A Tree-Based Approach for Building Efficient Task-Oriented Dialogue Systems

Tao Gan(✉), Chunang Li, Yuhui Xi, and Yanmin He

School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
gantao@uestc.edu.cn

Abstract. Task-oriented dialogue systems have attracted increasing attention. The traditional rule-based approaches suffer from limited generalization ability as well as the high cost of system deployment, whereas the data-driven deep learning approaches are data-hungry and the domain-specific data is insufficient for full training their models. In this paper, we present a hybrid method which combines the strengths of both rule-based and data-driven approaches. We first establish intent-slot trees from the standard multi-turn dialogue corpus in specific domain. During the dialogue, the power of deep language understanding model is exploited to enhance the generalization ability of the system and the multi-turn dialogue proceeds following the path of the intent-slot tree established. Experimental results show that the proposed approach achieves superior performance over deep learning ones which demonstrates its effectiveness in building task-oriented dialogue systems under a limited amount of training data.

Keywords: Task-oriented · Rule-based · Data-driven · Intent-slot tree

1 Introduction

In recent years, the task-oriented dialogue system has attracted more and more attention in both academic and industrial communities. A task-oriented dialogue system aims to help the user to accomplish certain tasks, such as restaurant reservation, flight booking or business consultation, etc. Unlike the non-task-oriented dialogue system which is in open-domain setting, the task-oriented one is more targeting at handling problems in specific domains.

The typical task-oriented dialogue systems often implement a complicated pipeline architecture, consisting of natural language understanding (NLU), dialogue management (DM) and natural language generation (NLG) [1]. The function of NLU is to transform the user's input into the form that the computer can understand using grammatical and semantic analysis. It usually has two main tasks: intent detection and semantic slot filling. The DM takes the output of NLU as well as dialogue context information to determine the corresponding actions to be taken by the system. A typical DM component includes two stages: dialogue state tracking and policy learning. In the last module NLG, the

X. Wang et al. (Eds.): AICON 2021, LNICST 396, pp. 526–536, 2021.
https://doi.org/10.1007/978-3-030-90196-7_45

system's responses are converted into appropriate statements in natural language that users can understand.

Different implementations of the above modules result in different dialogue systems. Approaches for developing dialogue systems are typically categorized into rule-based and data-driven. Traditional systems are based on rules [2]. In such systems, a set of domain-dependent rules are predefined and user utterances are matched against the rules during the dialogue. Typical approaches to dialogue management are based on finite state models, where user utterances trigger transitions between the dialogue states, and these states determine the system's response [3, 4]. The main advantage of rule-based systems is that they do not need training data and so that they have no cold-start problem. Yet such systems have a disadvantage of limited generalization ability. When the user utterances do not match the established rules well, the dialogue performance will noticeably decline. Moreover, the use of hand-crafted rules for the state and action space representations makes it expensive and time-consuming to deploy a real dialogue system [1]. In contrast, data-driven approaches rely on training the dialogue system using training data. For example, in dialogue state tracking, the generative model is built to learn the relevant joint probability density distribution from the training data and calculate the conditional probability distribution of all dialogue states [5, 6]. Furthermore, to model the dialogue state more accurately, the discriminative model which treats dialogue state tracking as a classification task is proposed [7]. In addition, with the advance of end-to-end neural generative models in recent years, many attempts have been made to construct end-to-end trainable frameworks for dialogue systems and promising results have been shown [8]. However, most data-driven approaches are data-hungry, requiring a large amount of data to fully train the model [9]. This poses a big problem to the application in task-oriented dialogue systems, where the domain-specific data are often hard to collect and expensive to annotate.

The goal of this work is to develop efficient task-oriented dialogue systems under a limited amount of training data. To this end, we propose a hybrid approach which leverages the benefits from both rule-based and data-driven approaches. We first establish intent-slot trees from the standard multi-turn dialogue corpus in specific domain. The intent-slot tree then servers as rules for subsequent dialogue. During the dialogue, the power of deep language understanding model is exploited to enhance the generalization ability of the system and the multi-turn dialogue proceeds following the path of the intent-slot tree established. Our main contributions are as follows:

1. We define an intent-slot tree structure to implicitly specify rules for the dialogue system at low cost.
2. Based on the intent-slot tree defined, we present a rule-based and data-driven hybrid approach for building efficient task-oriented dialogue systems where the domain-specific data is insufficient for full training deep learning models.

2 Methodology

As mentioned above, in task-oriented dialogue circumstance, both the rule-based and the data-driven approaches have their own advantages and disadvantages. We attempt to

take the benefits from the both approaches. Our main idea is to integrate domain-specific rules into the dialogue system in a simple and effective way. To do so, we propose the intent-slot tree model to represent and organize the rules, based on which the dialogue system is built.

2.1 Overview

The proposed dialogue system consists of three principle components: mode selection, single-turn dialogue and multi-turn dialogue. In mode selection, the mode of single-turn dialogue or multi-turn dialogue is chosen based on a joint intent-slot model. In single-turn dialogue, a two-stage text matching algorithm is used. The matching algorithm combines relevance matching and semantic matching for retrieving proper answers to simple questions or greetings, where the semantic matching is based on semantic matching model. While in multi-turn dialogue, the user will be guided in completing his or her tasks following intent-slot trees previously constructed. The system structure is illustrated in Fig. 1. In the following sections, we first introduce the proposed intent-slot tree model and then present the above principle components in detail.

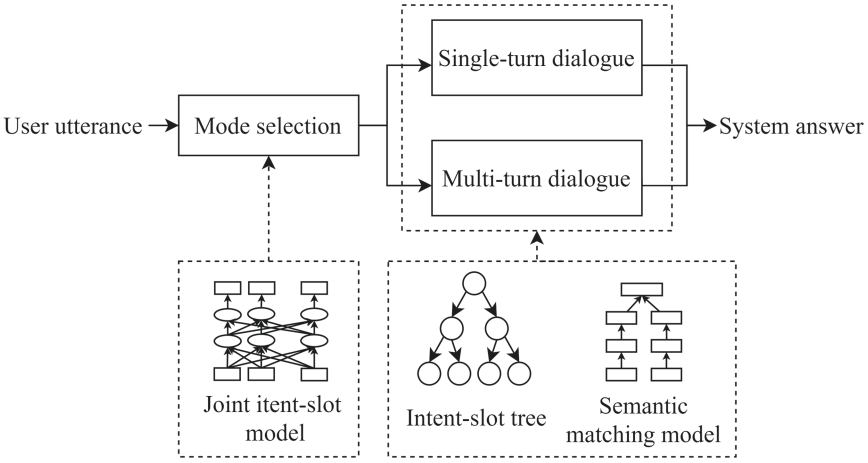

Fig. 1. The proposed system framework.

2.2 Intent-Slot Tree

Before getting into the details of the proposed intent-slot tree, we first introduce the corpus we used.

Dialogue Corpus. The corpus contains multiple standard multi-turn dialogues between the users and the robot. One dialogue represents one typical interaction scene that the robot helps the user in completing one specific task. for notation convenience, the content of the dialogue is logically represented in a tree structure (hereinafter referred to as

dialogue tree $\overline{T}$). Suppose the level number of the tree starts from 1, i.e., level 1 is the first level, so the largest level number of $\overline{T}$ is even. The nodes in odd levels of the tree store user sentences, and each one of which has only one child node. The nodes in even levels store the robot sentences, and each of which has one or more child nodes. Figure 2 gives an example of dialogue tree that represents one typical interaction between the use and the robot in bank business consultation.

Intent-Slot Tree Construction. For one input dialogue tree $\overline{T}$, we target to construct a corresponding intent-slot tree T. the construction procedure consists of the following steps:

Intent Labeling. Assign a user's intent u for the input dialogue tree $\overline{T}$. Take the dialogue tree depicted in Fig. 2 for example, the intent u assigned is "consult + deduction + fail".

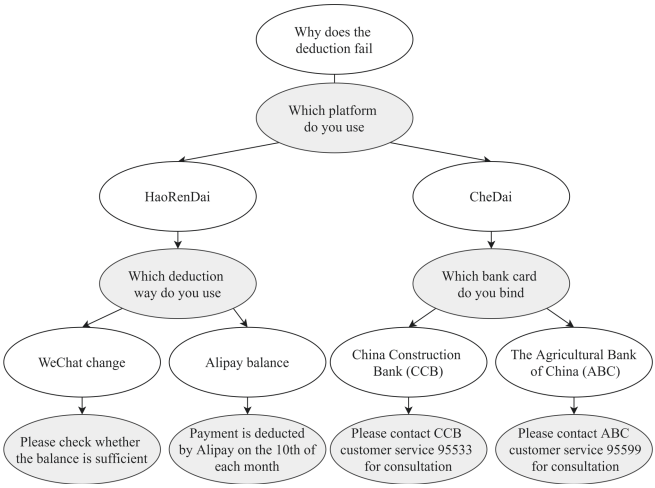

Fig. 2. An example of the multi-turn dialogue tree. White and gray nodes represent user and robot parts, respectively.

Root Node Creation. 1) Create a pair (u, a), where a is the sentence contained in the child of root of $\overline{T}$. 2) Create a tree node n with pair (u, a) as its content, and add the node n to empty T as its root.

Other Nodes Creation. For each node in odd level of $\overline{T}$ do: 1) Extract one slot value s manually for the sentence contained in current node. 2) Create a pair (s, a'), where a' is the sentence contained in the child of current node. 3) Create a tree node n' with pair (s, a') as Its content, and add the node n' to T as the child of the node that corresponds to the grandparent of current node in $\overline{T}$.

Figure 3 shows an example of an intent-slot tree created from the dialogue tree in Fig. 2.

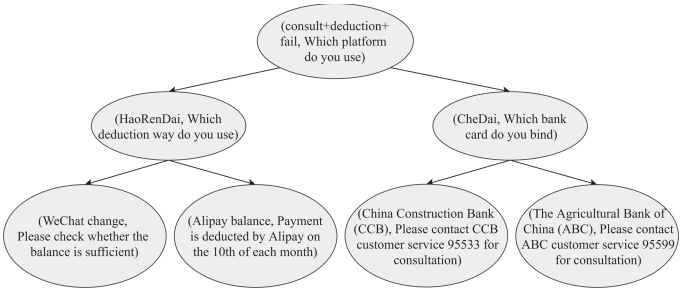

Fig. 3. Example of an intent-slot tree created from the dialogue tree in Fig. 2.

2.3 Mode Selection

As the first step of our method, we use a joint intent-slot model to extract intent from the user input utterance. Based on the result of intent extraction, the mode of the following dialogue is determined.

Joint Intent-Slot Model. We use BERT-based intent-slot joint recognition model [10]. Figure 4 Illustrates a high-level view of the model.

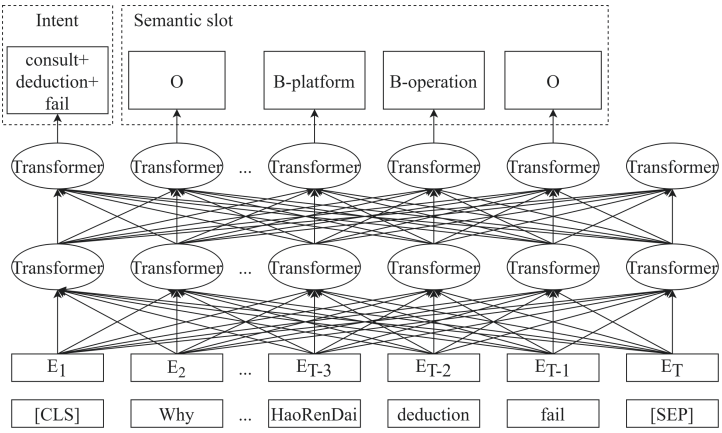

Fig. 4. A high-level view of BERT-based intent-slot joint recognition model. The input sentence is "Why does HaoRenDai deduction fail".

BERT [11] can be easily extended to a joint intent classification and slot filling model. A special classification embedding ([CLS]) is inserted as the first token and a special token ([SEP]) is added as the final token. Given an input token sequence $x = (x_1, \ldots, x_T)$, the output of BERT is $H = (h_1, \ldots, h_T)$.

Based on the hidden state of the first special token ([CLS]), denoted h_1, the intent is predicted as:

$$y^i = softmax\left(W^i h_1 + b^i\right) \tag{1}$$

For slot filling, this model feed the final hidden states of other tokens $h_2, \ldots, h_T$ into a softmax layer to classify over the slot filling labels, the formula is expressed as.

$$y_t^s = softmax\left(W^s h_t + b^s\right), t \in 2 \ldots T \tag{2}$$

where h_t is the hidden state corresponding to the first sub-token of word x_t.

To jointly model intent classification and slot filling, the objective is formulated as:

$$p\left(y^i, y^s|x\right) = p\left(y^i|x\right) \prod_{n=1}^{N} p\left(y_n^s|x\right) \tag{3}$$

The learning objective is to maximize the conditional probability $p\left(y^i, y^s|x\right)$. The model is fine-tuned end-to-end via minimizing the cross-entropy loss.

Dialogue Mode Selection. For a user input utterance, we try to extract the main intent u using the joint intent-slot model previously trained. If the intent is extracted successfully, the multi-turn dialogue mode is chosen and the intent-slot tree whose intent matches u is selected as the matching tree T_M for the following multi-turn dialogue. Otherwise, the single-turn dialogue mode will be selected for completing the dialogue task.

2.4 Single-Turn Dialogue

In real task-oriented dialogue, users often have simple questions which are frequently asked in specific domain or even they just give greetings. In that case, the system may just give the answer directly. We design a framework specific for such single-turn dialogue situations.

Firstly, we collect the frequently asked questions and their standard answers. We organize them as questions and answer (QA) pairs in database. Then, during the dialogue, the system retrieves the question in database that matches the user utterance best and returns its answer to the user. Thus, the key task here is to develop an accurate and efficient way to perform the match task. Toward this end, we propose a two-stage matching framework for retrieving the matched question in database. In the first stage, a group of relevant questions are retrieved through relevance matching. In the second stage, the similarity between the retrieved questions and the user utterance are computed using similarity matching technique. Finally, the overall score of each retrieved question is computed by combining the scores obtained in two stages and the question with highest score is selected as the question of best match. The framework of the single-turn dialogue is shown in Fig. 5.

In the first stage, we use the BM25 [12] to perform the relevance matching. BM25 is a ranking function to estimate the relevance of documents to given search query. BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document.

In the second stage, we build the Sentence-BERT (SBERT) [13] semantic matching model to perform the similarity matching. SBERT uses a Siamese network structure based on BERT which calculates the fixed sized sentence embeddings. At inference,

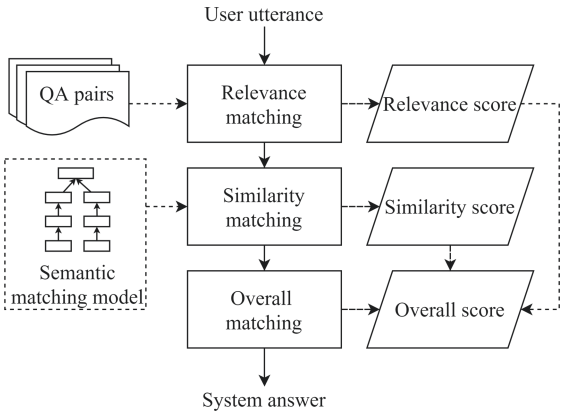

Fig. 5. The basic flow of the single-turn dialogue.

SBERT computes the similarity score of two sentences by calculating the cosine similarity of their embeddings. In training the SBERT model, we use the corresponding pre-trained model for initialization and fine-tune the parameters on a few domain-specific data.

In the overall matching, we integrate the literal relevance and the semantic similarity. Suppose the relevance score is S_1 and the similarity score is S_2, then the overall score S_3 is calculated by the following formula:

$$S_3 = \alpha \cdot S_1 + (1 - \alpha) \cdot S_2 \tag{4}$$

with α the adjustment factor which is derived from experiments.

2.5 Multi-turn Dialogue

In contrast to most data-driven approaches, our policy for multi-turn dialogue does not heavily rely on the model trained by the data. Instead, the dialogue proceeds following the path of the intent-slot tree established from the standard corpus. The whole procedure consists of following two steps: start point location and tree matching dialogue.

Start Point Location. In the previous mode selection step, we have obtained the matching tree T_M which matches the intent extracted from the user utterance. Besides intent, one or more slot values may also be extracted. We define a set SV to contain all these slot values. The values in SV may happen to be equal to the ones in the child nodes of the root of T_M. In this case, it is inappropriate to start the dialogue from the root of T_M, since the user will have unpleasant feelings when being asked the questions whose answers have been previously given. So once the matching tree has been selected, it is better to find its node that is most appropriate for starting the multi-turn dialogue. We achieve this by executing the following matching procedure.

Suppose the matching tree has M leaves, then from root to leaf there are M different paths. For each of these M paths, starting from its second node, we try to find a consecutive

node list in which the slot value of each node matches one of element of SV. Here, we utilize the semantic matching technique described in the previous section to measure the matching degree. Then we compare the lengths of lists of all paths and select the one that has the longest length. The elements which match the nodes of the selected list are removed from SV. Finally, the procedure returns the last node of the selected list. This node severs as the start node of the following dialogue.

Tree Matching Dialogue. From the start point, the dialogue runs iteratively following one path of the T_M. In each iteration, each child of the current node is matched against all elements of SV. Again, we utilize the semantic matching technique to measure the matching degree. If there is a successful match, we remove the corresponding element from SV and the current iteration ends by setting the corresponding child node as the working node for next iteration. Otherwise, the robot sentence contained in the current node is read and replied to the user. After the user gives a response, the slot values are then extracted from the response and the one with the highest prediction confidence is selected. Then we match each child of the current node against the selected slot value semantically. If there is a successful match, we set the corresponding child node as the working node for next iteration. Otherwise, the following re-matching procedure is launched for the first time and out-of-domain process is executed for the other time. The dialogue continues until the leaf of the tree is reached.

Re-matching Strategy. Like other pipeline-based systems, our dialogue system faces the problem of error propagation. If the NLU module does not correctly extract the semantic slots from the user utterance in multi-turn dialogue, the above tree matching will fail, leading to the failure of the dialogue. To circumvent this problem, we propose the following strategy. For Each node of the intent-slot tree, we maintain a weighting parameter w and initialize it to 0. The weight w is increased by 1 once the corresponding node has been visited. For the case that none of the children of the current node can match the user slot value, a single-choice question is raised by the system. The options of the question come from slot values of the children with m highest weights. Based on the new reply given by the user, the tree matching dialogue will continue as usual.

3 Experiments

We extracted 1283 multi-turn dialogue corpus and 2000 frequently asked simple questions from dialogue records in bank business consultation. From the corpus, we labeled 45 intents and 13 slots, and for each intent, we built an intent-slot tree.

3.1 Baselines for Comparison

We use the open-source toolkit, ConvLab-2 [14], to build two dialogue systems as baselines. The first system, named Baseline (Rule), is mainly based on rules and the second one, named Baseline (Learning), employs deep learning method for dialogue state tracking. Table 1 shows the components modules of baseline systems and ours in detail.

Table 1. The composition of each module of baselines and our system.

	NLU	DM	NLG
Baseline (Rule)	Joint Intent-slot	RuleDST + Rule Policy	TemplateNLG
Baseline (Learning)	-	TRADE + Rule Policy	TemplateNLG
Our system	Joint Intent-slot	Intent-slot tree	-

We use a set of rule-based models in ConvLab-2 in this experiment. Specifically, the RuleDST model and the Rule Policy model are chosen for DM, and the TemplateNLG model is deployed for NLG. For the Baseline (Learning) system, we deploy the TRADE (Transferable Dialogue State Generator) [15] model for dialogue state tracking, in which an end-to-end deep learning method is used to directly generate dialogue states from user utterances. For the Baseline (Rule) and our system, we implement the joint intent-slot model introduced in Sect. 2.3 for NLU.

3.2 Results and Analysis

Model Performance. We evaluate the performance of the entire dialogue system on task finish rate T which is defined as follows:

$$T = \frac{M_f}{M_t} \tag{5}$$

where M_f is the number of multi-turn dialogues that successfully completed the task, and M_t is the total number of multi-turn dialogues.

Table 2. Performance comparison between baselines and our system.

	NLU Accuracy (%)	DM Accuracy (%)	System Task finish rate (%)
Baseline (Rule)	85.43	77.52	69.14
Baseline (Learning)	-	68.62	60.75
Our system	85.43	82.19	76.21

Table 2 shows the performance comparison of baselines and our system. As can be seen from the table, our system shows the best performance among the three systems. Compared with baseline (Rule), our system achieves accuracy gain of 4.67% for DM module and 7.07% improvement on the task finish rate. Compared with baseline (Learning), our system achieves gains of 13.57% and 15.46% for DM accuracy and task finish rate, respectively. The inefficiency of baseline (Learning) system is due to the fact that there is no sufficient labeled training data to fully train the model.

Table 3. Performance comparison between the system with and without re-matching strategy. Sys-with represents the system with re-matching strategy, Sys-without represents the system without it. Avg. turns represents the average number of turns of all test dialogues.

	Task finish rate	Avg. turns
Sys-without	67.41	3.3
Sys-with	76.21	4.5

Re-matching Strategy. To demonstrate the effectiveness of re-matching strategy, we compare the performance between our system and the one without re-matching strategy. The results are shown in Table 3.

The task finish rate of the system with re-matching strategy is significantly higher than that without it. We notice that the average number of turns of our system with re-matching strategy is larger than that without it. This is because that during re-matching process, the system raises single-choice questions to the user, leading to the increase of the number of dialogues turns. Whereas in case of the system without re-matching strategy, the system just launches the out-of-domain process which directly ends the whole dialogue.

4 Conclusion

This paper presents a practical way to develop task-oriented dialogue systems under a limited amount of training data. The proposed approach leverages the benefits from both rule-based and data-driven approaches. The intent-slot tree is built to implicitly specify rules for the dialogue system at low cost. Meanwhile, the power of deep language understanding model is used to enhance the generalization ability. Future work is expected to extend this method to deal with more complex tasks, such as cross-domain dialogue applications.

References

1. Chen, H., Liu, X., Yin, D., et al.: A survey on dialogue systems: recent advances and new frontiers. ACM SIGKDD Explor. Newsl. **19**(2), 25–35 (2017)
2. Nakano, M., Miyazaki, N., Yasuda, N., et al.: Wit: a toolkit for building robust and real-time spoken dialogu systems. In: 1st SIGdial Workshop on Discourse and Dialogue, pp. 150–159 (2000)
3. Goddeau, D., Meng, H., Polifroni, J., et al.: A form-based dialogue manager for spoken language applications. In: IEEE Processing, vol. 2, pp. 701–704 (1996)
4. Zue, V., Seneff, S., Glass, J.R., et al.: JUPlTER: a telephone-based conversational interface for weather information. IEEE Trans. Speech Audio Process. **8**(1), 85–96 (2000)
5. Thomson, B., Young, S.: Bayesian update of dialogue state: a POMDP framework for spoken dialogue systems. Comput. Speech Lang. **24**(4), 562–588 (2010)

6. Young, S., Gašić, M., Keizer, S., et al.: The hidden information state model: a practical framework for POMDP-based spoken dialogue management. Comput. Speech Lang. **24**(2), 150–174 (2010)
7. Williams, J.D., Raux, A., Henderson, M.: The dialog state tracking challenge series: a review. Dialogue Discourse **7**(3), 4–33 (2016)
8. Zhao, T., Eskenazi, M.: Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In: Proceedings of the SIGDIAL 2016 Conference, pp. 1–10. Association for Computational Linguistics, Los Angeles (2016)
9. Zhang, Z., Takanobu, R., Zhu, Q., Huang, M., Zhu, X.: Recent advances and challenges in task-oriented dialog systems. Sci. China Technol. Sci. **63**(10), 2011–2027 (2020). https://doi.org/10.1007/s11431-020-1692-3
10. Chen, Q., Zhuo, Z., Wang, W.: BERT for joint intent classification and slot filling. ArXiv preprint arXiv:1902.10909 (2019)
11. Devlin, J., Chang, M.W., Lee, K., et al.: 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, pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (2019)
12. Robertson, S., Hugo, Z.: The probabilistic relevance framework: BM25 and beyond. Now Publishers Inc. (2009)
13. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. ArXiv preprint arXiv:1908.10084 (2019)
14. Zhu, Q., Zhang, Z., Fang, Y., et al.: Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. ArXiv preprint arXiv:2002.04793 (2020)
15. Wu, C.S., Madotto, A., Hosseini-Asl, E., et al.: Transferable multi-domain state generator for task-oriented dialogue systems. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 808–819. Association for Computational Linguistics, Florence, Italy (2019)