Symbolic Dialogue for general Domain State Tracking

Hung Nguyen Quoc
Department of Information Technology
FPT University
Ho Chi Minh, Vietnam
hungnqse162007@fpt.edu.vn

Tien Nguyen Van
Pythera AI
tien.nguyen@pythera.ai

Hieu Pham Trung
Pythera AI
hieu.pham@pythera.ai

Trung Nguyen Quoc
Department of Information Technology
FPT University
Ho Chi Minh, Vietnam
trungng46@fpt.edu.vn

Vinh Truong Hoang
Faculty of Information Technology
Ho Chi Minh City Open University
Ho Chi Minh City, VietNam
vinh.th@ou.edu.vn

Tuan Le Viet
Faculty of Information Technology
Ho Chi Minh City Open University
Ho Chi Minh City, VietNam
tuan.lv@ou.edu.vn

***Abstract*—Task-oriented dialogue (TOD) systems are built to help users accomplish specific objectives. However, even though ongoing reviews and improvements have been made to its components, an official industrial standard has yet to be established. Additionally, TOD systems face limitations in detecting out-of-scope events, deciding when to access a database, and offering scalability for further processing. To address these issues, we introduce a comprehensive TOD framework and present solutions to overcome these limitations. We also investigate dialogue state tracking (DST), the initial phase of the system, and assess how well it can identify out-of-scope events triggered by user actions not predefined in the conversation.**

***Index Terms*—task-oriented dialogue, schema-guided, symbolic reasoning, open-domain dialogue, dialogue system, schema actions**

I. Introduction

AI has become an essential tool for assisting humans in everyday tasks, particularly in roles such as instruction. Task-oriented dialogue (TOD) systems, for instance, are designed to help users achieve specific goals through conversation. In customer support services, workers need to understand a company's products and how to sell them effectively. However, the sales process often involves more than a simple transaction, extending into areas like technical questions or product comparisons. These situations, known as out-of-scope queries, require employees to either consult guidelines or escalate the case to a specialized department. In contrast, an open-domain dialogue (ODD) system can better manage such diverse cases by retrieving information in less constrained, non-predefined circumstances.

According to [1], TOD systems can be divided into four components: Natural Language Understanding (NLU), Dialogue State Tracking (DST), Dialogue Policy (DP), and Natural Language Generation (NLG). Dialogue State Tracking (DST) focuses on predicting the current dialogue slot values within a conversation. Often, this involves identifying the slots that are being requested or informed. The schema-guided method uses a well-documented schema, detailing possible slots and intents, to provide additional context. This approach enhances NLU tasks and reduces the likelihood of inconsequential conversations caused by ambiguous statements, while also enabling zero-shot capabilities.

While schema-based descriptions are useful, language models often struggle to maintain a complete dialogue flow in TOD systems. To tackle this challenge, we've restructured the TOD system into three main components: State Tracking, Action Decision, and Response Generation. Unlike the traditional separation [1], we combine the NLU and DST modules into the State Tracking task to leverage implicit context between them. Additionally, we introduce two new tags: ***target_acts*** and ***dependencies***, corresponding to the user's intent and the chain of actions to achieve that intent. Additionally, we've implemented ***undefined_actions*** for both user and system interactions. These are designed to improve out-of-scope detection, as we believe these explicit signals will help the system make more informed decisions about subsequent steps in the dialogue.

II. Literature review

State Tracking typically involves several subtasks such as slot filling and handling user request slots. BERT-based encoder architectures have traditionally been used for these tasks. Although these encoder-based models perform well, they tend to be inefficient because of the resource-intensive pre-training processes required [2] [3] or because of their complex structures [4] [5] [6] [7] [8]. FastSGT [9] categorizes encoder models into single-pass and multi-pass types. Although the single-pass model is simpler, its performance is not as impressive as that of multi-pass models in benchmark evaluations. Consequently, in-context learning [10] comes around by leveraging more deeply grounded knowledge.

Two main approaches have emerged from this: **end-to-end** and **modular**. End-to-end systems work by frequently concatenating previous dialogue turns into the context, but they struggle with long input sequences. Moreover, these systems are prone to accumulating errors when the generated responses are not accurate. On the other hand, modular approaches, which are more commonly used in software design, are preferred for their flexibility and ability to improve individual components independently. This method divides the system into distinct sub-modules, improving both tracking performance and system adaptability.

End-to-end method. SimpleTOD [11] is a unified language model that integrates all TOD modules into a single system. This structure allows for a straightforward, iterative inference process, utilizing all available contexts to support predictions. However, SimpleTOD faces challenges in efficiency and struggles with noisy datasets like MultiWOZ [12]. It also lacks fine-tuning, which limits its performance. In contrast, SPACE-3 [13] adopts a multi-task approach with a better-quality dataset, though it has its drawback, requiring separate decoders for each TOD task. Beyond these models, there are numerous areas for further improvement, including enhancing schema robustness [14] [15], developing task-optimized adapter architectures [16], and using large language models with schema support [17].

Task-oriented with open-domain dialogue. Conversations are not just about reaching a goal, but also about providing social support. Therefore, preventing bias in TOD systems is crucial. UniDS [18] addresses this by combining chit-chat datasets with TOD datasets to handle informal dialogue scenarios. OPERA [19] takes a different approach by introducing knowledge grounding to ensure accurate responses to open-domain utterances, making it effective for both question-answering and TOD tasks. These models, along with advancements in dataset quality [20], suggest that open-domain dialogue systems should use retrieval models as knowledge validators.

Language model for State Tracking. Given the slot information and the conversation, the model must identify all the slots and values. However, slot filling alone is not sufficient for achieving the desired goals; the system also needs to track request-inform actions and intent. D3ST [21] uses supporting information to describe the slots and intents, while SDT [22] takes a slightly different approach by demonstrating the process and revising it according to the current conversation. However, it is harder to extend due to its reliance on n-shot learning. Despite these innovations, a significant challenge remains: action unification. Actions vary across different TOD datasets, making it difficult to track dialogue policy consistently. To address this, researchers have proposed methods such as unified action datasets [14] and extracting action latent spaces [23] [24].

Symbolic methods. [25] firstly describe the effectiveness of writing syntax for representing specific meaning. [26] explored symbolic methods in large language models for mathematical reasoning, and several studies have shown that symbolic reasoning [27] [28] [29] enhances model performance on reasoning tasks. Models like SDT [21] and D3ST [22] implicitly utilize symbolic methods to achieve state-of-the-art results. By introducing symbolic tokenization, models may learn to associate tokens with more generalized concepts, potentially improving their ability to generalize to unseen data, even in distinct domains. Building on symbolic methods, AnyTOD [30] combines language models with flexible action sets by explicitly incorporating possible actions into the input prompt. Similarly, GradTOD [31] integrates both task-oriented and open-domain dialogue capabilities by using symbolic methods to detect the type of dialogue, which leads to competitive results across various domains.

III. Method

Natural language, commonly used in everyday conversation, is highly flexible and incorporates various attributes such as emotion and context. However, when context is lacking, it often becomes ambiguous, requiring humans to supply additional external context to clarify meanings and expressions. Machines, in contrast, cannot provide context independently, relying solely on external sources. Programming language, on the other hand, is the opposite of natural language. It adheres to strict syntax rules, eliminating the risk of ambiguity or misunderstanding. We believe that a simple solution for TOD systems is to combine the strengths of both languages—leveraging the flexibility of natural language and the clarity of programming language. Based on this approach, we have established three essential criteria to structure the framework:

- Schema-based Guide: The system must be able to work cross-domain with only instructions with defined form.
- Out-of-scope Awareness: Conversation between humans also contains context switching and irrelevant context. Gracefully handling every situation is critical to effectively talking to the end user.
- Scalability: Despite working in a system as a whole, each module needs to independently grow and easily substitute.

A. Task-oriented Framework

As mentioned in Section I, we present a minimal TOD system composed of three core modules: State Tracking, Action Decision, and Response Generation. However, in more complex scenarios, the dialogue flow expands, such as when a user books a hotel and then needs to book a taxi to that hotel. In Fig. 1, an additional module, Intent Detection, is introduced, and the Response Generation module is divided into two submodules: Item-based Response Generation and Document-based Response Generation.

Let's explore each component in Fig 1, from input utterance to response generation. First, Module 1) processes the user's utterances to produce abstract actions and slot values. The primary goal at this stage is to convert natural language

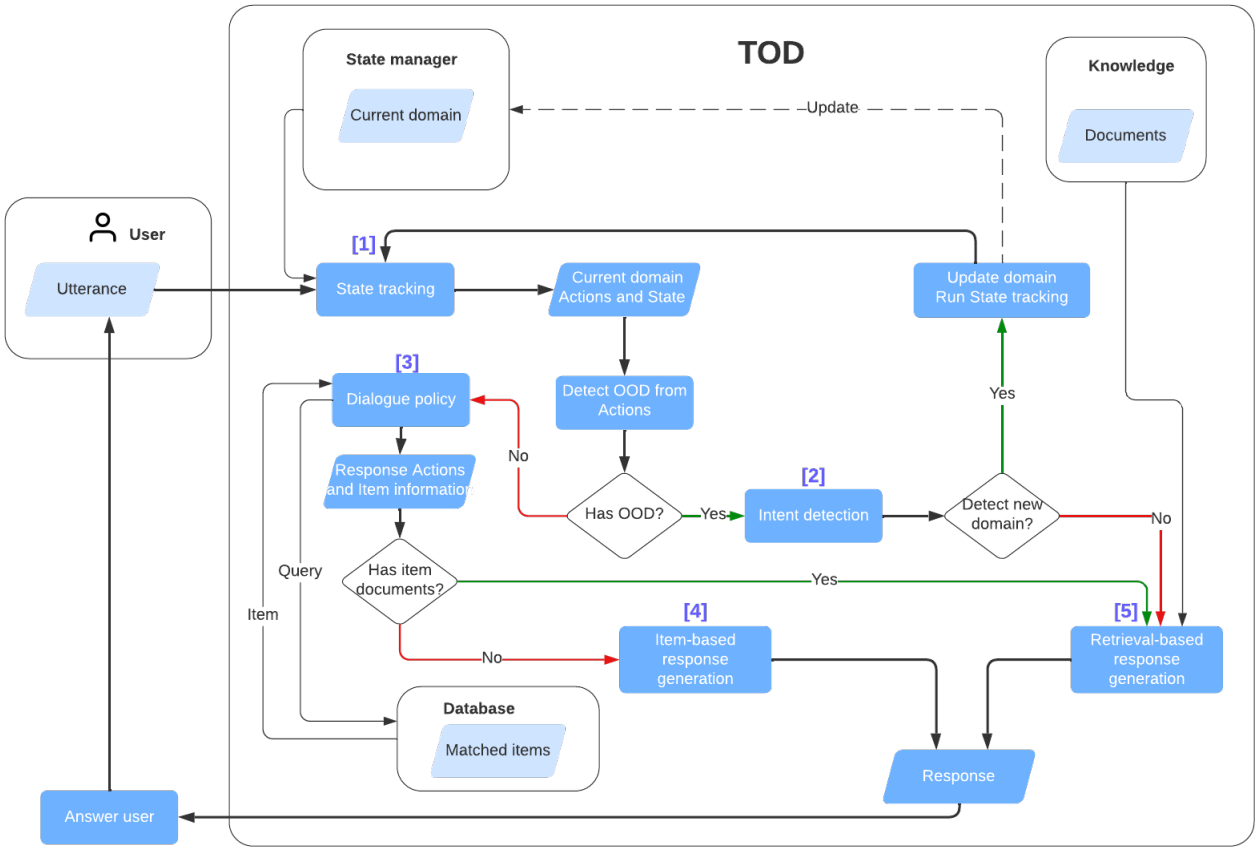

Fig. 1: Overview of the TOD system with retrieval capabilities: State Tracking (1) processes user utterances to extract abstract actions, slot values, and undefined actions. Intent Detection (2) is triggered when undefined actions occur, determining whether a new intent has been introduced. If a new intent is detected, the system restarts the dialogue flow using the schema for that intent. When the schema lacks sufficient information for a response, Dialogue Policy (5) taps into document-based resources to provide a more accurate and informed reply. If State Tracking (1) detects no issues, Dialogue Policy (3) uses the identified actions and slot values to interact with external databases. This interaction provides item-based information for (4) or document-based information for (5).

into parameters and function calls, similar to programming languages. Additionally, it aims to detect ***undefined_actions*** — those not specified in the schema. If undefined actions are detected, Module 2) is used to check if a new intent has been introduced. Module 1) is reused to track this new intent by leveraging the actions already defined in the schema. If no new intent is found, the system identifies actions with descriptive natural language outputs, like "user wants to book a hotel room." If a new intent (undefined action) is detected, the flow restarts with the new intent's schema. If the schema doesn't cover the user's utterance, Module 5) utilizes knowledge documents to generate responses based on previous cases or documented information. If no anomalies are found in Module 1), Module 3) interacts with external storage using the identified actions and slot values. The process concludes by providing abstract actions with context for response generation in Module 4). In some cases, Module 3) may trigger Module 5) to retrieve documents instead of proceeding to Module 4).

Each component has specific requirements and responsibilities. Module 1) must be capable of distinguishing between in-domain information and out-of-scope actions. As the only natural language interface for downstream systems, it must ensure that as much relevant information as possible is captured. This module should also handle noisy input, such as misspellings or abbreviations, requiring extensions to mitigate these issues. Once predicted actions are added to the schema, Module 2) functions similarly to Module 1) since both intents and slots are represented as actions, and its output directs the next step. Module 3) acts as the system's core, responding to actions from Module 1) and interacting with external resources. This module can be implemented with hard-coded programming or using reinforcement learning techniques [32] [33] [34]. The external database then

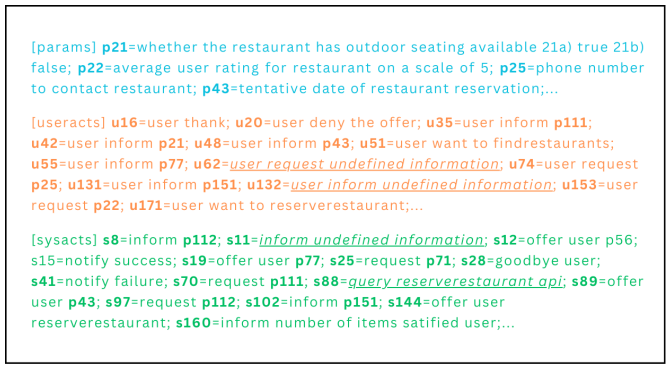

Fig. 2: The scheme-based syntax for user and system actions is organized in the following way. The underlined terms represent ***undefined_actions*** and ***query_<domain_api***. The labels [useracts] and [sysacts] are categorized as **Behaviors**, whereas [params] fall under **Attributes**. The bold terms are linked to a randomization number identity for distinction.

queries relevant data using the conversation context, with each retrieved item linked to unstructured data documents. Module 3)'s output is passed to the next module, where response generation occurs with the added item context. Modules 4) and 5) can use any state-of-the-art model for response generation, such as REML [35], GPT-2 [36], and others.

B. Schema-based Definition

AnyTOD [30] previously proposed actions as recommendations for achieving a zero-shot dialogue policy. However, standard SGD [37] actions do not sufficiently support dialogue policy, prompting us to introduce additional novel actions to the schema, specifically ***inform_undefined***, ***request_undefined***, and ***query_<domain>_api***, as shown in Fig. 2. These "undefined" actions indicate when the system detects domain switches. While the "query" action might seem redundant, it creates opportunities for future advancements in automated Dialogue Policy.

Theorically, State Tracking splits a user's natural utterance into two parts: **Behaviors** and **Attributes**. Behaviors are actions performed through conversation between the user and system, while Attributes are information that helps to define that action. For example, in Fig 2, "user inform p25" is a Behaviour, and "p25=a phone number to contact restaurant" is an Attribute. To extend these Attributes, in Fig. 3, the content of the conversation is also supplied. On top of that, there exist works that leverage the chain of these action shifts. So, we add a weak-supervised information called ***dependencies***. We additionally use ***target_acts*** and hypothesize that the system might automatically decide to query external resources outside without defining another slot value.

To improve generalization, we've introduced randomization for the number identities of slot values and the order of Behaviors in the training dataset (referred to as IFST_Xrand).

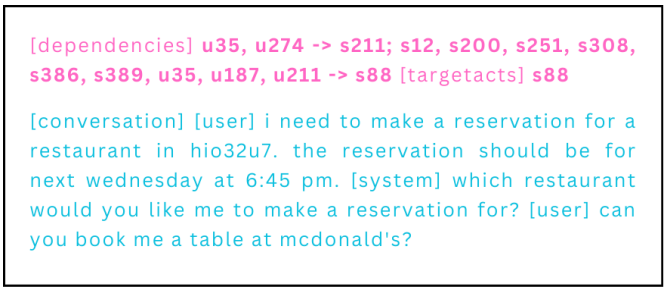

Fig. 3: The scheme-based syntax for ***dependencies*** and ***target_act***. The behavior is executed through a series of actions followed by another action, with semicolons separating each user-system interaction turn.

This approach was inspired by Pix2Seq [38], which found that random output sequence ordering yields the best performance. We believe this randomization will help the model focus more on understanding the surrounding context rather than just the meaning of slot values themselves.

C. In-depth Fine-tuning State Tracking

In-depth fine-tuning adapts models to specific domains by incorporating data from various sub-tasks during training, unlike standard fine-tuning which uses only new domain data. This approach helps the model understand the problem's core rather than focusing on a single aspect. The process targets three main sub-tasks:

- Slot filling: The model uses multiple QAs to outline the dialogue's current state, including available slots and their values.
- Action tracking: The model identifies actions in user and system utterances using action-related tags and given utterances.
- Next action prediction: The model recommends the next action using dialogue history and action-related tags, rather than relying solely on conversation data.

In certain scenarios, a conversation may span multiple domains, with turns from earlier domains consisting of actions specific to those domains. These actions provide significant context for understanding the nuances of different actions. During training, we convert all previous domain request actions into **undefined** requests, akin to inform actions. Additionally, query actions are removed when they fall outside the scope defined by the domain schema. These transformations serve as straightforward methods for the model to learn about out-of-scope contexts, as the number of undefined actions must correspond one-to-one with in-domain actions.

In the inference phase, the model is expected to articulate the last few turns since both states and actions are accumulated and stored in memory. From an efficiency standpoint, separating behaviors and attributes enhances the transformer architecture. The actions and states of one turn may not be affected by the constraints of the input length, allowing this

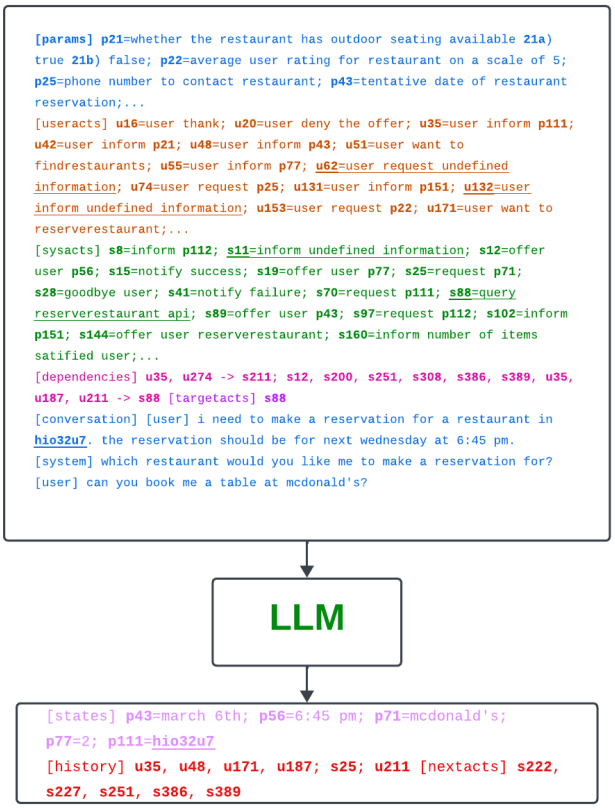

Fig. 4: One State Tracking example for training. Schema Input is all defined, remark that order and number identity are all random, except for conversation and dependencies. Schema Output will be a state of current utterance, history of action's shift, and next actions recommendation.

decomposition to eliminate the necessity for a complete string representation of the input.

IV. Experiments

A. Metrics

Out-of-scope is a situation when the schema is no longer able to handle (domain switching, open domain, etc.). When an action is not represented in the schema but exists in user utterance, the system detects this turn as a domain switching and categorizes it as ***undefined_action***. User-undefined Actions F1 (UUAF1) is the key metric to influence out-of-scope effectiveness. In each turn, the outcome is calculated by:

$$outcome = \begin{cases} 1, & \text{if } \textbf{\textit{undefined\_action}} \text{ exists} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

For each dialogue, the F1 score is calculated using the outcome of 1. To recall that the F1 formula is

$$F_1 = \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2)$$

Additionally, there are three other metrics utilized in State Tracking research: Joint Goal Accuracy (JGA) for slot filling, Action F1 (AF1) for action tracking, and System Action F1 (SaF1) for next action prediction. These metrics differ technically from those in AnyTOD [30].

Firstly, the AF1 score encompasses tracking actions from both the user and the system. Secondly, the SaF1 metric specifically records only system query and inform actions, rather than capturing all defined actions in the SGD framework. The primary reason for this approach is to focus on essential actions that enhance modular performance, leaving other actions to be managed by the Dialogue Policy. For both SaF1 and AF1 metrics, the outcome for each turn is:

$$outcome = \begin{cases} 1, & \text{if } \textbf{\textit{predict\_actions}} == \textbf{\textit{label\_actions}} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

B. Results

We use the Flan-T5 series [39] with enabling to effectively follow instructions. This model is distinguished with the previous T5 versions because of fine-tuning instruction. Our experimented datasets are SGD [37] and SGD-X [40] The model was fine-tuned on the three sub-tasks mentioned earlier as part of the in-depth fine-tuning (IFST) process, utilizing the Hugging Face[1] Trainer API with a learning rate of 5e-4, a batch size of 2048, and a maximum of 3000 steps.

As Table I, although the IFST results are lower than those of the state-of-the-art on Seen SaF1 metrics, only achieving 85.9% accuracy compared to ANYTOD [30] with 89.8% scores, we have a comparable Unseen F1 ones, which indicating that our model demonstrated ability to handle these out-of-scope cases. Though our primary contribution lies in the TOD system, our state-tracking performance does not match that of previous studies. One possible explanation for this could be the limitations of model size, which may not be sufficient to exhibit emergent abilities [41] [42].

Randomizing method for schema tends to reduce slot-filling performance slightly. As Tables I and II indicate incorporating more explicit tags may enhance the model's

[1]https://huggingface.co/

TABLE I: UUAF1, SaF1 and AF1 evaluated on the SGD test set. The outcomes of AT are derived from [30]. Announcement SaF1, as measured in IFST, monitors system queries and informs actions only.

Model	All UUAF1	All AF1	Seen SaF1	Unseen SaF1
AT T5 1.1 Base	-	-	89.8	86.1
AT T5 1.1 XXL	-	-	91.3	88.9
$IFST_{Xtags\_Xrand}$ Base	85.7	66.3	85.9	82.1
- $Xtags$	85.4	65.2	82.3	79.6
- $Xtags\_Xrand$	85.6	65.2	81.9	79.5
$IFST_{Xtags\_Xrand}$ Large	93.2	81.2	89.4	88.2
- $Xtags$	85.7	66.3	85.9	82.1
- $Xtags\_Xrand$	85.5	65.7	86.2	82.2

TABLE II: JGA on the SGD test dataset. The outcomes of AT and SDT-seq are derived from references [30] [22].

Model	Seen JGA	Unseen JGA
SDT-seq T5 1.1 XXL	95.8	86.4
AT T5 1.1 Base	89.9	62.4
AT T5 1.1 XXL	94.8	82.2
$IFST_{Xtags\_Xrand}$ Base	77.9	61.1
- $Xtags$	78.4	63.9
- $Xtags\_Xrand$	77.2	60.7
$IFST_{Xtags\_Xrand}$ Large	85.4	72.2
- $Xtags$	86.6	75.0
- $Xtags\_Xrand$	85.1	72.3

ability to predict subsequent actions, albeit with a slight trade-off in slot-filling performance. This outcome is expected, as these metrics focus solely on the query and inform actions. Overall, our results still lag behind those of the AnyTOD model, primarily because our model size is not sufficiently large and is challenged by the complexity of the knowledge schema. Another possibility is a disconnect between slots and their corresponding values due to these infrequently random tokens.

V. Conclusions

We've developed a Task-Oriented Dialogue (TOD) system framework that can work seamlessly with retrieval-augmented systems. Our approach uses symbolic methods, which offer comparable performance with conventional dialogue systems. The Interoperable Fine-tuned State Tracking (IFST) system we've created can handle users' out-of-scope activities while maintaining performance levels. However, our study has some contraints due to some reasons. First, our evaluation was conducted with only SGD dataset, which may not fully capture the complexities of all real-world task-oriented dialogue systems. Second, we lack the resources to experiment with larger models like Llama, GPT-3, and Mistral for further evaluation. We will aim to assess our system in conjunction with open-domain contexts and provide a more comprehensive study using these large language models.

References

[1] Hongru Wang, Lingzhi Wang, Yiming Du, Liang Chen, Jingyan Zhou, Yufei Wang, and Kam-Fai Wong. A survey of the evolution of language model-based dialogue systems, 2023.

[2] Chien-Sheng Wu, Steven C.H. Hoi, Richard Socher, and Caiming Xiong. TOD-BERT: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929, Online, November 2020. Association for Computational Linguistics.

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

[4] Mourad Jbene, Smail Tigani, Abdellah Chehri, Hasna Chaibi, and Rachid Saadane. Tracking dialog states in goal-oriented dialogues using a bert-based siamese network. *Procedia Comput. Sci.*, 225(C):80–87, mar 2024.

[5] Miao Li, Haoqi Xiong, and Yunbo Cao. The sppd system for schema guided dialogue state tracking challenge, 2020.

[6] Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiying Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. An end-to-end dialogue state tracking system with machine reading comprehension and wide deep classification, 2020.

[7] Yu-Ping Ruan, Zhen-Hua Ling, Jia-Chen Gu, and Quan Liu. Fine-tuning bert for schema-guided zero-shot dialogue state tracking, 2020.

[8] Yang Zhang, Vahid Noroozi, Evelina Bakhturina, and Boris Ginsburg. Sgd-qa: Fast schema-guided dialogue state tracking for unseen services, 2021.

[9] Vahid Noroozi, Yang Zhang, Evelina Bakhturina, and Tomasz Kornuta. A fast and robust bert-based dialogue state tracker for schema-guided dialogue dataset. *CoRR*, abs/2008.12335, 2020.

[10] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[11] Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. A simple language model for task-oriented dialogue, 2022.

[12] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 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, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[13] Wanwei He, Yinpei Dai, Min Yang, Jian Sun, Fei Huang, Luo Si, and Yongbin Li. Space-3: Unified dialog model pre-training for task-oriented dialog understanding and generation, 2022.

[14] 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. GALAXY: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. *CoRR*, abs/2111.14592, 2021.

[15] Adib Mosharrof, M. H. Maqbool, and A. B. Siddique. Zero-shot generalizable end-to-end task-oriented dialog system using context summarization and domain schema, 2023.

[16] Namo Bang, Jeehyun Lee, and Myoung-Wan Koo. Task-optimized adapters for an end-to-end task-oriented dialogue system, 2023.

[17] Vojtěch Hudeček and Ondřej Dušek. Are llms all you need for task-oriented dialogue?, 2023.

[18] Xinyan Zhao, Bin He, Yasheng Wang, Yitong Li, Fei Mi, Yajiao Liu, Xin Jiang, Qun Liu, and Huanhuan Chen. Unids: A unified dialogue system for chit-chat and task-oriented dialogues, 2021.

[19] Miaoran Li, Baolin Peng, Jianfeng Gao, and Zhu Zhang. Opera: Harmonizing task-oriented dialogs and information seeking experience, 2022.

[20] Zhiyu Chen, Bing Liu, Seungwhan Moon, Chinnadhurai Sankar, Paul Crook, and William Yang Wang. Ketod: Knowledge-enriched task-oriented dialogue, 2022.

[21] Jeffrey Zhao, Raghav Gupta, Yuan Cao, Dian Yu, Mingqiu Wang, Harrison Lee, Abhinav Rastogi, Izhak Shafran, and Yonghui Wu. Description-driven task-oriented dialog modeling. *CoRR*, abs/2201.08904, 2022.

[22] Raghav Gupta, Harrison Lee, Jeffrey Zhao, Yuan Cao, Abhinav Rastogi, and Yonghui Wu. Show, don't tell: Demonstrations outperform descriptions for schema-guided task-oriented dialogue. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2022.

[23] Qingyang Wu, James Gung, Raphael Shu, and Yi Zhang. Diact-tod: Learning generalizable latent dialogue acts for controllable task-oriented dialogue systems, 2023.

[24] Yuncheng Hua, Xiangyu Xi, Zheng Jiang, Guanwei Zhang, Chaobo Sun, Guanglu Wan, and Wei Ye. Dialog-to-actions: Building task-oriented dialogue system via action-level generation, 2023.

[25] Adam Santoro, Andrew Lampinen, Kory Mathewson, Timothy Lillicrap, and David Raposo. Symbolic behaviour in artificial intelligence, 2022.

[26] Vedant Gaur and Nikunj Saunshi. Reasoning in large language models through symbolic math word problems. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5889–5903, Toronto, Canada, July 2023. Association for Computational Linguistics.

[27] Jerry Wei, Le Hou, Andrew Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, and Quoc V. Le. Symbol tuning improves in-context learning in language models, 2023.

[28] Qian Liu, Fan Zhou, Zhengbao Jiang, Longxu Dou, and Min Lin. From zero to hero: Examining the power of symbolic tasks in instruction tuning, 2023.

[29] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023.

[30] Jeffrey Zhao, Yuan Cao, Raghav Gupta, Harrison Lee, Abhinav Rastogi, Mingqiu Wang, Hagen Soltau, Izhak Shafran, and Yonghui Wu. Anytod: A programmable task-oriented dialog system, 2023.

[31] Truc Nguyen Liem, Sinh Nguyen Cao Hoai, Hung Nguyen Quoc, Tien Nguyen Van, Hieu Pham Trung, Trung Nguyen Quoc, and Vinh Truong Hoang. Gradtod - a unified dialogue state tracking model for task-oriented and open domain dialogues. In *2023 IEEE 15th International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 711–719, 2023.

[32] Kai Xu, Zhengyu Wang, Yuxuan Long, and Qiaona Zhao. Deep reinforcement learning-based dialogue policy with graph convolutional Q-network. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4555–4565, Torino, Italia, May 2024. ELRA and ICCL.

[33] Jorge A. Mendez, Alborz Geramifard, Mohammad Ghavamzadeh, and Bing Liu. Reinforcement learning of multi-domain dialog policies via action embeddings, 2022.

[34] Christian Geishauser, Carel van Niekerk, Nurul Lubis, Michael Heck, Hsien-Chin Lin, Shutong Feng, and Milica Gašić. Dynamic dialogue policy for continual reinforcement learning, 2022.

[35] Hamed Zamani, Fernando Diaz, Mostafa Dehghani, Donald Metzler, and Michael Bendersky. Retrieval-enhanced machine learning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, jul 2022.

[36] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[37] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 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, 2020.

[38] Ting Chen, Saurabh Saxena, Lala Li, David J. Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection, 2022.

[39] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

[40] Harrison Lee, Raghav Gupta, Abhinav Rastogi, Yuan Cao, Bin Zhang, and Yonghui Wu. Sgd-x: A benchmark for robust generalization in schema-guided dialogue systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10938–10946, 2022.

[41] Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. Larger language models do in-context learning differently, 2023.

[42] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022.