Dynamic Prompting: Large Language Models for Task Oriented Dialog

Jan Nehring[1], Akhil Juneja[1], Adnan Ahmad[2], Roland Roller[1] and Dietrich Klakow[3]

[1]German Research Center for Artificial Intelligence (DFKI), Alt-Moabit 91c, 10559 Berlin, Germany
[2]TU Berlin, Straße des 17. Juni 135, 10623 Berlin, Germany
[3]Saarland University, Campus, 66123 Saarbrücken, Germany

Abstract
Large Language Models show impressive results in many different applications, most notably in the context of question-answering and open dialog situations. However, it is still an open question how to use those models for task-oriented dialogs such as booking or customer information systems, and such. In this work, we propose Dynamic Prompting, an architecture for task-oriented dialog, integrating the benefits of Large Language Models and showcasing the approach on the MultiWOZ 2.2 dataset. Our architecture leads to a high task success rate, provides sensible and specific answers, and is resistant to hallucinations. Further, we show that Dynamic Prompting is able to answer questions that were not anticipated by the dialog systems designer and that it can correct several types of errors and other characteristics of the system.

Keywords
Dialog Systems, Large Language Models, Task-Oriented Dialog, Dynamic Prompting,

1. Introduction

Task-Oriented Dialog Systems (TODS) assist users in completing a task within a conversation [1], for instance, in the context of customer information and bookings (train/restaurant). In an applied setting with real users, it is important that those systems provide correct answers, tasks can be quickly solved, and lead ideally to high user satisfaction. To ensure this, TODS often provide a high level of control over its dialog management and answer behavior for system developers. Existing solutions normally either manually implement a dialog manager to control the complete interaction, or train it on large amounts of dialog interactions [2, 3, 4, 5].

In contrast, Large Language Models (LLMs) are very good at open-domain dialog and provide fluent and convincing messages in different styles. However, those answers might be misleading and even false (hallucination) [6, 7, 8]. In task-oriented dialog, the model could possibly 'break out' of the given dialog task.

Using LLMs for task-oriented dialog is still in its infancy. Madotto et al. [9] used LLMs for the whole pipeline of Natural Language Understanding, Dialog State Tracking, Dialog Policy and Natural Language Generation. Hudeček and Dusek [10] expand on this idea by evaluating the abilities of LLMs to generate complete task-oriented multi-turn dialogs. They used LLMs for NLU and DST also but, unlike our work, they used a static prompt. Other approaches to LLMs for task-oriented dialog are presented by Cao [11], Hu et al. [12], Wei et al. [13], Li et al. [14].

To address those limitations and concerns, we propose *Dynamic Prompting*, a technique to combine a traditional task-oriented dialog system pipeline with the benefits of LLMs. Showcased and tested in the context of restaurant booking, we present the advantages and limitations of our approach.

2. Experimental Setup

2.1. Dataset

For our experiments, we use the Multi-domain Wizard-of-Oz 2.2 (MultiWOZ) dataset [15, 16]. The dataset contains conversations between users and task-oriented chatbots that were collected using the Wizard-of-Oz [17] method. For our setup, we only use conversations about the restaurant domain. In the conversations, users ask chatbots about restaurants in Cambridge, England. The systems can recommend various restaurants based on attributes such as the type of food, location or price range. Also, the system can perform bookings.

Table 1 shows an example dialog from the dataset. The restaurant domain encompasses two intents (find, book) and ten slots. For each user utterance, the dataset contains annotations for intent, slots in the utterance and dialog state (detected slots over the whole dialog).

CLiC-it 2024: Tenth Italian Conference on Computational Linguistics, Dec 04 — 06, 2024, Pisa, Italy
jan.nehring@dfki.de (J. Nehring); juneja.akhil@dfki.de (A. Juneja); adnan.ahmad@tu-berlin.de (A. Ahmad); roland.roller@dfki.de (R. Roller); dietrich.klakow@lsv.uni-saarland.de (D. Klakow)

User:	Can you find me an expensive place serves panasian food?
Bot:	i haven't found any expensive panasain restaurants. should i try another price range?
User:	Are there restaurants that serve Mediterranean food?
Bot:	I have a couple. Both are in the centre. la mimosa, shiraz restaurant
User:	I am interested in La Mimosa. Please book a table for 7 at 15:15 on Wednesday. Thanks for your help!
Bot:	Booking was successful. The table will be reserved for 15 minutes. Reference number is : ER5DFE50 . Is there anything else I can help you with?
User:	No, that's it. Thank you so much. Goodbye.
Bot:	Have a great day!

Table 1
MultiWOZ 2.2 example dialog.

2.2. Dialog System Architecture

Task-Oriented Dialog Systems (TODS) consist, traditionally, of the following main components [1]: Natural Language Understanding (NLU), Dialog State Tracking (DST), Dialog Manager (DM) and Response Generation (RG). The NLU performs intent detection and slot filling on the current user utterance, whereas DST keeps track of the slots over the whole dialog. The DM selects the next action of the bot based on the results of NLU and DST. Lastly, RG constructs the response to the user.

Dynamic Prompting In the following, we introduce *dynamic prompting*, a TODS architecture, extended by the capabilities of an LLM. Figure 1 shows the architecture. We chose to use a trained model for the NLU component to handle intent recognition and entity extraction, as Hudeček and Dusek [10] highlighted the limited performance of LLMs in these tasks. For NLU, we use the RASA NLU component, powered by the DIET classifier [18], while for DST, we use a simple hashmap that stores the most recent NLU results. We trained the NLU component on user utterances only from the restaurant domain of the train split of the dataset, focusing on the find and book intents. During training, we also provided use case-specific entities, including categorical (pricerange, area, bookday, bookpeople), as well as non-categorical (food, name, booktime, address, phone, postcode, and reference).

We replace/extend the Dialog Manager and Response Generation with a **Prompt Generation** and an **LLM**. The prompt generation uses input from the DST and NLU and uses a series of rules, similar to a dialog manager. Also, it fetches data from a database and generates a prompt. It uses prompting templates that consist of three parts: 1) A general task description, 2) content from the database, NLU states in JSON format, and 3) the previous conversation in a theater script style. Table 2 shows an example prompt of our system in the restaurant domain with the detected intent 'book restaurant'. The prompt is then sent to a LLM to generate a corresponding user reply. In our experiment, we use GPT-3.5-Turbo (ChatGPT) by accessing the model via API calls, as it has demonstrated leading performance in the results presented by Hudeček and Dusek [10].

Appendix A shows the prompting templates and the rules that we developed for our prompt generator. Our prompt construction approach involved multiple iterations of 'trial and error' process on the on training set, evaluating their effectiveness based on the system's task completion and relevance to the conversation. Initially, we introduced single instructions in the prompts. However, scenarios such as having no available restaurants, multiple options, or booking a restaurant required more specific instructions. This led us to implement dynamic prompts with tailored rules for each situation.

Assist the user in booking a restaurant. Always assume the restaurant is available to confirm a successful booking. Provide a reference number when the restaurant name, bookday, bookpeople, and booktime are given. Prompt if these details are missing. Omit information about fictional bookings.

Dialog State: {"food": "mediterranean", "pricerange": "expensive", bookday": "wednesday, bookpeople": "7", booktime": "15:15"}

Conversation History

User: Can you find me an expensive place serves panasian food?

Bot: i haven't found any expensive panasian restaurants. should i try another price range?

User: Are there restaurants that serve Mediterranean food?

Bot: I have a couple. Both are in the centre. la mimosa, shiraz restaurant

User: I am interested in La Mimosa. Please book a table for 7 at 15:15 on Wednesday. Thanks for your help!

Table 2
Example prompt of Dynamic Prompting, which is sent to ChatGPT.

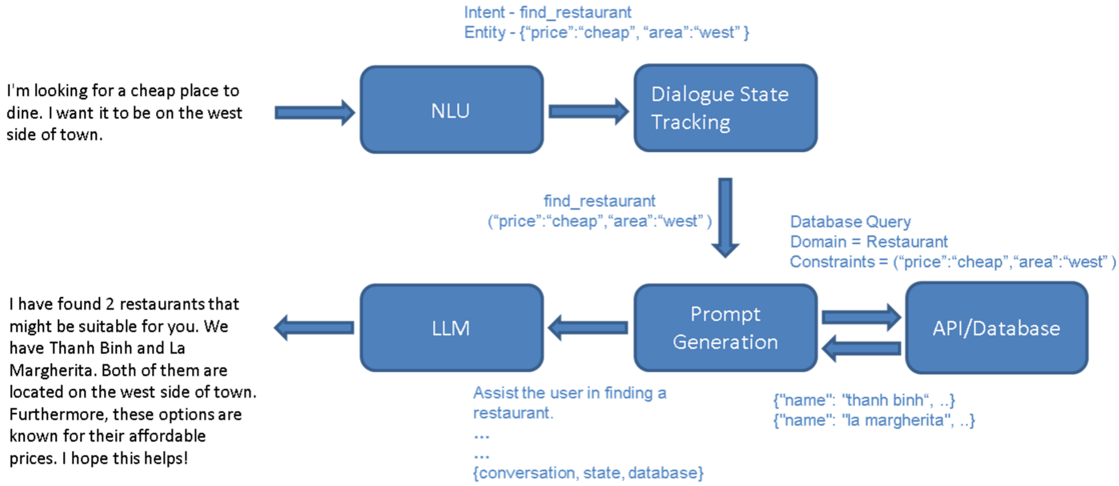

Figure 1: Processing pipeline of Dynamic Prompting

2.3. Evaluation

In our experiment, the chatbot generates a response using our dynamic prompting system for each dialog turn. We evaluated its performance on the test split of the dataset's restaurant domain. To evaluate the responses on different levels, we label them by two human annotators, given the following criteria. The annotation guidelines can be found in the supplementary materials.

- **Task Success Rate** describes the ratio of successful dialogs to the total number of dialogs. Following the definition of Wen et al. [19] and Nekvinda and Dušek [20], we mark a dialog as successful if 1) the system's recommendation aligns with the user's requests (such as price range, location, and cuisine) and 2) the system adequately addressed subsequent requests by the user, such as providing the telephone number or confirming a successful booking.
- **Prompt instruction performance**, a binary scale to assess whether responses aligned with the prompt instructions.
- **Information Extraction Performance**, a binary scale, if the system is able to fetch the relevant address from the JSON information.
- **Response slot accuracy**, the ratio of correctly predicted slot values and the number of slot values in the response. It measures if our system is able to return all desired slots to the user. We compute ratios across all annotated turns from these metrics.
- **Sensibleness** describes if the utterance makes sense given the context [21, 22].
- **Specificity** describes if the utterance is specific regarding the context [21, 22]. LMs are used to generate unspecific answers such as "this is great", which are sensible but not desired.
- **Interestingness** describes if the utterance captures someone's attention, arouses curiosity or exhibits traits such as unexpectedness, wit, or insightfulness [22]. Interestingness contributes to a compelling and engaging user experience.

3. Results

Table 3 shows the task success rate of our system compared to other TODS on the MultiWOZ 2.2 dataset. Although the other systems use the whole dataset and, thus, are not perfectly comparable to ours, it still shows that Dynamic Prompting has a similar performance compared to SOTA systems. This is remarkable, particularly as we use a relatively simple NLU component, which by itself might produce errors. However, if we do not use the NLU system of our pipeline but instead use the entity annotations from the dataset, we get a 'perfect' NLU without any errors. In this case, our Dynamic Prompting achieves a task Success Rate of 0.94 - which highlights the efficiency of the LLM solution.

Table 5 shows further performance metrics. The dialog success rate is supported by the high sensibility and specificity scores, which indicate that the system answers on point and does not deviate from the dialog's goal. However, the response slot accuracy is only 80% and needs to be improved - but this is not the focus of this work. Extracting information from the database works almost

System	Task Success
Yang et al. [23]	0.83
Lee [3]	0.80
Su et al. [24]	0.85
Dynamic Prompting	0.81
perfect NLU + Dynamic Prompting	0.94

Table 3
Comparison of Task Success Rates on MultiWOZ 2.2 data, with an inter-annotator agreement of 1 for Dynamic Prompting.

perfectly (Information Extraction Performance=0.98). Although the system does not always follow all instructions from the prompt (Prompt Instruction Performance=0.82), the task success is still quite high, so we assume that only minor errors cause the relatively low Prompt Instruction Performance.

3.1. Qualitative Analysis

In the following, we analyze the conversations and, particularly, the generated responses of our Dynamic Prompting in more detail.

3.1.1. Handling Unusual Requests

In one situation the user asked to send the information via email, which the designers of the original dataset did not anticipate. In those situations, traditional dialog systems then can only answer with “I did not understand”. Our approach instead was able to produce a sensible response, although it has never been trained for this case (see Table 4).

3.1.2. Politeness and Engagement

Similar to our findings in Section 3.1.4, the responses of our system are not only longer but also more engaging compared to the ground truth. For example, in one situation, our system produced an answer such as “You’re welcome! If you have any more questions or need further assistance, feel free to ask. Have a great day too!” while the crowd worker wrote only “Thank you. Goodbye”. Overall, we counted ‘polite’ phrases in the responses and found out that dynamic prompting uses them more often than the ground truth, such as “enjoy your meal” (15.5 more often), “have a great day” (2.2), “you’re welcome” (4.8), “certainly!” (61.0), “great!” (20.0). Table 9 in the appendix shows more detailed examples.

3.1.3. Formatting Addresses and Names

The database entries are formulated in a different format. Names are often lowercase, and the crowd workers did not correct this issue when they wrote the system responses. Also, postcodes are stored in the format “cb17aa” in the database, although the correct format would be “CB1 7AA” in the Cambridge area. Our approach consistently fixes these errors out of the box.

3.1.4. Diverse Responses

Dynamic Prompting produces responses that are, on average, 2.41 times longer and more diverse than the responses of the crowd workers in the WOZ dataset, with lexical diversity measured by an MTLD score [25] of 80.41 compared to 72.26 for the WOZ dataset. We assume that the crowd workers were interested in providing fast and minimalistic answers. However, while diverse replies might be considered as positive as they make the interaction more enjoyable, it needs to be determined if long answers are preferable over short answers containing all necessary information. Table 7 in the appendix shows examples of this behavior.

3.1.5. Adaptive NLU Error Recovery

We detected a user dialog in which the NLU failed to detect the correct area. The user asked for a restaurant in the center of the city, but due to the NLU error, the prompt generator wrote two restaurants in the prompt, one in the center and one in the south. The LLM was able to detect that the user wanted a restaurant in the center and presented only the correct restaurant in its response (see Table 8 in the Appendix).

3.1.6. Error Analysis

Our system produced two types of problems. First, it was not able to display the total count of available restaurants correctly. Although the total count of available restaurants is encoded in the JSON prompt, the LLM opted to count the number of elements within the JSON options array instead.

The second common problem was the generation of fake booking reference numbers. At the end of the dialog, users often ask for a booking reference number. We do not include booking reference numbers in the prompt. Therefore, the system hallucinated booking reference numbers. We evaluated including reference numbers in the JSON format in the database section of the prompt on ten samples, and we eliminated all occurrences of fake reference number generation in our small sample set. Still, we did not include this in the whole dataset, so our results serve as an indicator only that this approach could solve the problem of hypothetical booking reference number generation.

Prompt:	Assist the user in finding a restaurant. A single restaurant matches the user's criteria. Provide requested details of the restaurant. If no specific request is made, offer general information like food type, area, and price range. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information. State: restaurant {"area": "centre", "food": "italian", "name": "stazione restaurant and coffee bar", "bookday": "saturday", "bookpeople": "1", "booktime": "12:45"} Database: restaurant {Count:1, Options: {"address": "Market Hill City Centre", "area": "centre", "food": "italian", "id": "19240", "name": "stazione restaurant and coffee bar", "phone": "01223352607", "postcode": "cb23nj", "pricerange": "expensive", "type": "restaurant"}}
User:	Terrific! You are great, please send me all this information to my e-mail, thanks.
TOD:	I apologize, but as a language model AI, I don't have the capability to send emails. However, you can easily save or take a screenshot of this conversation for future reference. Remember to have a great time at stazione restaurant and coffee bar on Saturday at 12:45!
Truth:	You're welcome, was there anything else you need?

Table 4
Showcasing the ability of the system to handle unusual requests.

Metric	Score	IA
NLU		
Intent Detection Accuracy	0.89	na
Entity Recognition Joint State Accuracy	0.76	na
LLM metrics		
Prompt Instruction Performance	0.82	1
Information Extraction Performance	0.98	0.65
Response Slot Accuracy	0.80	na
Sensibility	0.94	1
Specificity	0.94	1
Interestingness	0.89	0.84

Table 5
The table shows the scores and the interannotator agreement (IA, Cohen Kappa) of the quantitative analysis.

4. Conclusion

We presented Dynamic Prompting, a technique integrating LLMs for task-oriented dialog. The results show high sensibility and specificity values, which indicate that the system answers on point and does not deviate from the dialog's goal. The relatively low Prompt Extraction Performance and Response Slot Accuracy values still result in excellent task success. The high values in the performance metrics Prompt Instruction Performance and Information Extraction Performance indicate that the LLM follows the task-oriented guidance of the dynamic prompts. The Information Extraction Performance of 0.98 shows that the system could very well reuse the database information embedded in the prompt in the JSON format.

In addition, our system shows various ways to correct errors, such as NLU errors, user requests not anticipated by the designer of DS, and errors in the format of the database entries. Moreover, the generated system answers are more diverse (Section 3.1.4) and more polite (Section 3.1.2) than the human-generated responses in the dataset. We would like to examine these qualitative results in future research in a more quantitative way.

Overall, we find that the widespread problem of hallucinations in LLMs is not an issue in our system as long as we present the correct information to the LLM. As soon as the user asks the system for information that is not present in the prompt, such as the booking reference numbers, the LLM starts to hallucinate.

Although we assess the system's performance solely on the restaurant domain, the dynamic prompting method can be extended to other domains in the Multi-WOZ 2.2 dataset, such as hotel, taxi, and train. Expanding to new domains will require updating the prompt generation module to accommodate new intents and state values, ensuring smooth integration with these additional domains.

Acknowledgements

This work has been supported by the Federal Joint Committee of Germany (Gemeinsamer Bundesausschuss) as part of the project smartNTX (01NVF21116).

References

[1] D. Jurafsky, J. H. Martin, Speech and Language Processing (Third Edition draft), https://web.stanford.edu/~jurafsky/slp3/ed3bookfeb3_2024.pdf, 2024. Accessed: 2024-3-10.

[2] W. He, Y. Dai, Y. Zheng, Y. Wu, Z. Cao, D. Liu, P. Jiang, M. Yang, F. Huang, L. Si, et al., Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit

policy injection, Proceedings of the AAAI Conference on Artificial Intelligence (2022).
[3] Y. Lee, Improving end-to-end task-oriented dialog system with a simple auxiliary task, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2021, Association for Computational Linguistics, Punta Cana, Dominican Republic, 2021, pp. 1296–1303. URL: https://aclanthology.org/2021.findings-emnlp.112. doi:10.18653/v1/2021.findings-emnlp.112.
[4] H. Sun, J. Bao, Y. Wu, X. He, Mars: Modeling context & state representations with contrastive learning for end-to-end task-oriented dialog, in: A. Rogers, J. Boyd-Graber, N. Okazaki (Eds.), Findings of the Association for Computational Linguistics: ACL 2023, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 11139–11160. URL: https://aclanthology.org/2023.findings-acl.708. doi:10.18653/v1/2023.findings-acl.708.
[5] Q. Wu, D. Alnuhait, D. Chen, Z. Yu, Using textual interface to align external knowledge for end-to-end task-oriented dialogue systems, 2023. arXiv:2305.13710.
[6] W. Sun, Z. Shi, S. Gao, P. Ren, M. de Rijke, Z. Ren, Contrastive learning reduces hallucination in conversations, in: Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence, AAAI'23/IAAI'23/EAAI'23, AAAI Press, 2023. URL: https://doi.org/10.1609/aaai.v37i11.26596. doi:10.1609/aaai.v37i11.26596.
[7] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, Q. V. Do, Y. Xu, P. Fung, A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity, in: J. C. Park, Y. Arase, B. Hu, W. Lu, D. Wijaya, A. Purwarianti, A. A. Krisnadhi (Eds.), Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Nusa Dua, Bali, 2023, pp. 675–718. URL: https://aclanthology.org/2023.ijcnlp-main.45. doi:10.18653/v1/2023.ijcnlp-main.45.
[8] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, P. Fung, Survey of hallucination in natural language generation, ACM Comput. Surv. 55 (2023). URL: https://doi.org/10.1145/3571730. doi:10.1145/3571730.
[9] A. Madotto, Z. Liu, Z. Lin, P. Fung, Language models as few-shot learner for task-oriented dialogue systems, 2020. arXiv:2008.06239.
[10] V. Hudeček, O. Dusek, Are large language models all you need for task-oriented dialogue?, in: S. Stoyanchev, S. Joty, D. Schlangen, O. Dusek, C. Kennington, M. Alikhani (Eds.), Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue, Association for Computational Linguistics, Prague, Czechia, 2023, pp. 216–228. URL: https://aclanthology.org/2023.sigdial-1.21. doi:10.18653/v1/2023.sigdial-1.21.
[11] L. Cao, Diaggpt: An llm-based and multi-agent dialogue system with automatic topic management for flexible task-oriented dialogue, 2024. arXiv:2308.08043.
[12] Z. Hu, Y. Feng, Y. Deng, Z. Li, S.-K. Ng, A. T. Luu, B. Hooi, Enhancing large language model induced task-oriented dialogue systems through look-forward motivated goals, 2023. arXiv:2309.08949.
[13] J. Wei, S. Kim, H. Jung, Y.-H. Kim, Leveraging large language models to power chatbots for collecting user self-reported data, 2023. arXiv:2301.05843.
[14] Z. Li, B. Peng, P. He, M. Galley, J. Gao, X. Yan, Guiding large language models via directional stimulus prompting, in: A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, S. Levine (Eds.), Advances in Neural Information Processing Systems, volume 36, Curran Associates, Inc., 2023, pp. 62630–62656. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/c5601d99ed028448f29d1dae2e4a926d-Paper-Conference.pdf.
[15] P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, M. Gašić, MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling, in: E. Riloff, D. Chiang, J. Hockenmaier, J. Tsujii (Eds.), Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 5016–5026. URL: https://aclanthology.org/D18-1547. doi:10.18653/v1/D18-1547.
[16] X. Zang, A. Rastogi, S. Sunkara, R. Gupta, J. Zhang, J. Chen, 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, Association for Computational Linguistics, Online, 2020, pp. 109–117. URL: https://aclanthology.org/2020.nlp4convai-1.13. doi:10.18653/v1/2020.nlp4convai-1.13.
[17] J. F. Kelley, An iterative design methodology for user-friendly natural language office information applications, ACM Trans. Inf. Syst. 2 (1984) 26–41.

URL: https://doi.org/10.1145/357417.357420. doi:10.1145/357417.357420.
[18] T. Bunk, D. Varshneya, V. Vlasov, A. Nichol, DIET: Lightweight language understanding for dialogue systems, 2020. arXiv:2004.09936.
[19] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gašić, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, S. Young, A network-based end-to-end trainable task-oriented dialogue system, in: M. Lapata, P. Blunsom, A. Koller (Eds.), Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, Association for Computational Linguistics, Valencia, Spain, 2017, pp. 438–449. URL: https://aclanthology.org/E17-1042.
[20] T. Nekvinda, O. Dušek, Shades of BLEU, flavours of success: The case of MultiWOZ, in: A. Bosselut, E. Durmus, V. P. Gangal, S. Gehrmann, Y. Jernite, L. Perez-Beltrachini, S. Shaikh, W. Xu (Eds.), Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021), Association for Computational Linguistics, Online, 2021, pp. 34–46. URL: https://aclanthology.org/2021.gem-1.4. doi:10.18653/v1/2021.gem-1.4.
[21] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu, Q. V. Le, Towards a human-like open-domain chatbot, 2020. arXiv:2001.09977.
[22] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, et al., LaMDA: Language Models for Dialog Applications, arXiv preprint arXiv:2201.08239 (2022).
[23] Y. Yang, Y. Li, X. Quan, Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2, in: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021, AAAI Press, 2021, pp. 14230–14238. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17674.
[24] Y. Su, L. Shu, E. Mansimov, A. Gupta, D. Cai, Y.-A. Lai, Y. Zhang, Multi-task pre-training for plug-and-play task-oriented dialogue system, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 4661–4676. URL: https://aclanthology.org/2022.acl-long.319. doi:10.18653/v1/2022.acl-long.319.
[25] P. Mccarthy, S. Jarvis, Mtld, vocd-d, and hd-d: A validation study of sophisticated approaches to lexical diversity assessment, Behavior research methods 42 (2010) 381–92. doi:10.3758/BRM.42.2.381.

Appendix

A. Prompting templates

Table 6 shows some prompting templates that have been developed for our system.

B. Example Conversations

Tables 7 - 9 show different examples conversations.

intent == find_restaurant:

len(database) $\geq$ 5:

Assist the user in finding a restaurant. Numerous restaurants match the user's criteria. Ask for missing details of food type, area, or price range to narrow down the search before providing the restaurant options. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information.
{State + Database + Conversation History}

len(database) $\geq$ 1:

Assist the user in finding a restaurant. Multiple restaurants match the user's criteria. Provide requested details from the available restaurant options. If no specific request is made, offer general information like food type, area, and price range. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information.
{State + Database + Conversation History}

len(database) == 1:

Assist the user in finding a restaurant. A single restaurant matches the user's criteria. Provide requested details of the restaurant. If no specific request is made, offer general information like food type, area, and price range. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information.
{State + Database + Conversation History}

len(database) == 0:

Assist the user in finding a restaurant. No restaurant matches the user's criteria. Suggest trying different food types, areas, or prices. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information.
{State + Database + Conversation History}

intent == book_restaurant:

Assist the user in booking a restaurant. Always assume the restaurant is available to confirm a successful booking. Provide a reference number when the restaurant name, bookday, bookpeople, and booktime are given. Prompt if these details are missing. Omit information about fictional bookings.
{State + Conversation history}

intent == none:

You're assisting the user in finding and booking a restaurant. Provide helpful information and guidance based on their requests.
{Conversation history}

Table 6
Prompting templates that we developed for our system.

Example 1:	
Prompt:	Assist the user in finding a restaurant. No restaurant matches the user's criteria. Suggest trying different food types, areas, or prices. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information. State: restaurant {"food": "mexican", "area": "centre"} Database: restaurant {Count:0}
User:	I am looking for a mexican restaurant in the centre.
TOD:	I'm sorry, but I couldn't find any mexican restaurants in the centre. However, there may be other options available. Would you like me to suggest some other types of cuisine or different areas within the centre?
Truth:	I'm sorry, there aren't any Mexican restaurants in the centre of town. Would you like to try a different cuisine?
Example 2:	
Prompt:	Assist the user in finding a restaurant. No restaurant matches the user's criteria. Suggest trying different food types, areas, or prices. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information. State: restaurant {"food": "venetian", "area": "centre"} Database: restaurant {Count:0}
User:	I am looking at a place to eat that serves venetian food in the centre.
TOD:	I apologize, but it seems that there are no restaurants in our database that serve Venetian food in the centre at the moment. However, I can suggest a few alternatives for you. You might consider trying some other types of cuisine such as Italian, seafood, or Mediterranean, which are commonly found in the centre. Alternatively, you could try searching for Venetian restaurants in nearby areas or expanding your search to include restaurants that are not strictly in the centre.
Truth:	Unfortunately, I don't see any restaurants that serve Venetian in the centre of town. Would you like to try a different area or type of cuisine?

Table 7

Showcasing the ability of the system to generate diverse responses for a similar scenario.

Prompt:	Assist the user in finding a restaurant. Numerous restaurants match the user's criteria. Ask for missing details of food type, area, or price range to narrow down the search before providing the restaurant options. Respond in a coherent, non-numbered, and fluent text format based only on the conversation history, state, and database information. State: restaurant {"food": "modern european", "pricerange": "moderate"} Database: restaurant {Count:5, Options: {"address": "152 - 154 Hills Road", "area": "south", "food": "modern european", "id": "14731", "name": "restaurant alimentum", "phone": "01223413000", "postcode": "cb28pb", "pricerange": "moderate", "signature": "slowroast sirloin of beef red onion celeriac and garlic", "type": "restaurant"}, {"address": "83 Regent Street", "area": "centre", "food": "modern european", "id": "3697", "name": "de luca cucina and bar", "phone": "01223356666", "postcode": "cb21aw", "pricerange": "moderate", "signature": "roasted barbary duck breast served with sweet potato wedges and mange tout with a red wine sauce", "type": "restaurant"}
User:	Doesn't really matter. Is there one in the centre?
TOD:	Yes, there is a modern European restaurant called "De Luca Cucina and Bar" located in the centre at 83 Regent Street. They offer a variety of dishes including their signature dish of roasted Barbary duck breast served with sweet potato wedges and mange tout with a red wine sauce. The price range is moderate. Would you like more information about this restaurant?
Truth:	Yes, there are 2: De Luca Cucina and Bar & Riverside Brasserie. Would you like further information?

Table 8

Showcasing the ability of system to adapt to NLU error.

Example 1:	
User:	Are there restaurants that serve Mediterranean food?
TOD:	Yes, there are restaurants that serve Mediterranean food. The options available are “La Mimosa" and “Shiraz Restaurant." Both of them are located in the center area of town. “La Mimosa" is situated on Thompsons Lane in Fen Ditton, and you can reach them at 01223362525. On the other hand, “Shiraz Restaurant" is located at 84 Regent Street in the City Centre, with a contact number of 01223307581. Both restaurants offer Mediterranean cuisine and fall under the expensive price range. Let me know if you need any additional information!
Truth:	I have a couple. Both are in the centre. la mimosa, shiraz restaurant.
Example 2:	
User:	Hi, I’m looking for a restaurant with modern european food. It should be in the moderate price range.
TOD:	Sure, I can help you with that. Can you please provide me with any specific area you prefer or is it fine if I search for restaurants in any area?
Truth:	What area are you wanting to be in?
Example 3:	
User:	Thank you so much. That is all I need for now. Have a great day.
TOD:	You’re welcome! If you have any more questions or need further assistance, feel free to ask. Have a great day too!
Truth:	Thank you. Goodbye.

Table 9
Showcasing the ability of system to generate polite and engaging responses.