Manual-Guided Dialogue for Flexible Conversational Agents

Ryuichi Takanobu[1], Hao Zhou[2], Yankai Lin[2], Peng Li[2], Jie Zhou[2], Minlie Huang[1]

[1]The CoAI group, DCST, Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems,

[1]Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

[2]Pattern Recognition Center, WeChat AI, Tencent Inc., China

`gxly19@mails.tsinghua.edu.cn, tuxzhou@tencent.com, aihuang@tsinghua.edu.cn`

Abstract

How to build and use dialogue data efficiently, and how to deploy models in different domains at scale can be two critical issues in building a task-oriented dialogue system. In this paper, we propose a novel *manual-guided dialogue* scheme to alleviate these problems, where the agent learns the tasks from both dialogue and manuals. The manual is an unstructured textual document that guides the agent in interacting with users and the database during the conversation. Our proposed scheme reduces the dependence of dialogue models on fine-grained domain ontology, and makes them more flexible to adapt to various domains. We then contribute a fully-annotated multi-domain dataset *MagDial* to support our scheme. It introduces three dialogue modeling subtasks: *instruction matching*, *argument filling*, and *response generation*. Modeling these subtasks is consistent with the human agent's behavior patterns. Experiments demonstrate that the manual-guided dialogue scheme improves data efficiency and domain scalability in building dialogue systems. **The dataset and benchmark will be publicly available for promoting future research**.

1 Introduction

Teaching machines to help humans accomplish their goals through conversation has long attracted many studies (Gao et al., 2019a; Zhang et al., 2020). Recent works (Hosseini-Asl et al., 2020; Peng et al., 2021) heavily rely on fine-grained domain ontology (Young et al., 2013) in building conversational agents to manage different scenarios, such as in-car navigation (Eric et al., 2017), customer service (Budzianowski et al., 2018) and home automation (Sciuto et al., 2018).

However, the construction of elaborate structured ontology, such as dialogue acts and slot-value pairs (Budzianowski et al., 2018; Rastogi et al., 2020), is extremely difficult and inefficient, as it requires both professional knowledge in dialogue systems and background knowledge across domains
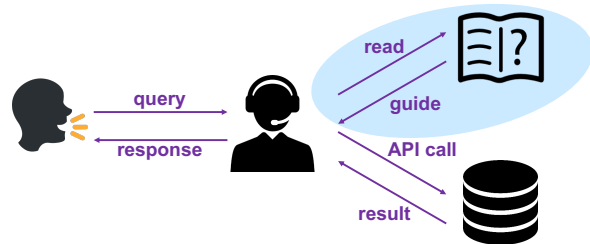

Figure 1: The overview scheme of a manual-guided dialogue system. The agent can access a textual manual (blue part) which provides the agent with guidance on completing user goals and making API calls during the conversation.

and services, which are hard to satisfy in practice. Furthermore, dialogue systems based on structured ontology knowledge are difficult to adapt to new domains and tasks, as the knowledge structure, *e.g.* the definition of dialogue acts, may change significantly across domains, which is challenging to model for existing ontology-based dialogue systems.

In this paper, we propose the manual-guided dialogue scheme, where the agent can conduct task-oriented dialogues based on unstructured manuals instead of structured ontology. A manual is a collection of textual *instructions* that contains the task guidelines and database documentation an agent needs in the task-oriented dialogue scenario. In addition to making API calls to the database, the agent can read the manual to find the guide on how to act and respond whenever the agent converses with users (see Fig. 1). The manual-guided dialogue scheme allows developers to build unstructured manuals based on task-specific knowledge (see Sec. 4.2), which takes little effort compared to designing designing fine-grained structured ontology. Once developers want to incorporate a new task, they only have to extend or modify the manuals, and then finetune the agent with few data to adapt to the new task.

To validate our hypothesis, we collect a six-

domain task-oriented dialogue dataset *MagDial* based on the proposed scheme, where 14 manuals are constructed with diverse language expressions to help and evaluate agents in understanding the unstructured task-specific knowledge. Three subtasks are introduced to facilitate training a manual-guided dialogue system following the above human behavior patterns: (1) *instruction matching* subtask that finds the correct instructions, *i.e.*, what the condition is and where the solution is, by scanning the manual to interpret user utterances, (2) *argument filling* subtask that extracts the value spans from dialogue history according to the selected instructions to make accurate API calls, (3) *response generation* subtask that generates appropriate responses conditioned on context and guided by the predicted outputs. Note that our data annotation process does not require well-defined ontology. A pipeline benchmarking three subtasks is also provided to demonstrate the usability of the manual-guided dialogue scheme. Results show that our pipeline improves data efficiency and domain scalability: Only hundreds of dialogue data are sufficient to yield effective dialogue models. Moreover, the trained models using our scheme perform well with unseen manuals.

Our contributions can be summarized as follows:

- We propose the scheme of manual-guided dialogue where the agent is guided by a manual consisting of task guidelines and database documentation during a task-oriented dialogue.

- We collect a dialogue dataset MagDial following the proposed scheme and introduce three corresponding dialogue modeling subtasks: instruction matching, argument filling, and response generation.

- We develop a flexible pipeline supported by the collected dataset. Extensive experiments demonstrate that the manual-guided dialogue scheme improves data efficiency and domain scalability.

2   Related Work

Most current techniques on task-oriented dialogue systems are data-intensive and require huge amounts of dialogue sessions to learn agent behaviors. Many large-scale corpora (Budzianowski et al., 2018; Byrne et al., 2019; Peskov et al., 2019; Rastogi et al., 2020) have been proposed to support

different approaches. Training a pipeline system built on these datasets requires plenty of specialized labels for each module, *e.g.*, language understanding (Wang et al., 2018; Goo et al., 2018) and dialogue state tracking (Wu et al., 2019; Heck et al., 2020). However, collecting high-quality labeled data is expensive and time-consuming. Though pretrained end-to-end models (Hosseini-Asl et al., 2020; Peng et al., 2021) achieve state-of-the-art (SOTA) performance, they predict dialogue states explicitly to inject semantics and constraints, which requires a large number of domain-specific labels as well.

In addition to the data-hungry models, careful design of domain ontology is essential to dialogue modeling. In MultiWOZ (Budzianowski et al., 2018), state trackers have to use heuristics by looking into the dataset and ontology to determine which slots should be categorical or span-based (Gao et al., 2019b; Zhou and Small, 2019). Such high annotation complexity also causes substantial noise (Eric et al., 2020; Zang et al., 2020; Han et al., 2020). Defining custom ontology in each work is another obstacle to rapid deployment. ABCD (Chen et al., 2021) designs more fine-grained intents and actions within a single domain. Others even introduce totally distinct ontology format, *e.g.*, semantic frames (El Asri et al., 2017) and meaningful representation (Novikova et al., 2017). Due to the wide variety of structured ontology, methods on different datasets vary significantly in model structures and optimization.

Our work is inspired by recent studies on conversational reading comprehension (Choi et al., 2018; Saeidi et al., 2018) which investigates understanding a text passage to answer a series of interleaved user queries that appear in a conversation. To the best of our knowledge, the most relevant works to ours are as follows: SGD (Rastogi et al., 2020) introduces natural language descriptions of intents and slots as input to support few-shot generalization in task-oriented dialogue modeling. Dialogue scenes in doc2dial (Feng et al., 2020) are grounded on various documents and the agent identifies relevant grounding content for information-seeking tasks. Some seed dialogue flows are first collected and then using user simulators for data augmentation to build a goal-oriented dialog agent efficiently via Alexa conversations (Acharya et al., 2021).

However, our work includes, but is not limited to, the following three critical features beyond ex-
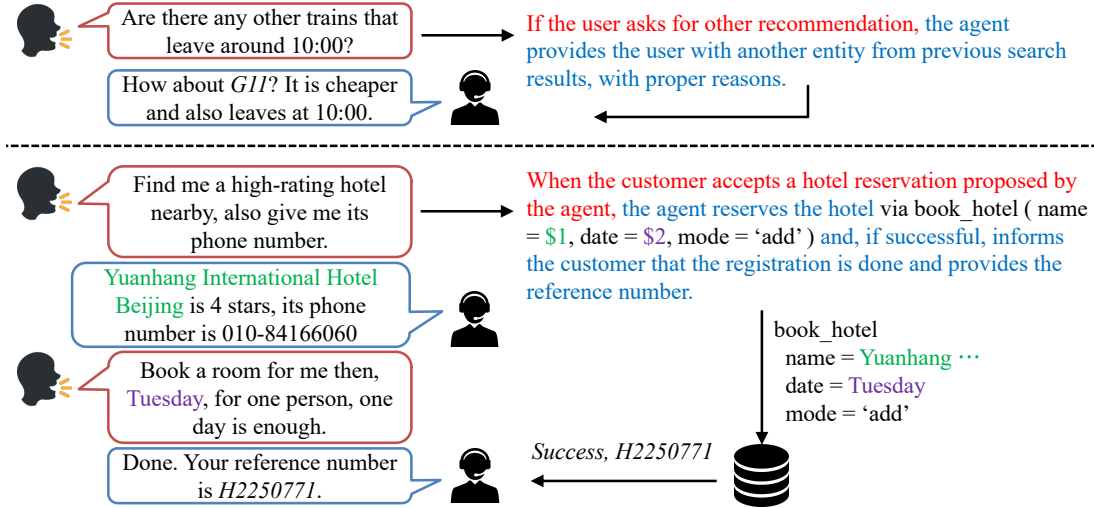
Figure 2: Several sample segments of conversation (left) in MagDial with selected instructions (right) in a manual. The paragraph marked in red/blue background color indicates the condition/solution part of an instruction. Each instruction may trigger an API call, and every API has its own input and output. As for argument filling, the agent detects the argument (*e.g.*, Tuesday) from the dialogue context and recognizes its attribute type by reading the attribute mentions (*e.g.*, date) in the selected instruction. The text spans to be extracted as an argument are marked in the same color with the corresponding attribute of APIs.

isting datasets: (1) The manual guides the agent in replying to users via conversation, and answers or responses cannot be derived from the manual itself. (2) The manual also guides making API calls to the database, rather than just explaining APIs by descriptions or comments. (3) Task-specific knowledge is represented in unstructured manuals. There are no annotations requiring domain expertise to define label categories or structures. (4) It is unnecessary to build a rule-based dialogue simulator that lacks scalability, nor train a model-based one with considerable extra labeled dialogue.

3 Manual-Guided Dialogue

In this section, we define some key concepts and formulate the task of manual-guided dialogue.

Definition 1 (Instruction). An instruction is a sentence describing the contextual *condition* of the current dialogue, the *API call* description (optionally) an agent may need to obtain attributes from the database, and the corresponding possible *solution* an agent may take to accomplish user goals.

Definition 2 (Argument). An argument of an API is the value that needs to be provided (possibly optional) for API calls to obtain the API results.

In a manual-guided dialogue, the user and agent converse with each other at each turn t, obtaining the user utterance U_t and agent response R_t. During the conversation, the agent can make API calls to retrieve relevant information from a database

DB including all candidate entities and corresponding attributes. Any API call holds the format of:

$$func\,(\text{attr}_1 = \text{arg}_1, \text{attr}_2 = \text{arg}_2, \cdots). \quad (1)$$

Meanwhile, the agent can read a manual as guidance. A manual $M = \{I_1, I_2, \dots\}$ is a textual document comprised of the group of all instructions I that instructs the agent how to interact with users and the database. Then the agent's goal is to learn a function f that selects the relevant grounding instructions I_t, extracts the required arguments A_t to be passed in API calls, and then generates an appropriate response R_t given the dialogue history $D_t = \{U_1, R_1, \dots, U_{t-1}, R_{t-1}, U_t\}$, the database DB and the manual M as input:

$$f : (D_t, DB, M) \to (I_t, A_t, R_t). \quad (2)$$

Let $I_t = \{I_{t,1}, I_{t,2}, \dots I_{t,l}\}$ where l denotes the number of instructions as multiple instructions can be selected at one dialogue turn. Similarly, $A_t = \{A_{t,1}, A_{t,2}, \dots A_{t,n}\}$ where n denotes the number of arguments. Each argument is a text span extracted from D_t, since any categorical attribute where legal arguments are enumerable can be transformed into a span-based counterpart[1]. Some instruction and argument examples are shown on the right side of Fig. 2.

---

[1] An extracted argument of a categorical attribute that is outside the list of legal arguments can be mapped to the most relevant argument through similarity calculation or other methods, which is beyond the scope of this paper.

4 MagDial Dataset

This section introduces the MagDial dataset where the agent follows the manual-guided dialogue scheme during the conversation with users. Unlike traditional task-oriented dialogue corpus, the manuals are incorporated in this dataset to guide the agent to interact with users and the database, aiming to reduce the dependency on fine-grained structured ontology. The dataset, including manuals and dialogues, is collected in Chinese. Fig. 2 presents two dialogue segments from the dataset. From sample segments, we can observe that the selected instructions summarize the current dialogue situation (the condition part) and provide the agent with some guidance on the next action (the solution part). The agent may choose different I_t from M at a turn t because multiple responses can be appropriate for the same D_t, leading to different valid dialogue policies towards task completion.

Most parts of the manual include API calls (*e.g.*, the second segment). The agent should follow the manual to extract arguments from the dialogue context to interact with the database properly. At each turn t, the agent selects relevant I_t by scanning each instruction on M, and extracts A_t to be passed in API calls by detecting text spans from D_t. When predicting text spans, the agent reads the I_t to focus on the target arguments, and does not need to recognize its attribute type beforehand. This indicates that all the annotations used in our scheme are domain-agnostic[2], thus improving the scalability across domains and tasks.

In the following text, we describe how we collect MagDial in three steps: (1) set up the background knowledge; (2) construct the manuals; (3) collect the dialogue and annotations.

4.1 Data Setup

We first set up the background knowledge needed in the task-oriented dialogue scenario, including the database, APIs, and user goals. Only the agent can interact with the database during the conversation, while the user has a user goal in mind and talks to the agent to accomplish the goal.

Database For constructing the database DB, we crawled six-domain entities and corresponding attributes in Beijing from the Web. We choose the domains following MultiWOZ (Budzianowski et al., 2018) to give an intuitive comparison with exist-

ing datasets in experiments. The statistics and full list of attributes are provided in Table 1 and 7 in Appendix due to the length limit.

API There are four main API calls in a general database: *find*, *add*, *edit* and *delete* operations. In MagDial, we simplify the available user intents in dialogue to two categories: *search* and *booking*. A detected *search* intent triggers a *find* call that searches for relevant information, while a recognized *booking* intent may trigger a(n) *add* / *edit* / *delete* call that makes / changes / cancels a reservation based on the dialogue context. Note that API calls in MagDial do not edit the entities in the database. *Carryover operation*[3] is always conducted in all API calls to carryover the arguments from previous turns, unless updated at the current turn.

User Goal A user goal $G = (C, R)$ is composed of constraints C that indicate the specific conditions of the goal (*e.g.*, find a *Japanese* restaurant) and requests R that represents what information or service the user seeks (*e.g.*, require for the address of a restaurant). To construct user goals, we first sample the domains, with a max number of 4 since too many domains in a goal make the dialogue unreal. Then we randomly sample the attributes as C or R accordingly. Note that one attribute should not be included in C and R simultaneously. For constraint attributes, we next sample the arguments from the database. Some rules are designed during the sampling process to eliminate unreasonable combinations.

4.2 Manual Construction

Although numerous manuals exist in real business scenarios, *e.g.*, telephone customer service, we can hardly obtain these manuals and corresponding dialogues due to commercial restriction and data privacy. In order to provide open data resources for the research community and demonstrate the relatively simple process of manual construction, we decide to build the manuals M from scratch.

An overview of the manual collection process is illustrated in Fig. 3. We first extract task-specific background knowledge, such as user goals

---

[2]More technical details can be found in Sec. 5.1 and 5.2.

[3]For example, if a user wants to find a Japanese restaurant with an API call $restaurant\_search$(food = Japanese) at first, and then he/she says that the restaurant should also be in the center of the city, the agent only has to make an API call $restaurant\_search$(area = center) and do not need to take "Japanese" as the argument again.
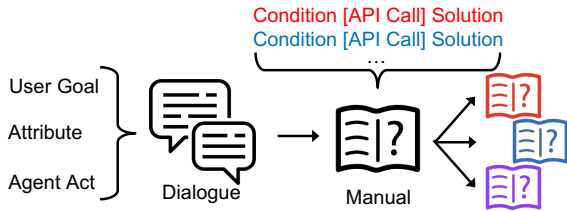
Figure 3: The process of manual construction.

Metric	Train	Dev	Test
dialogues	900	100	100
Turns	5,350	562	652
Tokens	294,956	30,633	37,571
Vocab	5,593	2,187	2,298
Manuals	10	10	4
Avg. dom. / dialogue	2.42	2.36	2.73
Avg. turn / dialogue	5.94	5.62	6.52
Avg. token / turn	55.14	54.51	57.62
Avg. instr. / turn	1.40	1.38	1.42
Avg. arg. / turn	1.17	1.22	1.19

Table 2: Statistics on dialogues. One turn here contains a user utterance and an agent response. The tokens are counted in Chinese.

and attributes, from the conversation in Multi-WOZ (Budzianowski et al., 2018). Then we manually design a small number of seed instructions, which contains a condition utterance describing user goals, an API call description (optionally) stating the method to obtain relevant attributes, and a solution utterance indicating the agent acts to complete user goals. Afterwards, we treat a manual M as the composition of all the instructions $\{I\}$. The number of instructions in each domain is presented in Table 1.

To approximate the real business scenario that the style and expressions of manuals may be diverse across services, we further ask workers to paraphrase the seed instructions. In order to generate diverse paraphrases in terms of lexical and syntactic representation, each instruction is rewritten by one worker to avoid similar paraphrases, and self-BLEU (Zhu et al., 2018) score should be smaller than 0.8. Finally, 14 paraphrases are created for each instruction, thus obtaining 14 manuals.

Domain	Entity	Attribute	Instruction
Attraction	465	8	90
Hospital	91	6	43
Hotel	1,133	13	80
Restaurant	951	12	114
Train	1,022	12	133
Taxi[4]	-	6	56

Table 1: Data statistics of the database and manuals.

4.3 Dialogue Collection

Following Byrne et al. (2019), we adopt a self-play approach to collect the dialogue and annotations. It has been demonstrated to render high-quality dialogue data efficiently and reasonably without too much unreality. The worker is asked to imagine the scenario where a user converses with an intelligent agent to complete the goal, and plays the user's

role and the agent's alternately until the user goal is completed to collect the dialogue on his/her own. Unlike Byrne et al. (2019) where workers write down the entire conversation using a single platform, the conversation environments for the "user" and "agent" are separate and different to ensure the workers switch the user/agent role during the conversation.

User Side At the beginning of the conversation, the worker is provided with a goal description and a table listing constraints and requests of the sampled goal. If a conversation ends without finishing the given goal, the goal will be put back to the task pool for other workers. To guarantee that the goal is properly expressed by the workers and successfully completed through conversation, the worker has to check the stated constraints at the current turn, or refill the value into the requests on the table at each turn. When all the constraints have been checked and all the requests have been filled, the task is completed.

Agent Side When the worker acts as an agent, he/she tries to make API calls for users and respond based on the API results. For each conversation, the agent side is provided with a fixed manual. In order to get argument labels at the same time as dialogue collection, any API call is recorded. We use fuzzy string matching to annotate the span index of arguments in the dialogue context by looking into the recorded API calls during data post-processing. Moreover, the worker ought to select all the instructions used in the current turn as well. While it is prohibitive to select instructions from dozens of candidates at each turn, we provide the worker with a search interface to narrow down the instruction candidates during the annotation collection process. For all turns where workers select no instruction,

---

[4]We do not make up "entities" in *Taxi* domain as anyone can book a taxi anytime and anywhere.

we check to confirm that no instruction can be selected indeed, otherwise we regard the dialogue as a failure and ask the worker to repair it as a penalty.

4.4 Data Analysis

Some statistics of MagDial are presented in Table 2. The dataset is built up from self-play conversations based on 1,100 distinct user goals. Each goal consists of 1 up to 4 domains, leading to 220/880 single-/multi-domain dialogues. The average number of turns is 5.97, while Multi-WOZ (Budzianowski et al., 2018) consisting of 1 to 5 domains for each dialogue has an average 6.84 number of turns. This indicates the moderate complexity of our dataset. The maximum number of selected instructions at a turn is 10, and the maximum number of triggered arguments in an instruction is 2.

To measure the model's generalization, 10 out of 14 manuals are used for training/development set, and the rest 4 are for the test set, as shown in Table 2. We collect 1,000 dialogues using 10 manuals, and the other 100 dialogues with the rest 4 manuals. The former 1,000 dialogues are randomly split into training and development sets. The 1,100 dialogues are constructed based on 1,100 distinct user goals, which leads to different semantics.

Fig. 7 in Appendix further illustrates some data distributions on the collected dialogues and manuals. The average number of sentence length is 22.27 and 32.41 for the user and agent side, respectively. After the manual paraphrasing process mentioned in Sec. 4.2, the average instruction length rises from 34.59 to 45.23, and its distribution becomes more normal. The number of instructions per turn shares a similar distribution with the number of arguments per turn since most instructions trigger one API call. Noticeably, neither an instruction is selected, nor an argument is triggered at a certain rate (around 20%) of all dialogue turns since some turns only involve simple and generic dialogue behaviors such as greeting and farewells.

5 Benchmarks

To evaluate the benefits of a manual-guided dialogue scheme on improving data efficiency and domain scalability, we propose three subtasks with each a benchmark on the MagDial dataset to facilitate experimental analysis: *instruction matching*, *argument filling* and *response generation*. Concerning benchmark models, we leverage the recent suc-

Model	Acc.	Prec.	Rec.	F1
BERT-rec	96.70	42.83	71.06	49.92
RoBERTa-rec	96.73	43.07	**75.33**	50.74
BERT	98.33	54.67	55.12	51.96
RoBERTa	**98.36**	**54.71**	58.30	**53.42**

Table 3: Results of instruction matching.

cess of pretrained language models (PLMs) (Devlin et al., 2019; Liu et al., 2019) on various NLP tasks and their potential towards general language understanding. An overview framework of the MagDial benchmarks is illustrated in Fig. 4. All results are evaluated on the test set in subsequent subsections.

5.1 Instruction Matching

First of all, the agent selects applicable instructions I_t from the manual M given the dialogue history D_t. Since multiple instructions can be selected at a turn, we regard this as a multi-label sentence pair classification task. Each time the model encodes D_t and a candidate instruction I_k, and predicts whether the condition part of I_k matches the dialogue context. We use BERT-base (Devlin et al., 2019) and RoBERTa-base (Liu et al., 2019) as the benchmark models. The model encodes the multi-turn D_t as a long sequence, as shown in Fig. 4a. In terms of automatic metrics, we apply sentence-level accuracy to measure if each instruction I_k in M is correctly predicted, and turn-level F1 score to measure the alignment between the selected instructions $\hat{I}_t$ and ground truth instructions I_t.

The results of instruction matching are presented in Table 3. We adopt two strategies during model optimization. One is to select the model with the highest recall on the development set during training iterations ($-rec$), the other uses F1 score as usual. It shows that high-recall targeted BERT and RoBERTa can achieve over 70% recall with the unseen manuals. We can also observe that a standard PLM can achieve above 50% F1 score and 98% accuracy.

5.2 Argument Filling

To enable the agent to interact with DB, we next need to extract relevant arguments A_t from D_t to be filled into the APIs triggered by the selected I_t. We regard this as a sequential tagging task with the `BIO` (Begin, Inside, Outside) format (Ramshaw and Marcus, 1999). Considering that there can be multiple attributes in one API call, we extend the *Begin* and *Inside* tags with an index indicating the
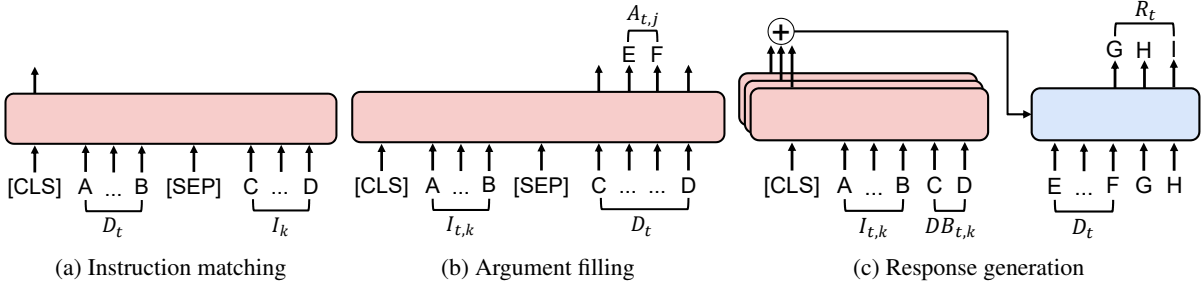
Figure 4: Benchmark modeling for each subtask. Red modules stand for encoders, and blue ones for decoders.

Model	Acc.	Prec.	Rec.	F1
BERT	98.58	91.57	90.69	91.13
RoBERTa	**99.14**	**92.34**	**92.57**	**92.46**
RoBERTa (w/o Manual)	91.23	56.96	64.59	60.54

Table 4: Results of argument filling.

Model	PPL↓	BLEU↑	AER↓
TT + BERT	**2.98**	44.18	29.24
TT + RoBERTa	**2.98**	**45.02**	**28.12**
TT (w/o Manual)	3.45	42.15	32.22

Table 5: Results of response generation.

attribute order in API to address the multi-argument extraction issue. Taking the second segment in Fig. 2 as an instance, the tags for *Yuanhang International Hotel Beijing* and *Tuesday* are `B-1, I-1, I-1, I-1` and `B-2` respectively, and `O` for other tokens. Here we apply the same benchmarks and similar encoding methods as the ones in the previous subtask, as shown in Fig. 4b. As for automatic metrics, given a certain $I_{t,k}$, token-level accuracy on tag classification and turn-level F1 score between extracted arguments $\hat{A}_t$ and ground truth arguments A_t are used.

Results on Table 4 demonstrate that both BERT and RoBERTa can achieve a relatively high f1 score (over 98%/90% on the token/turn level) on argument filling thanks to our domain-agnostic annotations. Though there are many domains and attributes, the model's output space can be restricted to an extremely low number by the nature of the proposed tagging scheme (|`BIO`| × |*index*|). The high performance also indicates that the trained models can correctly recognize different attributes on manuals and associate them to the proper arguments in the dialogue context. Noticeably, the F1, precision, and recall scores decrease significantly if the model cannot access manuals (the last line). It implies that manual-guided dialogue can teach the agent to interact with the database through reading manuals.

5.3 Response Generation

Finally, we aims to generate appropriate R_t given all the D_t, I_t, and API results DB_t. We model this subtask as a conditional language genera-

tion task. TransferTransfo (TT) (Wolf et al., 2019), a generative pretrained dialogue system, is used as the benchmark generator. Similarly, BERT and RoBERTa are adopted as manual encoders. Since there can be multiple instructions in I_t and therefore multiple results in DB_t, we apply a scaled dot-product attention mechanism Attention(Q, K, V) (Vaswani et al., 2017) to get a hidden representation over I_t and DB_t, as shown in Fig. 4c:

$$
\begin{aligned}
\mathbf{H}_t &= \text{Encoder}([I_t; DB_t]), \\
\mathbf{z}_{t,j} &= \text{Generator}([D_t; R_{t,<j}]), \\
\mathbf{z}_{t,j} &= \mathbf{z}_{t,j} + \text{Attention}(\mathbf{z}_{t,j}, \mathbf{H}_t, \mathbf{H}_t),
\end{aligned} \tag{3}
$$

where $\mathbf{z}_{t,j}$ is the hidden vector used to compute the j-th token's probability of the generated sample R_t. To evaluate the guidance of manuals on response generation, we also apply a single TT as a baseline for ablation study, which directly encodes the concatenated sequence of DB_t and D_t as conditions. For a fair comparison, the encoder's parameters are frozen during optimization for those methods encoding manuals. We adopt Perplexity (PPL) and BLEU score (geometric mean of 1∼4-gram) (Papineni et al., 2002) to measure the generation quality. Furthermore, Argument Error Rate (AER) is introduced to evaluate whether the agent replies to the user with meaningful information by including return results DB_t in the generated response $\hat{R}_t$.

Table 5 shows that the agent gives better responses after using manuals. A clear decrease on PPL and a distinct increase on BLEU are realized by encoding manuals as additional conditions. With the guidance of manuals, the agent achieves
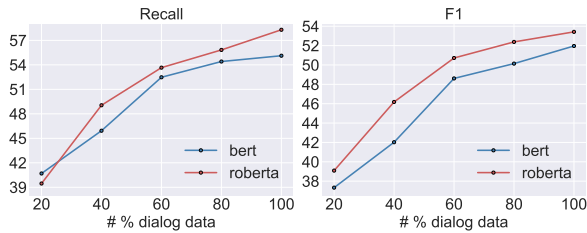
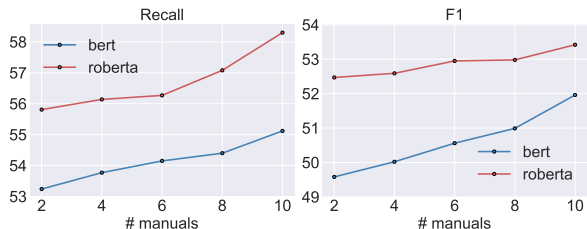Figure 5: Results w.r.t. the size of training data.

Figure 6: Results w.r.t. the number of manuals in training data.

lower AER, therefore the generated response offers more accurate information.

6 Experiments

In this section, we further demonstrate the ability of a conversational agent grounded on manual-guided dialogue in data efficiency and manual/domain scalability. We focus on *joint instruction matching and argument filling subtasks* in this section, since these serve for the understanding of background knowledge in task-oriented dialogue and thus is essential for downstream tasks.

6.1 Data Efficiency

Fig. 5 examines the agent's dependency on training data's size. We train the agent with 20%, 40%, 60%, 80% of the training set, respectively. We can observe that the agent performs better with the growth of training data size. Moreover, both recall and F1 scores increase rapidly before 60% data. We also found that models trained on about 500 dialogues (60% of training data) can already reach a 0.5 F1 score, which is close to the performance of models trained on full data. These show that manual-guided dialogue scheme has a strong few-shot ability that can develop the task-oriented dialogue system with less training data.

6.2 Manual Scalability

Fig. 6 studies the agent's dependency on the number of training manuals to help it adapt to a different manual *within domains*. We optimize the agent by using only part of (2, 4, 6, or 8) manuals in training and development sets. We find that using more

Domain	Att.	Hos.	Hot.	Res.	Tra.
Full	<u>48.49</u>	**54.44**	54.04	49.90	50.40
- Attraction	<u>40.63</u>	45.85	**48.80**	45.77	42.61
- Hospital	<u>43.18</u>	48.42	**49.37**	44.79	45.37
- Hotel	37.57	**39.36**	<u>37.23</u>	36.55	37.59
- Restaurant	<u>42.95</u>	46.00	**47.77**	45.34	45.99
- Train	43.12	**45.79**	45.18	45.41	<u>41.28</u>

Table 6: F1 score of BERT by excluding one domain from training data (row) and tested on the dialogues that involve a certain domain (column). For each row, the best score is **bolded**, and the worst score is <u>underlined</u>.

manuals boosts the agent's performance as well, but surprisingly, conversational agents can quickly obtain around 0.5 F1 scores with only a couple of manuals. This indicates that the proposed scheme has relatively good scalability within domains.

6.3 Domain Scalability

To further explore the agent's scalability *across domains*, we exclude one domain from training and development sets by removing the dialogues which involve that domain. As expected, Table 6 reveals a slight decline in all domains, whichever domain is excluded. Interestingly, the model's performance on the unseen domain varies from domain to domain (good for *Hospital* and *Restaurant*, poor for *Attraction* and *Train*). This implies manual-guided dialogue has moderate domain scalability across domains.

7 Conclusion

We propose a manual-guided dialogue scheme where the agent learns the tasks from both dialogues and manuals. The manual contains the task guideline to help the agent understand the context and converse with users, and also contains the database documentation to assist the agent in interacting with the database. Through reading the unstructured textual manuals, conversational agents can accomplish task goals more efficiently and are more flexible to handle multiple domains and make API calls without relying on the elaborate structured ontology. Based on the proposed scheme, we collect the MagDial dataset and introduce three subtasks to build a benchmark pipeline grounded on it. Extensive experiments demonstrate that the manual-guided dialogue scheme improves data efficiency and domain scalability.

References

Anish Acharya, Suranjit Adhikari, Sanchit Agarwal, Vincent Auvray, Nehal Belgamwar, Arijit Biswas, Shubhra Chandra, Tagyoung Chung, Maryam Fazel-Zarandi, Raefer Gabriel, Shuyang Gao, Rahul Goel, Dilek Hakkani-Tur, Jan Jezabek, Abhay Jha, Jiun-Yu Kao, Prakash Krishnan, Peter Ku, Anuj Goyal, Chien-Wei Lin, Qing Liu, Arindam Mandal, Angeliki Metallinou, Vishal Naik, Yi Pan, Shachi Paul, Vittorio Perera, Abhishek Sethi, Minmin Shen, Nikko Strom, and Eddie Wang. 2021. Alexa conversations: An extensible data-driven approach for building task-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 125–132.

Paweł 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*, pages 5016–5026.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4506–4517.

Derek Chen, Howard Chen, Yi Yang, Alexander Lin, and Zhou Yu. 2021. Action-based conversations dataset: A corpus for building more in-depth task-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3002–3017.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Layla El Asri, Hannes Schulz, Shikhar Kr Sarma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. 2017. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 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*, pages 422–428.

Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. Doc2dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8118–8128.

Jianfeng Gao, Michel Galley, and Lihong Li. 2019a. Neural approaches to conversational ai. *Foundations and Trends® in Information Retrieval*, 13(2-3):127–298.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019b. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 264–273.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Wei Peng, and Minlie Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation. *arXiv preprint arXiv:2010.05594*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21st Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44.

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*, volume 33, pages 20179–20191.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Building task bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Denis Peskov, Nancy Clarke, Jason Krone, Brigi Fodor, Yi Zhang, Adel Youssef, and Mona Diab. 2019. Multi-domain goal-oriented dialogues (multidogo): Strategies toward curating and annotating large scale dialogue data. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4518–4528.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2087–2097.

Alex Sciuto, Arnita Saini, Jodi Forlizzi, and Jason I Hong. 2018. "hey alexa, what's up?" a mixed-methods studies of in-home conversational agent usage. In *Proceedings of the 2018 Designing Interactive Systems Conference*, pages 857–868.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A bi-model based rnn semantic frame parsing model for intent detection and slot filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 309–314.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Xiaoxue Zang, Abhinav Rastogi, 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.

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.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *arXiv preprint arXiv:1911.06192*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

A Appendices

A.1 Data Statistics

Table 7 presents the full list of attributes used in the database and manuals. Fig. 7 illustrates some data distributions on dialogues and manuals of MagDial. The average number of sentence length is 22.27 and 32.41 for the user and agent side, respectively. After the manual paraphrasing process, the average instruction length rises from 34.59 to 45.23, and its distribution becomes more normal. The number of instructions per turn shares a similar distribution

Attribute	address, area, arrive, car, choice, class, day, department, departure, destination, facility, food, id, leave, name, people, phone, postcode, price, reference num., score, station, star, stay, time, type

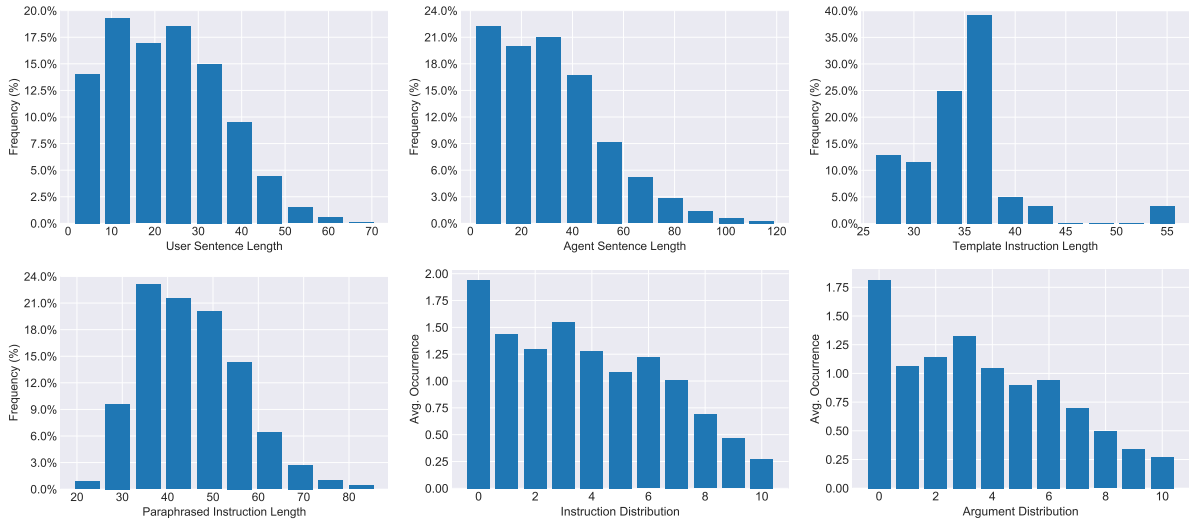Table 7: Full list of attributes extracted in the database, and used in manuals.

Figure 7: Distributions on dialogues and manuals of MagDial. User/Agent sentence length distribution (top left/top middle), instruction length distribution before/after paraphrasing (top right/bottom left), and distribution of number of instructions/arguments per turn (bottom middle/bottom right).

with the number of arguments per turn since most instructions trigger one API call. Noticeably, neither an instruction is selected, nor an argument is triggered at a certain rate (around 20%) of all dialogue turns since some turns only involve simple and generic dialogue behaviors such as greeting and farewells.

A.2 Examples

Fig. 8 presents four dialogue segments from the dataset. From sample segments, we can observe that the selected instructions summarize the current dialogue situation (the condition part) and provide the agent with some guidance on the next action (the solution part). The agent may choose different I_t from M at a turn t because multiple responses can be appropriate for the same D_t, leading to different valid dialogue policies towards task completion. For example, in the first and second segments, the agent can either ask users what they require, or recommend another entity given the same user query, thereby producing different responses. The essence of the manual-guided dialogue scheme is to help the agent **understand** what the task is and how to act by reading manuals, so that the agent can better handle diverse user behaviors and agent policies instead of simply imitating a function mapping from conversation to response. Therefore, the

agent guided by manuals can learn to interact with users **more efficiently**.

Most parts of the manual also include API calls (*e.g.*, the third/fourth segment). The agent should follow the manual to extract arguments from dialogue context to interact with the database properly. At each turn t, the agent selects relevant I_t by scanning each instruction on M, and extracts A_t to be passed in API calls by detecting text spans from D_t. When predicting text spans, the agent reads the I_t to focus on the target arguments, and does not need to recognize its attribute type beforehand. This indicates that all the annotations used in our scheme are domain-agnostic, thus improve the scalability across domains and tasks.

A.3 Annotation Interface

Fig. 9 provides the user side interface with a goal description (the left column) and a table listing constraints for quality control (the middle column). Fig. 10 presents the agent side interface with API calls and results (the left column) and manuals (the middle column).
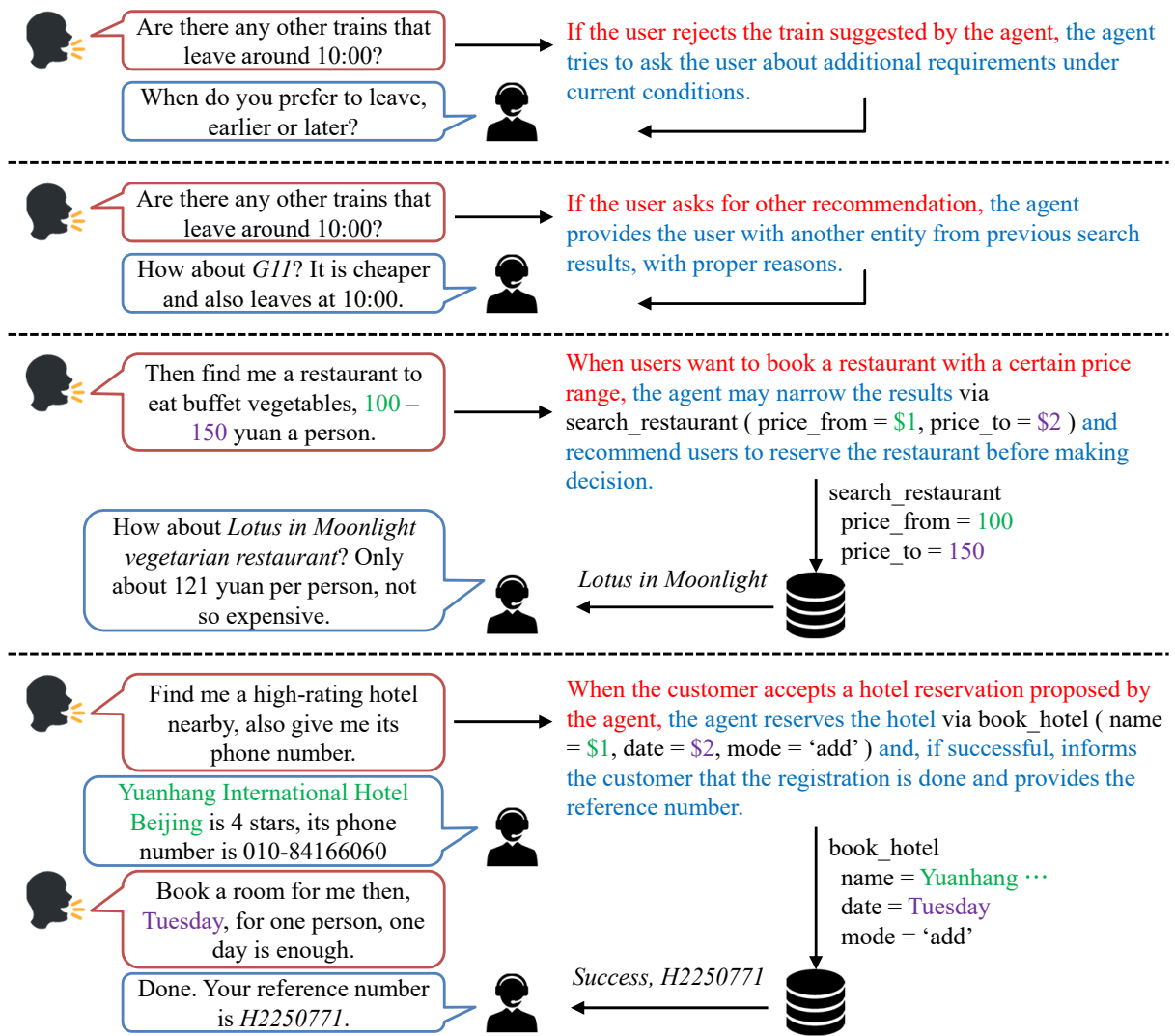
Are there any other trains that leave around 10:00?

When do you prefer to leave, earlier or later?

If the user rejects the train suggested by the agent, the agent tries to ask the user about additional requirements under current conditions.

Are there any other trains that leave around 10:00?

How about *G11*? It is cheaper and also leaves at 10:00.

If the user asks for other recommendation, the agent provides the user with another entity from previous search results, with proper reasons.

Then find me a restaurant to eat buffet vegetables, 100 – 150 yuan a person.

When users want to book a restaurant with a certain price range, the agent may narrow the results via search_restaurant (price_from = $1, price_to = $2) and recommend users to reserve the restaurant before making decision.

search_restaurant
price_from = 100
price_to = 150

Lotus in Moonlight

How about *Lotus in Moonlight vegetarian restaurant*? Only about 121 yuan per person, not so expensive.

Find me a high-rating hotel nearby, also give me its phone number.

Yuanhang International Hotel Beijing is 4 stars, its phone number is 010-84166060

When the customer accepts a hotel reservation proposed by the agent, the agent reserves the hotel via book_hotel (name = $1, date = $2, mode = 'add') and, if successful, informs the customer that the registration is done and provides the reference number.

book_hotel
name = Yuanhang ⋯
date = Tuesday
mode = 'add'

Success, H2250771

Book a room for me then, Tuesday, for one person, one day is enough.

Done. Your reference number is *H2250771*.

Figure 8: Several sample segments of conversation (left) in MagdDial with selected instructions (right) in a manual. The paragraph marked in red/blue background color indicates the condition/solution part of an instruction. Each instruction may trigger an API call, and every API has its own input and output. As for argument filling, the agent detects the argument (*e.g.*, Tuesday) from the dialogue context and recognizes its attribute type by reading the attribute mentions (*e.g.*, date) in the selected instruction. The text spans to be extracted as an argument are marked in the same color with the corresponding attribute of APIs.

任务描述

1. 你想乘坐一列前往北京的火车(id=1)。你需要从东营南出发。你希望火车18:14出发。北京的火车站为北京。你希望预订周四的车票，共5张。你想知道这个火车的车次、等级、预订订单号。
2. 你要找一家医院(id=2)看病。你需要这家医院配备的科室是加强医疗科。你想知道这个医院的名称、地址。

提交表单

	id	领域	槽	值
☐	1	火车	始发站	东营南
☐	1	火车	出发时间	18:14
☐	1	火车	车站	北京
☐	1	火车	车次	
☐	1	火车	等级	
☐	1	火车	预订订单号	
☐	1	火车	人数	5
☐	1	火车	星期	周四
☐	2	医院	科室	加强医疗科
☐	2	医院	名称	
☐	2	医院	地址	

user0

我想查一下周四东营南到北京的火车

完成对话   终止对话   发送 (Enter)

User Goal   **Quality Constraints**   **Dialogue**

Figure 9: Interface from the user side, where the user goal description/quality constraints/dialogue is presented in the left/middle/right column respectively.

景点	餐馆	酒店	医院	火车

名称		
门票	100	~	200
游玩时间		
评分	4.5	~	

查询

	名称
☑	北京欢乐谷
☐	北京海洋馆
☐	北京杜莎夫人蜡像馆
☐	北京野生动物园
☐	世界公园
☐	观复博物馆

槽	值
domain	[景点]
user	[询问] 地址
entity	1
book	无关
system	[告知] 地址

API calls and results

提交表单

领域	景点
用户意图	询问
用户槽位	地址
系统意图	
系统槽位	地址
结果个数	
预定情况	

查询

	手册
☐	当用户寻求某景点的地址时，立即查询并告知用户景点的地址。
☐	当用户想要获得某地景点推荐和信息时，立即查询并回复景点的地址，并作适当的推荐。
☑	当用户想要得到某个景点的信息时，立即查询并回复。

Manuals

test1

你好，我想找个景点玩，希望评分高一点，4.5分以上，门票可以稍微贵一点，不超过200元，你有什么推荐吗？

为你找到北京欢乐谷，评分是4.5，门票价格是195元

这个景点的地址是什么？

地址shi

嗯？

不好意思，消息发错了，地址是：北京市朝阳区金蝉西路东四环小武基北路(四方桥东南)

请您尽快提交表单　终止对话　发送(Enter)

Dialogue

Figure 10: Interface from the agent side, where the API calls and results/manuals/dialogue is presented in the left/middle/right column respectively.