluate the benefits that the DPO brings to the framework. We first do the abla-

Model	Attraction			Hotel			Restaurant			Taxi			Tarin		
	BLEU	Info.	Succ.	BLEU	Info.	Succ.	BLEU	Info.	Succ.	BLEU	Info.	Succ.	BLEU	Info.	Succ.
Base System (All agents trained with SFT)															
DIMF-base	14.8	98.7	83.2	14.2	89.6	74.8	13.7	96.2	85.3	15.2	100.0	85.1	15.0	90.1	78.1
w/ Intent Classification Agent DPO															
DPO-Ori	11.9	86.3	71.0	13.1	90.0	75.2	12.2	90.2	79.1	12.7	100.0	73.3	15.0	90.5	80.0
DPO-DDA	14.8	99.1	83.7	13.7	90.3	76.7	13.6	96.2	85.3	15.6	100.0	86.0	14.9	91.4	78.4
w/ Intent Classification Agent DPO-DDA & Slot Filling Agent DPO															
DPO-Ori	11.0	81.7	69.4	12.7	80.5	73.1	12.9	83.4	73.3	14.8	100.0	79.1	12.5	79.6	71.9
DPO-DDA	17.1	99.1	90.2	16.2	90.6	83.6	15.9	96.2	89.7	17.1	100.0	88.2	16.7	90.8	83.2
w/ Intent Classification Agent DPO-DDA & Slot Filling Agent DPO-DDA & Response Agent DPO															
DPO-Ori	19.6	99.1	90.2	17.3	91.0	83.1	16.0	96.2	89.0	18.8	100.0	89.6	19.2	92.3	82.7
DPO-DDA	19.4	99.1	90.2	17.7	91.3	84.0	16.3	96.5	89.7	18.6	100.0	89.6	19.5	92.3	83.2

Table 3: Results of different DPO training method on each agent of DIMF. The gray data indicates the degradation data. The DPO-Ori represents the original DPO training method which directly leverage the bad cases for training. The DPO-DDA represents our proposed Data Distribution Adaptation method.

tion tests. Then, we demonstrate the effectiveness of our proposed DDA-based DPO method.

6.2.1 Ablation Studies on DPO

In order to better understand the effect of the DPO training method on each agent , we perform an ablation test and present the results in Table 2. All the results in this section are obtained using our proposed DDA training strategy for DPO. The results show that DPO training improves the accuracy of each stage in the system, thereby alleviating the problem of error accumulation.

As we can see in Table 2, compared to the other two agents, the improvement of DPO in the Intent Classification Agent is limited. We believe this is because the model trained after SFT already possesses relatively good capabilities. However, the Slot Filling Agent and the Response Agent still show significant improvement in the BLEU and Success metrics after our DDA-based DPO training. The experimental results also demonstrate that, compared to other methods, our DIMF approach, which trains the Slot Filling Agent separately and isolates the Response Agent, is very effective in enhancing performance in the TOD system.

6.2.2 Results of Data Distribution Adaptation Method for DPO Training

In this section, we aim to demonstrate that our Data Distribution Adaptation method can effectively mitigate the issue of DPO degradation. The test set contains 5 domains with different numbers (Attraction (396), Hotel (394), Restaurant (437), Taxi (195) and Train (495)). We present the results of each domain in Table 3. We define that if the performance of a specific domain drops below the

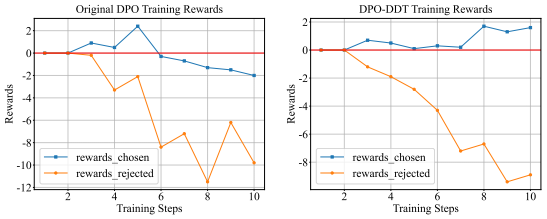

Figure 3: The rewards of the chosen data and rejected data during the Slot Filling Agent DPO training. The left figure is the original DPO method and the right one is our proposed DDA method. The red line represents the reward of 0.

average accuracy, then the model has a degradation issue in that domain. Due to testing issues, the Inform for the Taxi did not change. The distribution of bad cases on the test set is similar to the validation set, so we will directly analyze the results on the test set between the two DPO methods.

Intent Classification Agent: Most of the errors are concentrated in the Hotel and Train domains after SFT training. Therefore, these two domains tend to appear more frequently in the chosen data of the original DPO method. Most of the data in the rejected data set belongs to the other three domains. The results show that the data distribution on the rejected data of the original DPO training method leads to a decrease in the generation probability of other domains. Therefore, we generate bad cases for the other three domains to align with the data distribution of the SFT process.

Slot Filling Agent: During the DPO training phase of Slot Filling Agent, the degradation issue appeared in more domains. We find that many bad cases at this stage occurred when information from multiple rounds of dialogue needed to be inherited.

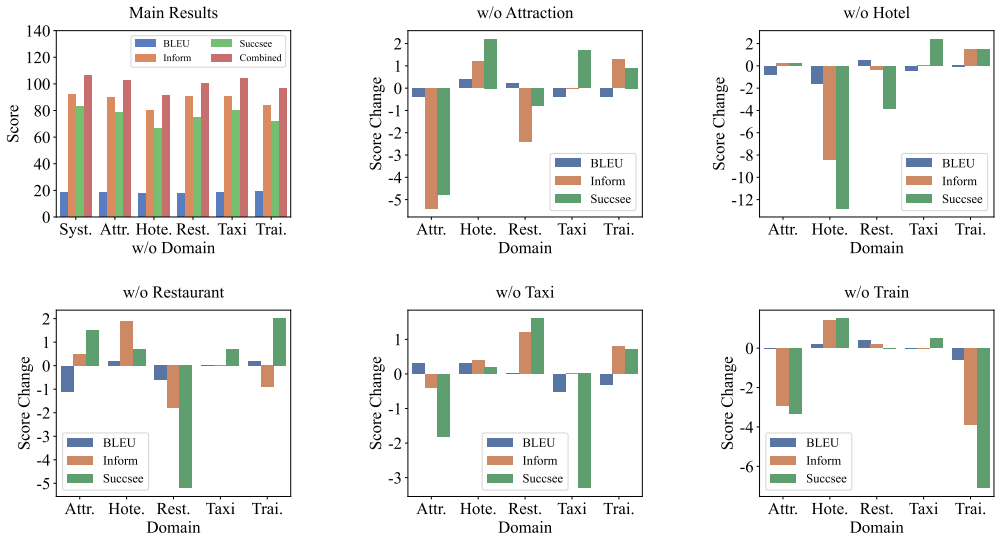

Figure 4: The Results of the DIMF after removing training data from a specific domain. The first sub-figure shows the results of the system after removing different domains. The other sub-figures shows the performance of each domain after removing a specific domain respectively.

These bad cases were very unevenly distributed across different slot categories, such as area, leading to degradation in various domains. Therefore, we generate bad cases for different slots to implement our proposed method.

Response Agent: The degradation issue of DPO is not significant in Response Agent. We select bad cases based on a certain threshold of BLEU and generate bad cases according to the distribution of domains.

Training Rewards: We show the training rewards of the chosen data and rejected data during the DPO training process of the Slot Filling Agent in Figure 3. In an ideal situation, "reward_chosen" should be greater than 0 and increase as training progresses, while "reward_rejected" should be less than 0 and decline. As we can see, the original DPO method encountered issues with the chosen reward decreasing and becoming less than 0. This issue leads to the degradation of the DPO training process, which demonstrates our analysis above. Our proposed DDA method can efficiently address this problem which is shown in the right figure. The experimental results demonstrate the effectiveness of our DDA-based DPO method. The other agents' results are appended in Appendix B.

6.3 Zero-shot Evaluation

We evaluate the zero-shot capabilities of our proposed framework in this section. For each agent in our method, we remove the data of one domain during the training process. We show the performance of the total system and each domain after removing the specific domain in Figure 4.

The first sub-figure presents the results of the system. The x-axis represents the results of the original system and the results after removing the training data of different domains. The results indicate that, except for the Hotel and Train domains, the performance of the system does not have a significant decrease compared to the original system after removing other domains. As for the Hotel and Train, the results in Table 3 show that these two domains are more challenging, and our system performs relatively poorly on them. We believe this is the reason for the decline of performance. Nevertheless, the performance of our proposed method still exceeds the same size LLM in Table 1 in these two experiments. The result demonstrates that our method enhances the generalization ability of the TOD system by refining tasks within the system.

The other sub-figures present the results on each domain after removing different domains. The results indicate that the accuracy of the specific domain decreased after removing its corresponding data, particularly in the Hotel and Train domains, which confirms the analysis in the last paragraph. Besides, we also observed a phenomenon in the experiment that the performance of some other domains declined after removing one domain. We

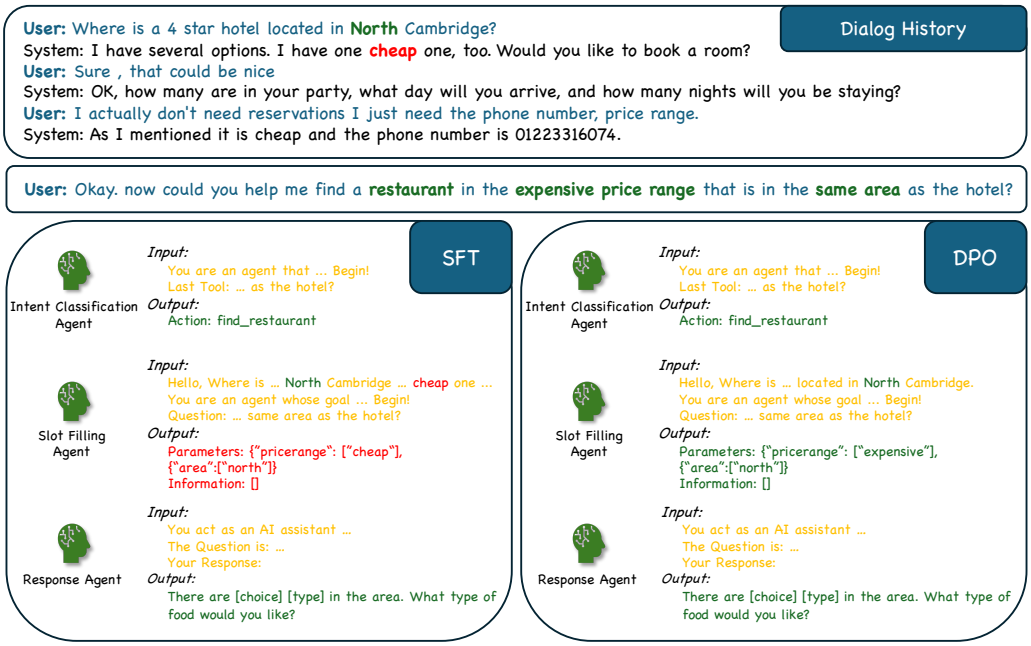

Figure 5: An example of one round of the conversation between user and our DIMF. This case contains the history of the conversation, the question of the user and the generation process of DIMF trained with different methods. The red word represents incorrect information and responses, and green represents correct ones.

think that this may be caused by the reduction in data diversity. Moreover, we find that the zero-shot setting has little impact on the BLEU metric.

6.4 Case Study

To further understand the detailed process of our method, we provide a case study that contains the output of each agent for a specific user's question. We select a more challenging case that requires inheriting information from the historical dialogue.

As shown in Figure 5, when our system receives a user's question, the question first be directly transferred into the Intent Classification Agent without dialogue history to obtain the user's intent. Next, the slot prompt of this specific domain with the dialogue history is input into the Slot Filling Agent to obtain the specific information in this domain that the user needs to inquire about. Finally, the results queried from the database are input into the Response Agent to obtain the response for the user.

In this case, we can see that the user does not specify the specific information in the "area" slot directly. The system needs to inherit this information and remove another irrelevant slot "cheap" from the last intent. The Slot Filling Agent implements this ability by adding the logic rule about inheriting historical dialogue information in the prompt. However, as shown in this case, the lightweight LLMs trained with the SFT method cannot fully learn this capability and sometimes make mistakes on this issue. The DPO method provides targeted training for this capability, effectively improving the shortcomings of the SFT method and improving the system's performance.

7 Conclusion

In this work, we propose a new framework, Domain-Independent Multi-Agent Framework (DIMF), for TOD systems. We separate the original complex task into three sub-tasks, Intent Classification Agent, Slot Filling Agent, and Response Agent, which reduces the complexity of each agent and makes the performance of lightweight LLMs more reliable. Our framework trained on the Qwen2.5-7B achieves better performance compared with all the baselines. Besides, during the training process, we leverage the advantages of the DPO method on this task to address the deficiencies in understanding logical rules in prompts during the SFT process. We propose a Data Distribution Adaptation (DDA) method to mitigate the degradation issues of DPO. The results prove that our method is easy to implement and effective. Moreover, we demonstrate that our system can better utilize the generalization capabilities of LLMs and has a good zero-shot ability.

8 Limitations

In this work, with a carefully designed TOD framework, we have revealed that current systems on TOD tasks severely suffer from insufficient task independence and model scalability. We further propose the DIMF and DDA training method to mitigate the phenomenon. However, our work still has limitations. Firstly, during the tool invocation stage, we directly access the database based on the results of the Slot Filling Agent. When facing more diverse, complex, or real tools, it may be necessary for the model to generate a unified invocation statement to address this issue. Secondly, our current experiment of zero-shot capabilities mainly evaluates the unseen domain in the same dataset with the training data. It would be better to further test this capability on other datasets in subsequent work.

References

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

Atheer Algherairy and Moataz Ahmed. 2024. A review of dialogue systems: current trends and future directions. *Neural Computing and Applications*, 36(12):6325–6351.

Namo Bang, Jeehyun Lee, and Myoung-Wan Koo. 2023. Task-optimized adapters for an end-to-end task-oriented dialogue system. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7355–7369, Toronto, Canada. Association for Computational Linguistics.

Erik Cambria. 2024. Understanding natural language understanding.

Willy Chung, Samuel Cahyawijaya, Bryan Wilie, Holy Lovenia, and Pascale Fung. 2023. Instructtods: Large language models for end-to-end task-oriented dialogue systems. *arXiv preprint arXiv:2310.08885*.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2024. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. 2024. Towards analyzing and understanding the limitations of dpo: A theoretical perspective. *arXiv preprint arXiv:2404.04626*.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. 2023. Towards llm-driven dialogue state tracking. *arXiv preprint arXiv:2310.14970*.

Aman Gupta, Anirudh Ravichandran, Ziji Zhang, Swair Shah, Anurag Beniwal, and Narayanan Sadagopan. 2024. Dard: A multi-agent approach for task-oriented dialog systems. *arXiv preprint arXiv:2411.00427*.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, et al. 2022. Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 10749–10757.

Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, and Milica Gašić. 2023. Chatgpt for zero-shot dialogue state tracking: A solution or an opportunity? *arXiv preprint arXiv:2306.01386*.

Nikitas Karanikolas, Eirini Manga, Nikoletta Samaridi, Eleni Tousidou, and Michael Vassilakopoulos. 2023. Large language models versus natural language understanding and generation. In *Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics*, pages 278–290.

Yangming Li, Kaisheng Yao, Libo Qin, Wanxiang Che, Xiaolong Li, and Ting Liu. 2020. Slot-consistent NLG for task-oriented dialogue systems with iterative rectification network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 97–106, Online. Association for Computational Linguistics.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*.

Baolin Peng, Michel Galley, Pengcheng He, Chris Brockett, Lars Liden, Elnaz Nouri, Zhou Yu, Bill Dolan, and Jianfeng Gao. 2022. Godel: Large-scale pre-training for goal-directed dialog. *arXiv preprint arXiv:2206.11309*.

Libo Qin, Wenbo Pan, Qiguang Chen, Lizi Liao, Zhou Yu, Yue Zhang, Wanxiang Che, and Min Li. 2023a. End-to-end task-oriented dialogue: A survey of

tasks, methods, and future directions. *arXiv preprint arXiv:2311.09008*.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. 2023b. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

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

Weizhou Shen, Chenliang Li, Hongzhan Chen, Ming Yan, Xiaojun Quan, Hehong Chen, Ji Zhang, and Fei Huang. 2024. Small llms are weak tool learners: A multi-llm agent. *Preprint*, arXiv:2401.07324.

Haipeng Sun, Junwei Bao, Youzheng Wu, and Xiaodong He. 2023. Mars: Modeling context & state representations with contrastive learning for end-to-end task-oriented dialog. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11139–11160.

Xiangru Tang, Anni Zou, Zhuosheng Zhang, Ziming Li, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. 2023. Medagents: Large language models as collaborators for zero-shot medical reasoning. *arXiv preprint arXiv:2311.10537*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghuai Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. 2023. Making large language models better reasoners with alignment. *arXiv preprint arXiv:2309.02144*.

Qingyang Wu, James Gung, Raphael Shu, and Yi Zhang. 2023. Diacttod: Learning generalizable latent dialogue acts for controllable task-oriented dialogue systems. *arXiv preprint arXiv:2308.00878*.

Heng-Da Xu, Xian-Ling Mao, Puhai Yang, Fanshu Sun, and He-Yan Huang. 2024. Rethinking task-oriented dialogue systems: From complex modularity to zero-shot autonomous agent. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2748–2763.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Xiao Yu, Qingyang Wu, Kun Qian, and Zhou Yu. 2023. Krls: Improving end-to-end response generation in task oriented dialog with reinforced keywords learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12338–12358.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. *arXiv preprint arXiv:2007.12720*.

Xiaoying Zhang, Baolin Peng, Kun Li, Jingyan Zhou, and Helen Meng. 2023. SGP-TOD: Building task bots effortlessly via schema-guided LLM prompting. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13348–13369, Singapore. Association for Computational Linguistics.

Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. 2020. Recent advances and challenges in task-oriented dialog systems. *Science China Technological Sciences*, 63(10):2011–2027.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. 2023. Sotopia: Interactive evaluation for social intelligence in language agents. *arXiv preprint arXiv:2310.11667*.

A Prompt

A.1 Prompt of Intent Classification Agent

We show an example of the Intent Classification Agent at the second-round of the conversation in Table A.1.

A.2 Prompt of Slot Filling Agent

We show an example of the Slot Filling Agent of the restaurant domain at the second-round of the conversation in Table A.2.

A.3 Prompt of Response Agent

We show an example of the Response Agent in Table A.3.

B DPO Training Loss

We present the results of the reward loss of the Intent Classification Agent and Response Agent in Figure 6 and Figure 7. Compared to Slot Filling Agent, the degradation issues on the original DPO method are not as severe for these two models. The Intent Classification Agent experienced a reduction in chosen reward, while the training of the Response Agent was relatively normal.

Figure 6: The rewards of the chosen data and rejected data during the Intent Classification Agent DPO training.

Figure 7: The rewards of the chosen data and rejected data during the Response Agent DPO training.

Table 4: Intent Classification Agent prompt

You are an agent that helps users choose the right tool or tools from the given tools list to solve their problems.

For each tool, you are first given its description and required parameters. Then, a logic module specifically explains the logical information needed for this tool to handle multi-turn conversation issues.

Tool APIs

find_hotel: search for a hotel to stay in
book_hotel: book a hotel to stay in
find_train: search for trains that take you places
book_train: book train tickets
find_attraction: search for places to see for leisure
find_restaurant: search for places to wine and dine
book_restaurant: book a table at a restaurant
find_hospital: search for a medical facility or a doctor
find_taxi: find or book taxis to travel between places
find_bus: search for a bus
find_police: search for police station
other: This tool is used to handle problems that cannot be addressed by any other tools.

Task Logic
If last query is find_restaurant, the user can use the same tool for the following types of query:
- restaurant-pricerange: price budget for the restaurant. only allowed values: [cheap, expensive, moderate]
- restaurant-area: area or place of the restaurant. only allowed values: [centre, east, north, south, west]
- restaurant-food: the cuisine of the restaurant you are looking for.
- restaurant-name: name of the restaurant.
- restaurant-bookday: day of the restaurant booking. only allowed values:
    [monday, tuesday, wednesday, thursday, friday, saturday, sunday]
- restaurant-bookpeople: how many people for the restaurant reservation. only allowed values: [1, 2, 3, 4, 5, 6, 7, 8]
- restaurant-booktime: time of the restaurant booking.

Output Format

Use the following format:

Last Tool: the tool used in last query
Question: the input question you must answer
Action: the action to take
Finish!

Begin!

Last Tool: find_restaurant
Question: Any sort of food would be fine. Could I get the phone number for your recommendation?

Table 5: Slot Filling Agent Filling prompt

You are an agent whose goal is to extract the required tool parameters and the content the user wants to query from their questions.

For a specific query, you are first given the parameters corresponding to the restaurant tool. Besides, you have also been informed the information that the specific information this tool can query. Finally, you are given the logic distinguish between Tool Parameters and Tool Information.

Tool Parameters

restaurant-pricerange: price budget for the restaurant. only allowed values: [cheap, expensive, moderate]
restaurant-area: area or place of the restaurant. only allowed values: [centre, east, north, south, west]
restaurant-food: the cuisine of the restaurant you are looking for.
restaurant-name: name of the restaurant.
restaurant-bookday: day of the restaurant booking. only allowed values: [monday, tuesday, wednesday, thursday, friday, saturday, sunday]
restaurant-bookpeople: how many people for the restaurant reservation. only allowed values: [1, 2, 3, 4, 5, 6, 7, 8]
restaurant-booktime: time of the restaurant booking.

Tool Information

The user can use restaurant tool to query the following questions:
address: the address of the restaurant.
area: the location information of the restaurant can be selected from the following options: [east, south, west, north].
food: the food of the restaurant.
id: the id number of the restaurant.
introduction: the introduction of the restaurant.
location: the coordinates of the restaurant.
name: the name of the restaurant.
phone: the phone of the.
postcode: the postcode of the restaurant.
pricerange: the level of the price of the restaurant.
type: .

Task Logic

- If the user's question includes a slot name and the slot value, then this query information belongs to the tool Parameters, and output must in a JSON type.
- If the user's question only includes a slot name without value, then this query information belongs to the tool Information.
- If the user needs information from the historical conversation, you can obtain it from the History Conversation slot.

History Conversation slot

restaurant:
"area": ["centre"], "pricerange": ["expensive"]

Output Format

Use the following format:

Question: the input question you must answer
Action: the tool that user used
Parameters: must a JSON object of the slot with its value
Information: the tool information in a list object
Finish!

Begin!

Question: Any sort of food would be fine, as long as it is a bit expensive. Could I get the phone number for your recommendation?
Action: restaurant

Table 6: Response Agent prompt

You act as an AI assistant to reponse user's question relied some given informations.
You should always communicate with the user in the first person and respond in a personified manner.
The Question is: I need train reservations from norwich to cambridge

Responce Rules

You should respond according to the following rules:

Make a conclusion based on the the user's question, Observation and conversation history. If there are several options, you can first respond the total number of the option, make a conclusion of the "conclusion informations" and then ask the question about the informations in "question content"
- example: "I have xxx options matching your request. Waht's the xxx you want to xxx"
- example with conclusion informations: "I have xxx options matching your request. The range of xxx in these options is xxx. Waht's the xxx you want to xxx"
If there is only one options, you can make a conclusion if it and respond to the user.
All the specific information in the response should be in this format: [type_name]

Observation

train information:
option number: 133
question content: arriveby, leaveat, trainid, day, price
conclusion informations:
arriveby: 06:35, 07:35, 08:35, 09:35, 21:35, 22:35, 23:35, 24:35
leaveat: 05:16, 06:16, 07:16, 08:16, 20:16, 21:16, 22:16, 23:16

Note

You should respond with more varied expressions.
Your respond should contain all the information in Observation, and your reply should no more than 25 words.

Your Response: