Integrating Pre-trained Model into Rule-based Dialogue Management

Jun Quan[1,2]*, Meng Yang[2], Qiang Gan[2], Deyi Xiong[1], Yiming Liu[2], Yuchen Dong[2], Fangxin Ouyang[2], Jun Tian[2], Ruiling Deng[2], Yongzhi Li[2], Yang Yang[2] and Daxin Jiang[2]

[1] School of Computer Science and Technology, Soochow University, Suzhou, China
[2] STCA NLP Group, Microsoft
terryqj0107@gmail.com, dyxiong@suda.edu.cn, {menyan, qigan, yiml, yucdong, faouya, juti, ruilingd, yongli, yayan, djiang}@microsoft.com

Abstract

Rule-based dialogue management is still the most popular solution for industrial task-oriented dialogue systems for their interpretablility. However, it is hard for developers to maintain the dialogue logic when the scenarios get more and more complex. On the other hand, data-driven dialogue systems, usually with end-to-end structures, are popular in academic research and easier to deal with complex conversations, but such methods require plenty of training data and the behaviors are less interpretable. In this paper, we propose a method to leverages the strength of both rule-based and data-driven dialogue managers (DM). We firstly introduce the DM of Carina Dialog System (CDS, an advanced industrial dialogue system built by Microsoft). Then we propose the "model-trigger" design to make the DM trainable thus scalable to scenario changes. Furthermore, we integrate pre-trained models and empower the DM with few-shot capability. The experimental results demonstrate the effectiveness and strong few-shot capability of our method.

Introduction

Task-oriented dialogue systems usually consist of three main components, including natural language understanding, dialogue management, and natural language generation. The Dialogue Manager (DM) keeps tracking the current dialogue state and determines the next action to be taken.

Rule-based DMs are widely used in industrial task-oriented dialogue systems, such as Google's Dialog Flow and Microsoft's Bot Framework. However, due to manual rules and rigid structure of dialogue flow, it's hard for developers to maintain a rule-based DM, especially when the scenario is complex. In recent years, data-driven approaches for task-oriented dialogue, usually with end-to-end architectures, have been an active area in the research community. Wu et al. (2020) propose a task-oriented dialogue BERT (ToD-BERT), which is trained on 9 English task-oriented datasets across over 60 domains and achieves good performance on 4 dialogue subtasks.

We are motivated to leverage both advantages of the methods from industrial production and academic research. We hope to effectively reduce the cost in manually developing DM rules while ensuring the performance.

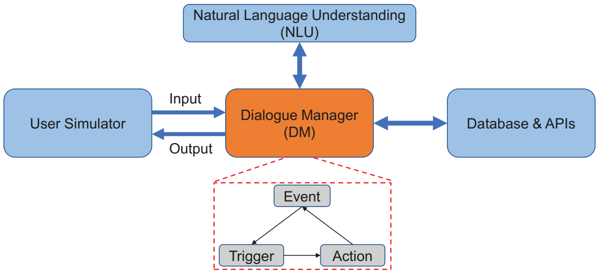


Figure 1: The architecture of Carina Dialog System

Based on CDS, we maintain its basic structure and incorporate the data-driven methods into the DM module. Specifically, we maintain the core architecture (i.e. Event-Trigger-Action mechanism) of the rule-based solution of the original DM to inherit its controllability, and take "model-trigger" design to replace original expression triggers. Furthermore, we integrate a task-oriented dialogue pre-trained model to make full use of the rich external linguistic knowledge learned from pre-training.

The contributions of our work are as follows. 1) We introduce an advanced dialogue management scheme of the industry-scale Carina Dialog System. 2) We introduce the "model-trigger" design in rule-based DM to achieve data-driven capability. 3) We propose an approach to integrating pre-trained model to DM. 4) We define dialogue actions in CDS based on CamRest676 dataset and re-annotate the dataset to evaluate our method. 5) We design experiments to prove the feasibility and strong few-shot capability of our method. 6) We build a demonstration system to visualize the process of our model-trigger DM [1].

Method

Carina Dialog System (CDS). It is an industrial task-oriented dialogue system, which has been applied to many real-world scenarios. The overall architecture of CDS is shown in Figure 1. It consists of four main components: 1) a User Simulator module; 2) a Natural Language Understanding (NLU) module; 3) a Dialogue Manager (DM); and 4) a

*Work done while Jun Quan was a research intern in Microsoft. Qiang Gan is the corresponding author (qigan@microsoft.com).

[1] Link to our video: https://youtu.be/suQTJ-L_3j4

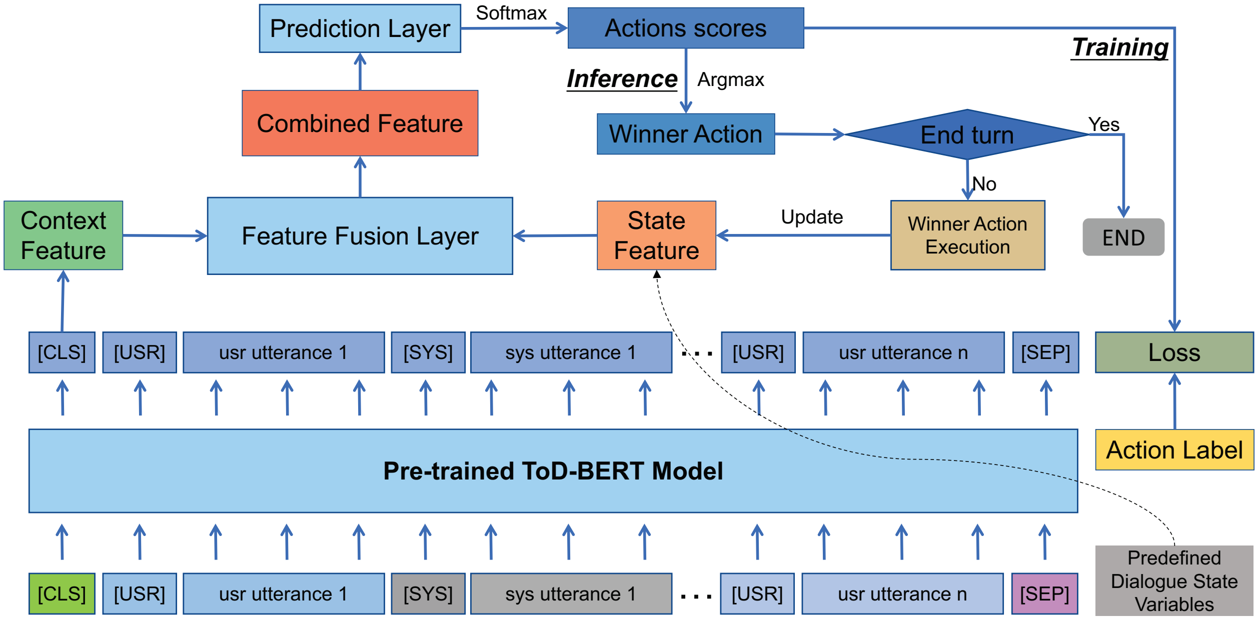


Figure 2: The architecture of Model-Trigger DM

Database & APIs module. In our work, we focus on the DM which decides next actions according to dialogue state.

Event-Trigger-Action. This is the core mechanism of the DM. The system receives an external event (e.g. *Start*, *Query* or *End*) when each turn starts. All triggers will be evaluated when new events come. If the triggering condition is satisfied for a certain trigger, it will be activated and its corresponding action will execute. Then dialogue state will be updated accordingly and system will generate new internal events. The "event-trigger-action" process is running as a loop until no trigger is activated any more. A winner action sequence (containing one or more actions) is selected out when a turn ends. We refer to the process of predicting each action as mini-turn.

The original trigger module of CDS is implemented through the expression trigger, where firing conditions are formed with manually written expressions composed of dialogue state variables, which can "listen to" various customized events. Developers can build flexible and interpretable dialogue logics with expressions. However, it is hard for them to maintain these logic rules when the scenario becomes complex. So we propose the "model-trigger" design to achieve data-driven capability and reduce developers' efforts to write expressions.

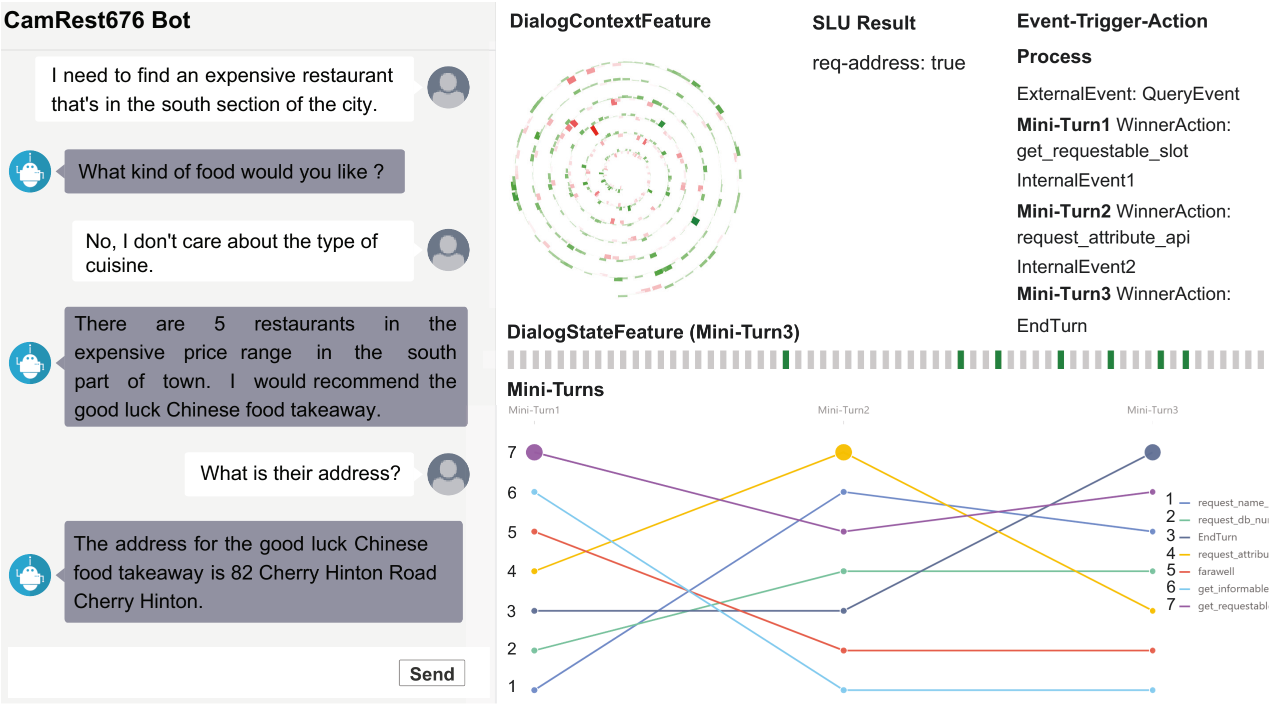
Model-Trigger DM. We are motivated to leverage a neural model to predict the winner action of each mini-turn according to not only the dialogue state used in expression triggers, but also the dialogue context. Thus, we hope the model trigger can cover most of the manually defined expressions. The architecture of our model-trigger DM is shown in Figure 2. In each mini-turn, the model takes two parts of features to predict a winner action, including context feature and state feature. 1) **Context Feature:** All utterances in dialogue context are concatenated into a flat sequence as the input. We use ToD-BERT to encode the context sequence and take the embedding of the "[CLS]" token as the context feature. 2) **State Feature:** We treat the binarization of pre-defined dialogue state variables during the process, which updates in each mini-turn, as the state feature.

We formulate the action prediction as a multi-class classification problem. The two parts of features are combined through a feature fusion layer and used to predict the winner action of each mini-turn over all possible actions through a prediction layer. Both the feature fusion layer and the prediction layer are implemented with deep neural networks.

To evaluate our method, we create a new version of CamRest676 dataset (Wen et al. 2016a,b) and conduct experiments on it. We define executable dialogue actions and annotate each turn of all dialogues with dialogue action labels. Each turn may contain a sequence of action labels, which means the system response in this turn is generated during the execution of those actions in order. Then we feed the dialogue sessions with labels into CDS and run in a "data-collection mode". In this mode, DM will execute the labeled actions in order to update the internal dialogue state features, while dialogue context features are fixed for each turn. Each turn may produce more than one training data records, corresponding to the number of action labels. Experiments show that it can achieve 95.73% of mini-turn accuracy to predict dialogue action, which is a great performance. Furthermore, few-shot experiments show a satisfactory performance using only small-scale training data.

Demonstration

As shown in Figure 3, we build a demonstration system based on our model-trigger DM. The main function of our system is to show the process of predicting dialogue actions according to the dialogue context and the current state. On the left side of the interface are the dialogue context display area and input area. On the right side, the results of



CamRest676 Bot

I need to find an expensive restaurant that's in the south section of the city.

What kind of food would you like ?

No, I don't care about the type of cuisine.

There are 5 restaurants in the expensive price range in the south part of town. I would recommend the good luck Chinese food takeaway.

What is their address?

The address for the good luck Chinese food takeaway is 82 Cherry Hinton Road Cherry Hinton.

SLU Result

req-address: true

Event-Trigger-Action Process

ExternalEvent: QueryEvent

Mini-Turn1 WinnerAction: get_requestable_slot

InternalEvent1

Mini-Turn2 WinnerAction: request_attribute_api

InternalEvent2

Mini-Turn3 WinnerAction: EndTurn

Figure 3: Demo showing the process of Model-Trigger DM.

natural language understanding in each turn are shown in the form of semantic frame at the top part. We visualize the 768-dimensional dialogue context feature and the 64-dimensional dialogue state feature in each mini-turn. The probability distribution of predicting dialogue actions in each mini-turn is shown at the bottom part. Through the demonstration system, we can clearly and intuitively observe the whole process of predicting dialogue actions by our model-trigger DM.

Conclusion and Future Work

In this paper, we introduce Carina Dialog System and propose "model-trigger" to reduce human efforts. We further integrate a pre-trained model for few-shot capability. Experiments show the good performance of our method. We build a demonstration system to visualize the process of the model-trigger DM. In our future work, we will further improve the model-trigger DM and apply it to the real-world products to reduce the manual efforts of traditional solution.

Acknowledgments

This work is supported by Microsoft STCA NLP Group. We would also like to thank the anonymous reviewers for their insightful comments.

References

Wen, T.-H.; Gasic, M.; Mrkšić, N.; Barahona, L. M. R.; Su, P.-H.; Ultes, S.; Vandyke, D.; and Young, S. 2016a. Conditional Generation and Snapshot Learning in Neural Dialogue Systems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2153–2162.

Wen, T.-H.; Vandyke, D.; Mrksic, N.; Gasic, M.; Rojas-Barahona, L. M.; Su, P.-H.; Ultes, S.; and Young, S. 2016b. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 438–449.

Wu, C.-S.; Hoi, S.; Socher, R.; and Xiong, C. 2020. Tod-bert: Pre-trained natural language understanding for task-oriented dialogues. *arXiv preprint arXiv:2004.06871* .