Spec-TOD: A Specialized Instruction-Tuned LLM Framework for Efficient Task-Oriented Dialogue Systems

Quang-Vinh Nguyen*, **Quang-Chieu Nguyen***, **Hoang Pham, Khac-Hoai Nam Bui**†
Viettel Artificial Intelligence and Data Services Center,
Viettel Group, Vietnam
{vinhnq29,chieunq, hoangpv4, nambkh}@viettel.com.vn

Abstract

Task-oriented dialogue (TOD) systems facilitate goal-driven interactions between users and machines. While recent advances in deep learning have improved the performance, TOD systems often struggle in low-resource scenarios with limited labeled data. To address this challenge, we propose Spec-TOD, a novel framework designed to train an end-to-end TOD system with limited data. Spec-TOD introduces two main innovations: (i) a novel specialized end-to-end TOD framework that incorporates explicit task instructions for instruction-tuned large language models (LLMs), and (ii) an efficient training strategy that leverages lightweight, specialized LLMs to achieve strong performance with minimal supervision. Experiments on the MultiWOZ dataset, a widely used TOD benchmark, demonstrate that Spec-TOD achieves competitive results while significantly reducing the need for labeled data. These findings highlight the potential of the proposed framework in advancing efficient and effective TOD systems in low-resource settings.

1 Introduction

Task-oriented dialogue (TOD) systems have emerged as a critical area of research in natural language processing (NLP), integrating multiple functionalities, such as understanding user input, dialogue state tracking, and response generation to effectively support users in accomplishing specific objectives (Huang et al., 2020). Nevertheless, the development of these systems continues to pose significant challenges (Ni et al., 2023). One of the primary obstacles in creating an effective TOD system is the requirement for extensive annotation and frequent retraining (Qin et al., 2023). Specifically, prevailing TOD models either fine-tune pre-trained language models such as GPT-2 (Yang et al., 2021) and T5 (Lee, 2021; Bang et al., 2023) or prepare high-quality data curation for training pre-trained TOD language models (He et al., 2022; Su et al., 2022), can achieve strong performance on TOD tasks (Figure 1 (a)). The reliance on domain-specific annotated datasets poses challenges for reproducibility and scalability (Bao et al., 2023). Despite progress in few-shot learning (Mi et al., 2022; Moradshahi et al., 2023), current

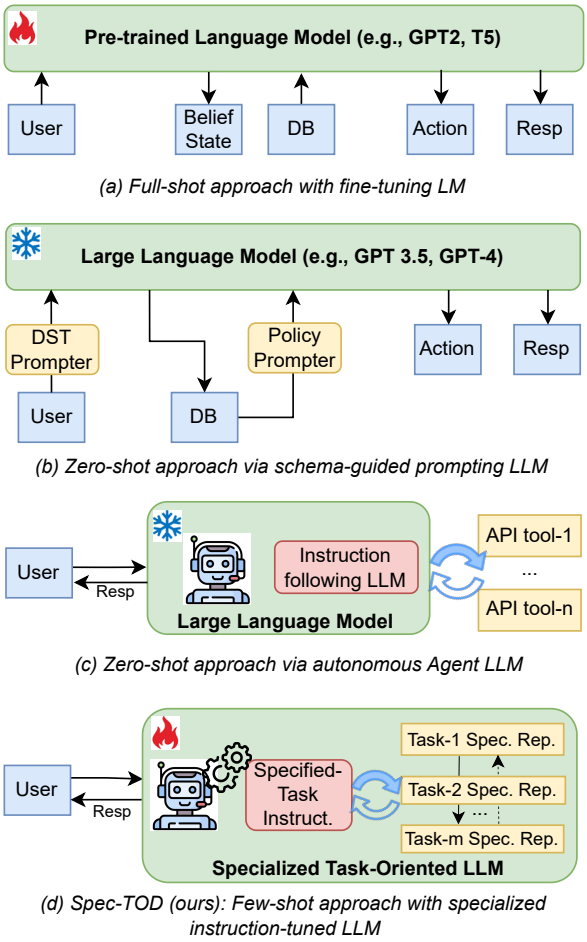

Figure 1: Overview of end-to-end TOD Approaches: (a) traditional full-shot fine-tuning; (b) zero-shot LLM; (c) zero-shot Agent LLM; and (d) few-shot with instruction-tuned LLM (ours).

*Equal contribution.
†Corresponding author.

Proceedings of the 26th Annual Meeting of SIGDIAL, pages 133–145
Aug 25–27, 2025. ©2025 SIGDIAL

methods struggle to generalize across new domains and remain suboptimal. Recent advancements in large language models (LLMs), including proprietary models (e.g., GPT series (OpenAI, 2023)) and open-source models (e.g., Llama series (Touvron et al., 2023)), have transformed NLP tasks by enabling zero-shot and few-shot generalization across various tasks, including the development of LLM-powered TOD systems (Figure 1 (b)). Furthermore, emerging agent-based LLM technologies have reshaped multi-task learning in TOD systems by leveraging general-purpose, instruction-following models (e.g., GPT-4) that autonomously execute tasks through predefined external APIs (Xu et al., 2024). Nonetheless, current LLM-based approaches, including agentic LLMs, encounter two key limitations in end-to-end TOD systems: (i) The efficacy of zero-shot prompting, whether guided by schema or driven by API-based task specifications in agent-based architectures, is heavily contingent on the underlying LLM's capabilities. Competitive performance typically necessitates extremely large models, often with hundreds of billions of parameters. Such models entail significant computational costs, pose challenges for local deployment, and limit accessibility; (ii) Relying solely on instructional prompts to execute all TOD tasks, such as dialogue state tracking (DST) and natural language generation (NLG), may lack adaptability. This issue often results in suboptimal performance, especially in complex dialogue scenarios (Feng et al., 2023). Building on this analysis of current challenges, we formulate the following research question:

How can the performance of end-to-end TOD systems be improved by leveraging trainable LLMs with limited labeled data?

To address this question, this study proposes Spec-TOD, a novel framework designed to optimize task-oriented dialogue systems by utilizing trainable LLMs for unifying multi-specified tasks via instruction tuning under constraints of limited labeled datasets. The main idea is to specify individual tasks with tailored instructions, enabling unified multi-task learning via instruction-tuned LLMs. Accordingly, Spec-TOD leverages multiple tasks of end-to-end TOD systems with specified instruction representations and a parameter-efficient tuning technique, enabling it to achieve competitive performance while utilizing much smaller models and limited computational resources. The primary contributions of this study are threefold:

- We introduce Spec-TOD, a novel framework tailored for TOD systems, leveraging trainable LLMs with task-specific instructions. To the best of our knowledge, Spec-TOD represents the first LLM-based framework explicitly designed for trainable end-to-end TOD systems.
- Spec-TOD employs parameter-efficient fine-tuning techniques with lightweight, open-source LLMs, enabling adaptability and cost-effective scalability for high-demand applications. The source code is available for further exploitation[1].
- We conduct extensive evaluations on multiple MultiWOZ versions, a widely used TOD benchmark. The results achieve promising performances compared with previous approaches, including full fine-tuning approaches with pre-trained language models and zero-shot approaches with large-scale LLMs. These findings highlight Spec-TOD's potential to advance research in TOD systems.

2 Literature Review

2.1 End-to-end task-oriented dialogue systems

The traditional method follows a modular pipeline architecture, which is divided into four separate components: natural language understanding (NLU), dialogue state tracking (DST), dialogue policy learning (DPL), and natural language generation (NLG). However, this approach cannot leverage shared knowledge across all modules. Furthermore, the errors in earlier modules are propagated to later ones, leading to error (Qin et al., 2023). To address this, the emerging models tend to shift

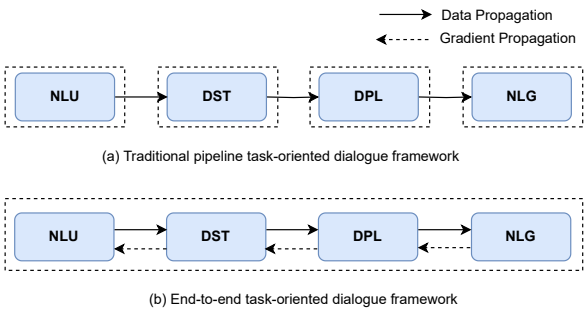

Figure 2: Overview of TOD training methods. The solid line arrows and the dashed arrows represent data propagation and gradient propagation, respectively.

to end-to-end TOD methods, as illustrated in the

[1]https://github.com/quangvinh2110/Spec-TOD

Figure 2. A key difference is that the end-to-end TOD methods can train all the components simultaneously by utilizing pre-trained language models (e.g., GPT-2 and T5) (Yang et al., 2021, 2022; Bang et al., 2023; Chieu et al., 2024). Accordingly, those models are typically developed by fine-tuning the pre-trained language models to learn task-agnostic language representations on specific data.

2.2 LLM-powered zero-shot and few-shot TOD systems

Despite the promising results achieved through full fine-tuning of pre-trained language models, a significant challenge persists for current TOD systems: their development and maintenance demand substantial human annotation efforts for real-world deployment (Yang et al., 2022). To address this issue, several studies have explored few-shot learning approaches to enhance the data efficiency of TOD systems (Lin et al., 2020; Mi et al., 2022; Su et al., 2022). However, these methods often struggle to generalize effectively to unseen domains, limiting their broader applicability (Imrattanatrai and Fukuda, 2023).

Sequentially, the advent of LLMs has transformed this research landscape by leveraging instruction-following techniques. InstructTODS (Chung et al., 2023), one of the earliest LLM-powered TOD frameworks, employs in-context learning (ICL) to enable zero-shot, end-to-end TOD systems, offering improved flexibility and scalability for adapting to new domains. Subsequently, SGP-TOD (Zhang et al., 2023), which operates on predefined task schemas (e.g., dialogue state tracking and natural language generation), introduces an instruction-following approach by utilizing zero-shot or few-shot prompting with several examples to enhance performance by providing comprehensive task-specific information. More recently, AutoTOD (Xu et al., 2024), an agent-based architecture, was proposed to autonomously execute tasks using instruction prompts and external predefined task APIs. However, the heavy reliance on underlying LLMs, which are often difficult to control, results in performance inferior to that of fully fine-tuned traditional approaches, particularly in complex scenarios (Hua et al., 2023).

To address these limitations, this study proposes a trainable, lightweight LLM framework tailored for end-to-end TOD systems. By integrating efficient training strategies and operating with limited labeled data, the proposed framework achieves competitive performance compared to existing approaches in this research field.

3 Methodology

This section establishes the foundation for our work by first providing background information on end-to-end TOD systems. Subsequently, we delve into the details of our proposed framework.

3.1 End-to-end TOD formulation

In traditional end-to-end TOD, at each turn t, the DST module summarizes the dialogue state (Belief State) B_t given the dialogue context C_t:

$$C_t = \{(U_1, B_1, R_1), .., (U_{t-1}, B_{t-1}, R_{t-1}), U_t\} \tag{1}$$

where U_i and R_i is user utterance and system response at turn i. The belief state B_t is defined as the concatenation of the domain/task (i.e., user intent) d_t and a set of slot-value pairs

$$B_t = d_t \oplus \{(s_1, v_1), ..., (s_n, v_n)\} \tag{2}$$

where n denotes the total number of pairs in the set. The output belief states B_t is then put into the NLG module for generating appropriate action A_t and response R_t based on database state DB_t:

$$\begin{aligned} DB_t &= QUERY(B_t) \\ A_t, R_t &= NLG(C_t, DB_1^t, A_1^{t-1}) \end{aligned} \tag{3}$$

Intuitively, the traditional formulation requires a substantial amount of labeled data across multiple tasks to ensure sufficient training coverage and achieve strong performance. To address this, we propose a new framework that specifies multiple tasks in the TOD system by reformulating each task with a dedicated instruction, enabling effective learning through few-shot fine-tuning.

3.2 Spec-TOD Framework

Figure 3 provides an overview of the proposed SpecTOD framework. The framework reformulates the four primary tasks of task-oriented dialogue (TOD) systems into a unified representation, enabling a cohesive multi-task learning approach through instruction-tuned LLMs. The core idea is to conceptualize multi-task learning in end-to-end TOD systems as a function-calling paradigm. Training samples are reformatted to align with instruction-based templates, drawing inspiration from chat-tuned LLMs (Li et al., 2024). Accordingly, each dialogue domain is represented as a distinct function, with slot values within the domain

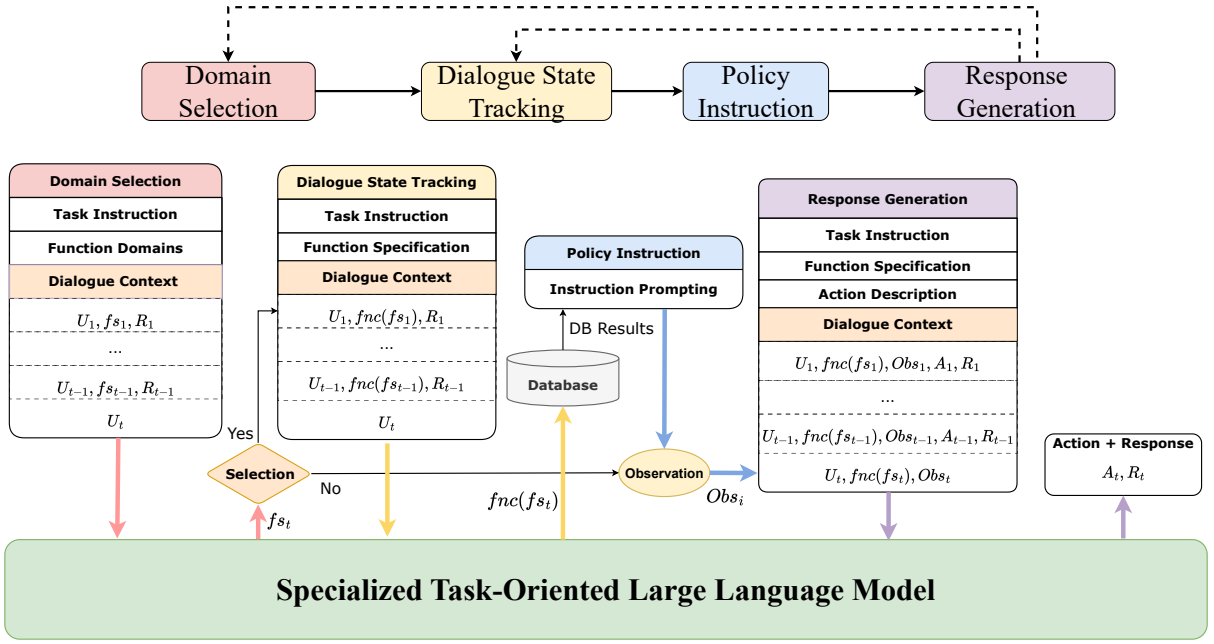

Figure 3: Overview of the Spec-TOD framework, which includes four main specified tasks: Domain Selection, Dialogue State Tracking, Policy Instruction, and Generation Response. Accordingly, except for Policy Instruction is executed with predefined prompt instructions, other task functions are executed and updated with few-shot learning via instruction tuning LLMs.

serving as function arguments. Technically, Spec-TOD integrates four core components: Domain Selection (DS), Dialogue State Tracking (DST), Policy Instruction (PI), and Response Generation (RG). These components are sequentially described as follows:

3.2.1 Domain Selection

Consider a set of functions representing distinct domains, denoted by $\mathcal{F} = \{f_0, f_1, \ldots, f_n\}$. Each function $f_i \in \mathcal{F}$ corresponds to a specific domain (e.g., restaurant, taxi, hotel...) and is defined by three key attributes: a unique identifier, a brief domain description, and a set of arguments. Formally, a function f_i is structured as follows:

```
f_i = {
"name": name of domain;
"description": domain description;
"arguments":{
"slot_name": name of slot;
"type": type of data;
"description": a textual explanation;
"possible_values": list of values}
}
```

The dialogue context for the domain selection task at turn t, denoted by $C_t(DS)$, encapsulates the sequence of interactions up to the current turn, which is re-formatted as follows:

$$C_t(DS) = \{(U_1, fs_1, R_1), .., (U_{t-1}, fs_{t-1}, R_{t-1}), U_t\} \quad (4)$$

where U_i represents the user utterance at turn i, $f_{s_i} \in \mathcal{F}$ denotes the selected domain at turn i, and R_i signifies the system response at turn i. The term U_t indicates the current user's utterance at turn t. The domain selection task at turn t involves determining the appropriate domain f_{s_t} based on the dialogue context and task instructions. This process is modeled as:

$$f_{s_t} = \text{LLM}\left(\text{task\_inst\_DS} \oplus \mathcal{F} \oplus C_t(DS)\right) \quad (5)$$

where task_inst_DS refers to the instruction prompt for the domain selection task (detailed in the Appendix A.2), $\oplus$ denotes the concatenation operation. The output f_{s_t} is the selected domain at turn t, which serves as input for the subsequent dialogue state tracking task.

3.2.2 Dialogue State Tracking

The objective is to generate arguments for the domain function f_{s_t} in JSON format. Specifically, the dialogue context for the DST task is reformatted as follows:

$$C_t(DST) = \{(U_1, fnc(f_{s_1}), R_1) , .., (U_{t-1}, fnc(f_{s_{t-1}}), R_{t-1}), U_t\} \quad (6)$$

where $fnc(f_s)$ represents the structured output of the DST task from previous turns, defined as:

$$\begin{aligned}&\texttt{fnc(}fs_i\texttt{)= \{}\\&\texttt{"name": }fs_i\texttt{;}\\&\texttt{"argument":\{(}s_1\texttt{,}v_1\texttt{),..,(}s_n\texttt{,}v_n\texttt{)\}\}}\end{aligned}$$

where (s_i, v_i) denotes slot-value pair. In this regard, the DST task is executed as follows:

$$\begin{aligned}fnc(f_{s_t}) = \\ LLM(task\_inst\_DST \oplus f_{s_t} \oplus C_t(DST))\end{aligned} \quad (7)$$

where $task\_inst\_DST$ refers to the instruction prompt for the DST task, as illustrated in the Appendix A.2.

3.2.3 Policy Instruction

Following the execution of the DST task, defined as $fnc(f_{s_t})$, on the database, the system generates an observation denoted as Obs_t, which encapsulates the search results. This observation can manifest in two distinct forms. Firstly, Obs_t may quantify the number of entities that fulfill the constraints specified by the function. Secondly, if the function remains unchanged between consecutive time steps (i.e., $fnc_{t-1} = fnc_t$) or if the function selected at time step t is the null function (i.e., $f_{s_t} = f_0$), then Obs_t may indicate that no function call is required, denoted as "Do not need to call function."

3.2.4 Response Generation

Based on the output of the DST task and the observation from the database, the system generates the action and response. The dialogue context of the task is formatted as follows:

$$\begin{aligned}C_t(GR) = \{(U_1, fnc(f_{s_1}), Obs_1, A_1, R_1),\\ .., (U_{t-1}, fnc(f_{s_{t-1}}), Obs_{t-1}, A_{t-1}, R_{t-1}),\\ (U_t, fnc(f_{s_t}), Obs_t)\}\end{aligned} \quad (8)$$

where A_i and R_i denote the action and response of the system in the previous turn i, respectively. Sequentially, the task is executed as follows:

$$\begin{aligned}A_t, R_t = LLM(task\_inst\_GR \oplus Act_{des}\\ \oplus fnc(f_{s_t}) \oplus C_t(GR))\end{aligned} \quad (9)$$

where $task\_inst\_GR$ denotes the prompt instruction of the response generation task (illustrated in the Appendix A.2). The response has to follow the predefined action description Act_{des}, which is defined in each schema of the specific domain (Lee et al., 2022). In this study, based on the schema of the benchmark datasets (Budzianowski et al., 2018), we specify the six descriptions into three typical actions, which are formulated as follows:

$$\begin{aligned}Act_{des} =< Info; Request; NoOffer;\\ Recommend; Select; General >\end{aligned} \quad (10)$$

where:

- **Info**: provide information about an entity (if multiple matched results exist, choose one) in the form of [value_xxx] if requested by the user.
- **Request**: inform the number of available offers ([value_choice]) and ask the user for more preferences on the requested entity to narrow down the search results.
- **NoOffer**: inform the user that no suitable offer could be found.
- **Recommend**: recommend an offer to the user and provide its information.
- **Select**: inform the user to select among options.
- **General**: greet and welcome the user, and inquire if there are anything further requirements.

3.3 Instruction-Tuned LLM Strategy

The primary goal of the proposed framework is to enhance the efficiency of multi-task learning in end-to-end TOD systems by fine-tuning an LLM with tailored instructions (Xin et al., 2024). To achieve this, we restructured dialogue samples into six distinct roles: system, user, domain, function, observation, and assistant. Rather than applying a uniform loss function across all tasks, we designated specific loss calculations to individual roles corresponding to each task. An illustrative example of the re-formatted dialogue sample, highlighting the six roles, is shown in Figure 4. The loss function is computed by aggregating the losses derived from distinct tasks. Specifically, domain selection employs the "domain" role within the loss function framework. For dialogue state tracking, the loss function is defined for the "function" role, which is tasked with generating the function's arguments. In response generation tasks, the loss is associated with the "assistant" role, encompassing both action selection and response generation for the user.

Role System:
You are a task-oriented dialogue system ....
<function_list>
[
{
"name": hotel
"description": hotel reservations and vacation stays
"arguments": {...}
}
{
"name": restaurant
"description": find places to dine and whet your appetite
"arguments": {...}
}
...
]
</function_list>
<function_specification>
{
"name": find_book_hotel
"description": hotel reservations and vacation stays.
"arguments": [
{
"name": pricerange,
"type": string,
"description": price budget of the hotel,
"possible_values": [expensive, cheap, moderate]
}
...
]
}
</function_specification>
Action description:
Name_1: Inform
Description_1: Provide information about an entity...
Name_2: Request
...

Role User:
I need to find mid-range price hotel on the south side?
Role Domain:
name: hotel
Role Function:
name: hotel, arguments: {area: south, pricerange: moderate}
Role Observation:
There are 2 entities match
Role Assistant:
Action: [hotel]: {[inform]: [choice,name,type], [select]}
Response: There are 2 hotel that fit you criteria. [value_name] and [value_name]. Which would you prefer?

Figure 4: An example of our prompt construction with 6 roles: System, User, Domain, Function, Observation, and Assistant. The system prompt itself is comprised of four descriptions: (i) the end-to-end task, (ii) the list of function domains (for domain selection task), (iii) the selected function specification, and (iv) the action description.

The overall loss function for the instruction tuning process is formulated as a composite of the aforementioned task-specific loss functions:

$$\mathcal{L} = \mathcal{L}_{FS} + \mathcal{L}_{FC} + \mathcal{L}_{RG} \quad (11)$$

4 Experiments

4.1 Datasets and Implementation Setting

For the benchmark dataset, we leverage various versions of the MultiWOZ dataset (2.0, 2.1, and 2.2), which contain multi-turn and multi-domain dialogues (Budzianowski et al., 2018; Eric et al., 2020; Zang et al., 2020; Ye et al., 2022). Each dataset version comprises 8438, 1000, and 1000 samples for training, development, and testing, respectively. The primary distinction between these versions lies in their dialogue annotation, with later versions addressing errors and inconsistencies present in earlier iterations. In the configuration, we employ LLaMA-3-8B (Grattafiori et al., 2024), as the lightweight backbone LLM model, and instruction-tuned with Low-Rank Adaptation (LoRA) using a small rank alpha. Our fine-tuning was restricted to the q_proj and v_proj modules. The fine-tuning process spanned four epochs, with a peak learning rate of 3×10^{-4} and a global batch size of 8. The context length is limited to 4096 tokens. As for evaluation, we follow previous works by adopting a combination of **Inform**, **Success**, and **BLEU** scores. The inform rate assesses the system's accuracy in providing relevant entities, while the success rate measures its ability to fulfill all user-requested slots. The combined score can be formulated as follows:

$$Combined = BLEU + 0.5 * (Inform + Success) \quad (12)$$

Furthermore, the Joint Goal Accuracy (**JGA**) metric is used as the benchmark measurement for the DST task.

4.2 Main Results

Table 1 presents the main results of our proposed framework compared to state-of-the-art TOD models under various training data settings. We derive the following key insights based on these results:

i) **Spec-TOD consistently outperforms prior few-shot models**: The experiment on MultiWOZ 2.0 datasets for few-shot strategies indicates the promising results of the proposed method, particularly in terms of Success and Inform scores, which are critical for evaluating task completion and dialogue informativeness. Compared to models like MinTL, PPTOD, and Mars-G, Spec-TOD demonstrates a significant improvement. For example, using only 10% of the training data, Spec-TOD-LLaMA-3-8B achieves 75.5% Success and 86.0% Inform on MultiWOZ 2.0, while the best of the few-shot baselines (Mars-G) only reaches 55.3% Success and 69.4% Inform. This indicates that Spec-TOD is more effective at learning how to complete user goals and convey correct informa-

Model	MultiWOZ 2.0				MultiWOZ 2.2			
	BLEU	Success	Inform	Combined	BLEU	Success	Inform	Combined
Full-shot fine-tuning								
SimpleTOD (Hosseini-Asl et al., 2020)	15.0	70.1	84.4	92.2	-	-	-	-
UBAR (Yang et al., 2021)	17.0	80.7	95.4	105.1	17.6	70.3	83.4	94.4
GALAXY (He et al., 2022)	20.8	84.9	93.5	110.0	19.6	75.7	85.4	100.2
TOATOD (Sun et al., 2023)	-	-	-	-	17.0	79.8	90.0	101.9
Mars (Sun et al., 2023)	19.9	78.0	88.9	103.4	19.6	78.0	88.9	103.0
Zero-shot prompting with LLMs (e.g., Llama-2-70B, GPT-3.5 and GPT-4)								
SGP-TOD-GPT-3.5 (Zhang et al., 2023)	9.0	69.8	83.8	85.9	9.2	72.5	82.0	86.4
AutoTOD-GPT-4 (Xu et al., 2024)	10.4	84.4	91.7	98.5	-	-	-	-
AutoTOD-Llama-2-70B (Xu et al., 2024)	7.8	69.8	73.3	79.4	-	-	-	-
Few-shot with pretrained TOD models								
MinTL (Lin et al., 2020) (10%)	15.6	55.5	44.9	65.8	-	-	-	-
PPTOD (Su et al., 2022) (10%)	15.7	53.7	68.3	76.7	-	-	-	-
Mars-G (Sun et al., 2023) (10%)	15.6	55.3	69.4	78.0	-	-	-	-
Few-shot with Spec-TOD Framework (Ours)								
Spec-TOD-LLama-3-8B (1%)	7.41	63.2	73.0	75.5	6.6	59.5	75.7	74.2
Spec-TOD-LLama-3-8B (5%)	9.1	63.9	79.2	80.6	9.2	66.4	80.5	82.6
Spec-TOD-LLama-3-8B (10%)	10.4	75.5	86.0	91.2	10.4	77.1	87.2	92.6

Table 1: Performance comparison on few-shot End-to-end benchmark on MultiWOZ 2.0 and MultiWOZ 2.2 across various training samples of the datasets. The difference in mean is statistically significant ($p < 0.01$).

tion with limited data.

ii) **Spec-TOD matches full-shot models in Success and Inform scores**: The proposed method achieves comparable or even better performance in some cases, despite using only 10% of the data. For instance, on MultiWOZ 2.2, Spec-TOD reaches 77.1% Success and 87.2% Inform, which achieves competitive results with full-shot models such as GALAXY (75.7% Success, 85.4% Inform). This highlights the data efficiency of Spec-TOD, achieving strong generalization with minimal examples.

iii) **Spec-TOD surpasses zero-shot prompting with large-scale LLMs in domain-specific tasks:** Despite their scale and general reasoning capabilities, LLMs such as GPT-3.5, GPT-4, and LLaMA-2-70B (with the exception of AutoTOD with GPT-4) underperform in domain-specific TOD. These results indicate that prompting alone cannot fully capture the structured behavior needed for such tasks, whereas Spec-TOD's lightweight adaptation more effectively grounds LLMs in domain-specific contexts.

iv) **Spec-TOD demonstrates strong generalization capabilities across datasets**: Despite the increased complexity and annotation consistency in MultiWOZ 2.2, Spec-TOD maintains or improves its performance when transitioning from MultiWOZ 2.0. With 10% data, it improves from 75.5% to 77.1% in Success and 86% to 87.2% in Inform from MultiWOZ 2.0 to 2.2. This robustness to data distribution shifts underscores the flexibility of the Spec-TOD framework.

v) **Spec-TOD remains highly effective in extremely low-resource settings**: Even using only 1% of the training data, Spec-TOD achieves 63.2% Success and 73.0% Inform on MultiWOZ 2.0, performing comparably to some larger zero-shot LLMs such as AutoTOD-LLaMA-2-70B (Success 69.8%, Inform 73.3%). Performance improves steadily with more data (e.g., at 5% data samples: 63.9% Success, 79.2% Inform), demonstrating graceful scalability and reinforcing its practicality for domains with limited annotation budgets.

Criteria/Model	PPTOD	Spec-TOD
Understandable (↑)	3.66	4.3
Relevant (↑)	3.29	4.1
Correct (↑)	2.76	3.75
Appropriate (↑)	3.42	4.15
Fluently (↑)	3.54	4.18
Direct (↑)	2.51	3.32

Table 2: Evaluation of GPT-score ranging from 0 to 5 for PPTOD and Spec-TOD across various criteria, using 10% of the training dataset. Detailed descriptions of each criterion are provided in the Appendix A.1.

Observationally, the BLEU score measures n-gram precision with a brevity penalty to ensure fluency and accuracy, which might affect the final results in case of the same semantic meaning between the predicted answer and the reference answer. Consequently, we execute another experiment by using GPT scores (Zheng et al., 2023),

a metric that uses a very large LLM (e.g., GPT-4o) serves as an evaluator to judge whether the predicted answer shares the same semantics as the reference answer, to evaluate the output of our proposed approach. The results of this evaluation are reported in Table 2. Accordingly, although the BLEU score is lower (10.4 vs. 15.7), GPT-score reflects better semantic accuracy.

4.3 Detailed Analysis

4.3.1 Impact of LLM Backbone

To validate the impact of different backbones on our approach, we conducted experiments with instruction versions of two additional modern open-source LLMs with a parameter size of 8 billion parameters, such as Qwen2 (Yang et al., 2024) and LLama-3.1-8B[2], as detailed in Table 3. Accordingly, despite using different backbone models, the proposed framework consistently achieves strong performance across both MultiWOZ 2.1 and 2.2 datasets. This demonstrates the robustness of our approach, maintaining high success, inform, and combined scores regardless of the underlying backbone.

Backbone	BLEU	Success	Inform	Comb.	JGA
MultiWOZ 2.1					
Qwen2	15.53	58.5	76.3	82.93	43.39
LLama3	10.41	75.5	86.0	91.18	47.06
LLama3.1	8.60	76.9	85.2	89.65	44.4
MultiWOZ 2.2					
Qwen2	**15.88**	58.6	75.7	83.03	45.66
LLama3	10.41	77.1	87.2	92.56	47.68
LLama3.1	8.65	77.5	84.7	89.75	45.8

Table 3: Comparison of Qwen2, LLama3 and LLama3.1 backbone performance when training with 10% on MultiWOZ 2.1 and MultiWOZ 2.2 datasets.

4.3.2 Impact of Lora Configuration

We conduct experiments on various LoRA configuration settings, as detailed in Table 4. Accordingly, the proposed method does not benefit from a high number of fine-tuned parameters. We hypothesize that with lower LoRA rank and alpha settings, the fine-tuned parameters are insufficient for effectively compressing useful information from the training data. In contrast, high LoRA rank and alpha configurations may lead to a loss of generalization in our model. Accordingly, the most suitable configuration is a LoRA rank of 32 and a LoRA alpha of 16, balancing the trade-off between information compression and generalization.

Lora-rank	Lora-alpha	Combined	JGA	Fn_Se
16	16	86.37	46.85	94.02
32	16	**92.56**	47.68	**94.75**
64	32	90.26	**47.88**	94.42
64	128	91.64	45.66	94.17

Table 4: Performance with varying LoRA-rank and LoRA-alpha settings, trained on 10% of the MultiWOZ 2.2 dataset. Fn_{Se} denotes the metric accuracy for the domain selection task.

4.3.3 Exploiting Number of Training Samples

We evaluated the performance of Spec-TOD using 10%, 50%, and 100% of the training data, respectively, as shown in Table 5. Notably, when fine-tuning with the full dataset (100% samples), the performance remains consistent with 50% of the training data. This finding demonstrates that our approach does not require extensive amounts of data to achieve performance comparable to previous full-training approaches. Accordingly, with only 10% of the training data, our model matches the performance of prior approaches that utilized the entire dataset. These results highlight the efficiency and robustness of our model in low-data scenarios.

Samples	BLEU	Success	Inform	Combined
10%	**10.41**	77.1	87.2	92.56
50%	10.09	**83.5**	**90.5**	**97.09**
100%	10.22	83.4	90.3	97.07

Table 5: Performance comparison with various numbers of training samples on MultiWOZ 2.2 dataset.

4.3.4 DST Task Performance

To further exploit the abilities of the proposed Spec-TOD, we evaluate the performance of the Function Calling (for DST) task, which is illustrated in Figure 5. Accordingly, the reported results indicate that with training on 10% of the data, Spec-TOD with Llama-3-8B outperforms GPT-3.5 and GPT-4 under zero-shot prompting.

5 Conclusion

TOD systems have long posed challenges in NLP due to limited flexibility and scalability to unseen domains. Recent emerging LLM technologies have

[2]https://ai.meta.com/blog/meta-llama-3-1/

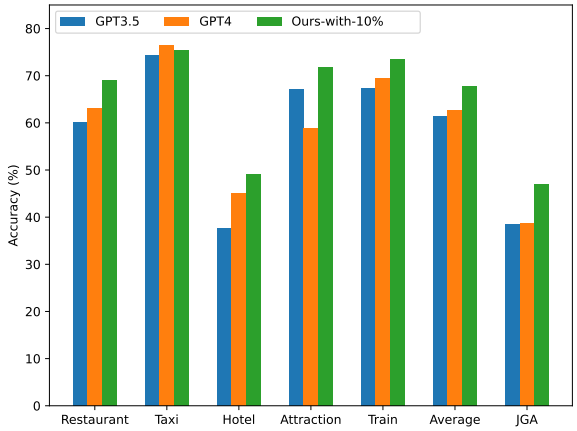

Figure 5: Performance comparison of our proposed method, training on 10 % data with GPT-3.5 and GPT-4 on DST task across different domains.

revolutionized NLP applications by enabling zero-shot and few-shot generalization capabilities. However, utilizing LLMs for tasks requiring knowledge grounding, such as TOD, continues to be a critical challenge that warrants further study, especially for end-to-end TOD systems. Therefore, this study presents a novel few-shot learning method for end-to-end TOD systems via instruction tuning with open-weight LLMs. The evaluation of various versions of the MultiWoz datasets indicates promising results of our proposed method with 10% of the training samples.

References

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, Toronto, Canada, July 9-14, 2023*, pages 7355–7369. Association for Computational Linguistics.

Jianzhu Bao, Rui Wang, Yasheng Wang, Aixin Sun, Yitong Li, Fei Mi, and Ruifeng Xu. 2023. A synthetic data generation framework for grounded dialogues. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10866–10882. Association for Computational Linguistics.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5016–5026. Association for Computational Linguistics.

Nguyen Quang Chieu, Quang-Minh Tran, and Khac-Hoai Nam Bui. 2024. Syntod: Augmented response synthesis for robust end-to-end task-oriented dialogue system. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 15493–15499. ELRA and ICCL.

Willy Chung, Samuel Cahyawijaya, Bryan Wilie, Holy Lovenia, and Pascale Fung. 2023. InstructTODS: Large language models for end-to-end task-oriented dialogue systems. In *Proceedings of the Second Workshop on Natural Language Interfaces*, pages 1–21, Bali, Indonesia. Association for Computational Linguistics.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Kumar Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 422–428. European Language Resources Association.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. 2023. Towards llm-driven dialogue state tracking. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 739–755. Association for Computational Linguistics.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2024. Gptscore: Evaluate as you desire. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 6556–6576. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur, and colleagues. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, Jian Sun, and Yongbin Li. 2022. GALAXY: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 10749–10757. AAAI Press.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yuncheng Hua, Xiangyu Xi, Zheng Jiang, Guanwei Zhang, Chaobo Sun, Guanglu Wan, and Wei Ye. 2023. Dialog-to-actions: Building task-oriented dialogue system via action-level generation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pages 3255–3259. ACM.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Trans. Inf. Syst.*, 38(3):21:1–21:32.

Wiradee Imrattanatrai and Ken Fukuda. 2023. End-to-end task-oriented dialogue systems based on schema. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10148–10161. Association for Computational Linguistics.

Harrison Lee, Raghav Gupta, Abhinav Rastogi, Yuan Cao, Bin Zhang, and Yonghui Wu. 2022. SGD-X: A benchmark for robust generalization in schema-guided dialogue systems. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 10938–10946. AAAI Press.

Yohan Lee. 2021. Improving end-to-end task-oriented dialog system with A simple auxiliary task. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 1296–1303. Association for Computational Linguistics.

Zekun Li, Zhiyu Chen, Mike Ross, Patrick Huber, Seungwhan Moon, Zhaojiang Lin, Xin Dong, Adithya Sagar, Xifeng Yan, and Paul A. Crook. 2024. Large language models as zero-shot dialogue state tracker through function calling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 8688–8704. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3391–3405. Association for Computational Linguistics.

Fei Mi, Yasheng Wang, and Yitong Li. 2022. CINS: comprehensive instruction for few-shot learning in task-oriented dialog systems. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11076–11084. AAAI Press.

Mehrad Moradshahi, Sina J. Semnani, and Monica Lam. 2023. Zero and few-shot localization of task-oriented dialogue agents with a distilled representation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 886–901. Association for Computational Linguistics.

Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, and Erik Cambria. 2023. Recent advances in deep learning based dialogue systems: a systematic survey. *Artif. Intell. Rev.*, 56(4):3055–3155.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Libo Qin, Wenbo Pan, Qiguang Chen, Lizi Liao, Zhou Yu, Yue Zhang, Wanxiang Che, and Min Li. 2023. End-to-end task-oriented dialogue: A survey of tasks, methods, and future directions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 5925–5941. Association for Computational Linguistics.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4661–4676. Association for Computational Linguistics.

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, Toronto, Canada, July 9-14, 2023*, pages 11139–11160. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Chunlei Xin, Yaojie Lu, Hongyu Lin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Zhongyi Liu, Xianpei Han, and Le Sun. 2024. Beyond full fine-tuning:

Harnessing the power of lora for multi-task instruction tuning. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 2307–2317. ELRA and ICCL.

Heng-Da Xu, Xian-Ling Mao, Puhai Yang, Fanshu Sun, and Heyan 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), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 2748–2763. Association for Computational Linguistics.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *CoRR*, abs/2407.10671.

Shiquan Yang, Xinting Huang, Jey Han Lau, and Sarah M. Erfani. 2022. Robust task-oriented dialogue generation with contrastive pre-training and adversarial filtering. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 1220–1234. Association for Computational Linguistics.

Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2021. UBAR: towards fully end-to-end task-oriented dialog system with GPT-2. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14230–14238. AAAI Press.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 351–360, Edinburgh, UK. Association for Computational Linguistics.

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. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

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, Singapore, December 6-10, 2023*, pages 13348–13369. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

A Appendix

A.1 GPT-score Criteria

Following the work in (Fu et al., 2024), we define the GPT-Score with six criteria for the measurement as follows:

- **Understand**: Are the responses understandable?
- **Relevant**: Are the responses relevant to the conversation?
- **Correct**: Are the responses correct to conversations?
- **Appropriate**: Are the responses semantically appropriate?
- **Fluently**: Are the responses fluent?
- **Direct**: Are the responses directly answering the request from the input query?

A.2 Specified Task Prompt Templates

For better reproducibility, we present all prompt templates in the appendix. Below is a quick reference list outlining the prompt templates and their usages:

- Figure 6: Prompt the task instruction for Domain Selection Task.
- Figure 7: Prompt the task instruction for Domain Dialogue State Tracking Task.
- Figure 8: Prompt the task instruction for Response Generation Task.

Task Instruction for Domain Selection

Task Instruction: You are a task-oriented assistant. Your role is to determine which domain the user is seeking information about or attempting to make a booking in during each turn of the conversation. Select the most relevant domain from the following domain description.

<function_list>

```
[
    {
     "name": hotel
      "description": hotel reservations and vacation stays
      "arguments": {...}
    }
    {
      "name": restaurant
      "description": find places to dine and whet your appetite
      "arguments": {...}
    }
   ...
]
```

</function_list>

<Dialogue Context>

-- Turn 1 --

Role User: "I need train reservations from norwich to cambridge"

Role Domain:<domain>hotel<\domain>

Role Assistant:"I have 133 trains matching your request. Is there a specific day and time you would like to travel?"

<\Dialogue Context>

-- Turn 2 --

Role User: ....

Figure 6: Prompt template for Domain Selection task

Task Instruction for Dialogue State Tracking

Task Instruction: You are a task-oriented assistant. Your primary objective is assisting users to finish their tasks, using the given function(s) if necessary. Don't make assumption about what values to plug into functions. Ask for clarification if a user request is ambiguous. Use only the argument values explicitly provided or confirmed by the user instead of the assistant. Don't add or guess argument values. Ensure the accuracy of arguments when calling functions to effectively obtain information of entities requested by the user.

<Function_Specification>

```
{
   "name": "hotel",
   "description": "hotel reservations and vacation stays.
   "arguments": [
      {
         "name": "pricerange",
         "type": "string",
         "description": "price budget of the hotel",
         "possible_values": [ "expensive", "cheap", "moderate" ]
      },
      {
         "name": "accommodation_type",
         "type": "string",
         "description": "what is the type of the hotel",
         "possible_values": [ "guesthouse", "guest house", "hotel" ]
      },
       ....
   ]
}
```

</Function_Specification>

To call a function with a JSON object of the following format: {"function": "function_name", "arguments": {"argument1": "argument_value", "argument2": "argument_value"}}

<Dialogue Context>

-- Turn 1 --

Role User: "I need to find mid-range price hotel on the south side?"

Role Function:<function_call>{"function": "hotel", arguments: {"pricerange": "moderate"}}</function_call>

Role Assistant:"There are 2 mid-range hotels that fit you criteria. Seton Hotel and Ac Hotel By marriott Beverly Hills. Which would you prefer?"

<\Dialogue Context>

-- Turn 2 --

Role User: .....

Figure 7: Prompt template for Dialogue State Tracking task

Task Instruction for Response Generation

Task Instruction: As a multi-domain task-oriented assistant, you'll interact with a database to provide meaningful responses to users. Your input will consists of two parts:
i) Function call: A description of the function used to query the database (e.g,. <function_call>{"function": "hotel", "arguments": {"area": "east", "stars": "4", "accommodation_type": "hotel"}}</function_call>).
ii) Query result: number of entities matching after querying the database.
Your task is to predict the appropriate action and then generate a helpful response. The possible actions you can take include:

1) name: inform
   description: provide information about an entity (if multiple matched results exist, choose one) in the form of [value_xxx] if requested by the user (required)
2) name: request
   description: inform the number of available offers ([value_choice]) and ask the user for more preference on the requested entity to narrow down the search results (optional)
3) name: nooffer
   description: inform the user that no suitable offer could be found
   ...

<Function_Specification>

```
{
   "name": "hotel",
   "description": "hotel reservations and vacation stays",
   "arguments": [
      {
         "name": "pricerange",
         "type": "string",
         "description": "price budget of the hotel",
         "possible_values": [ "expensive", "cheap", "moderate" ]
      },
      {
         "name": "accommodation_type",
         "type": "string",
         "description": "what is the type of the hotel",
         "possible_values": [ "guesthouse", "guest house", "hotel" ]
      },
      {
         "name": "area",
         "type": "string",
         "description": "area or place of the hotel",
         "possible_values": [ "centre", "east", "north", "south", "west" ]
      }
      ....
   ]
}
```

</Function_Specification>

<Dialogue Context>

-- Turn 1--

Role User: "I need to find mid-range price hotel on the south side?"

Role Function:<function_call>{"function": "hotel", arguments: {"pricerange": "moderate"}}</function_call>

Role Observation: There are 2 entities match

Role Assistant:

Action: [hotel]: {[inform]: [choice,name,type], [select]}
Resonse: There are [value_choice] [value_type] that fit you criteria. [value_name] and [value_name]. Which would you prefer?"

<\Dialogue Context>

-- Turn 2 --
Role User: .....

Figure 8: Prompt template for Response Generation task