ChatGPT for Zero-shot Dialogue State Tracking: A Solution or an Opportunity?

Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic,
Shutong Feng, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, Milica Gašić
Heinrich Heine University Düsseldorf, Germany
`{heckmi,lubis,ruppik,revuk100,fengs,geishaus,linh,niekerk,gasic}@hhu.de`

Abstract

Recent research on dialogue state tracking (DST) focuses on methods that allow few- and zero-shot transfer to new domains or schemas. However, performance gains heavily depend on aggressive data augmentation and fine-tuning of ever larger language model based architectures. In contrast, general purpose language models, trained on large amounts of diverse data, hold the promise of solving any kind of task without task-specific training. We present preliminary experimental results on the ChatGPT research preview, showing that ChatGPT achieves state-of-the-art performance in zero-shot DST. Despite our findings, we argue that properties inherent to general purpose models limit their ability to replace specialized systems. We further theorize that the in-context learning capabilities of such models will likely become powerful tools to support the development of dedicated and dynamic dialogue state trackers.

1 Introduction

Dialogue state tracking (DST) is a critical component for task-oriented dialogue systems. Its purpose is to extract and track user's goals throughout a conversation (Young et al., 2010). DST is challenging due to the infinite possibilities of user/agent conversations, and because services and schemas/APIs that dialogue systems interface are subject to constant change (Ren et al., 2018). Although traditional approaches achieve high accuracy when operating on a pre-defined set of concepts called an ontology (Mrkšić et al., 2017; Liu and Lane, 2017; Zhong et al., 2018), ongoing research explores transfer to new domains with little to no additional learning (Rastogi et al., 2020) using ontology independent architectures to allow seamless adaptation to out-of-ontology concepts.

Many strategies for zero-shot transfer to unseen domains have been proposed. Li et al. (2021) treat DST as a question answering (QA) task by leveraging data augmentation. Zhao et al. (2022) propose DST by relying on schema descriptions while Heck et al. (2022) utilize natural language descriptions to facilitate zero-shot transfer. Gao et al. (2020) and Lin et al. (2021) suggest learning from non-dialogue QA data which are available in large amounts to improve generalization. Campagna et al. (2020) harness large synthesized data based on abstract dialogue models. However, none of these techniques are ideal solutions. Fine-tuning is challenging due to computational costs, risk of over-fitting and the need for expensive (Budzianowski et al., 2018) task-specific data. Cross-task transfer still requires curated data and careful consideration of suitable learning tasks. Data augmentation requires high level task knowledge and an adequate synthesizing strategy.

A new generation of large language models (LLMs) (Brown et al., 2020; Ouyang et al., 2022; Glaese et al., 2022) comes with the promise to be equipped to solve any task without task-specific fine-tuning, but solely with world knowledge they acquired during self-training on massive amounts of data. Such LLMs have been shown to perform remarkably well on in-context learning (ICL), where only a natural language prompt and examples are provided to condition the generation process, achieving significant improvements over fine-tuned approaches in few-shot setups (Brown et al., 2020; Wang et al., 2022). ChatGPT (OpenAI, 2022) – trained using human feedback and reinforcement learning – is the most recent of such models and single-handedly solves an array of challenging natural language processing (NLP) tasks with super-human capabilities, all through a natural language dialogue interface.

In this work, we aim to answer the question: *does ChatGPT solve the problem of zero-shot DST?* We show that crafting intuitive natural language prompts is sufficient to achieve state-of-the-art performance with ChatGPT, exceeding conventional, engineering-heavy approaches to zero-shot DST

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 936–950
July 9-14, 2023 ©2023 Association for Computational Linguistics

by a large margin. However, despite our findings, we argue that properties inherent to general purpose models inhibit their ability to simply replace specialized systems. We speculate that while in the foreseeable future general purpose models may not become holistic solutions to complex problems, they will provide ample opportunities to empower specialized systems to go beyond their pre-defined scopes, enable on-the-fly extensibility and generation of high quality training data by zero-shot synthesizing or automatic labeling.

2 Background

Dialogue state tracking is tasked to (1) determine for every turn t in a dialogue $\{(U_t, M_t)\}_1^T$ with U_t and M_t being current user and preceding system utterance whether any of the slots in $S = \{S_n\}_1^N$ is present, to (2) predict values for each S_n and to (3) track the dialogue state DS_t $\forall t \in [1, T]$. The DS is cumulative, i.e., $DS_t = \text{update}(DS_{t-1}, \widehat{DS}_t)$ is updated given the predictions of slot-value updates $\widehat{DS}_t$.

ChatGPT (OpenAI, 2022) is a dialogue agent (Leike et al., 2018), and in its core a GPT-3.5 LLM fine-tuned on human-written prompt-response pairs followed by reinforcement learning with human feedback (RLHF) (Christiano et al., 2017; Stiennon et al., 2020). RLHF utilizes a reward model trained on human feedback to improve generation quality and adequacy via proximal policy optimization (Schulman et al., 2017), thereby *aligning* model output to human values and user's expectations. At the time of writing this work, ChatGPT is proprietary. As a sibling model to InstructGPT, details of its training are elaborated by Ouyang et al. (2022).

3 Zero-shot DST with ChatGPT

Our investigative approach to zero-shot DST with ChatGPT differs considerably from related works. We decode dialogue state updates with a general purpose model, without undergoing any parameter updates. Consequently, we neither employ data augmentation nor cross-task transfer learning. Instead, we solely rely on the general capacities of ChatGPT as an aligned dialogue agent. We take a most rigorous approach to zero-shot transfer where we do not allow the provision of any examples, nor of a formal task definition. Instead, we only permit natural language explanations of what the model is supposed to do. This sets our investigation apart from the closely related IC-DST (Hu et al., 2022).

In zero-shot DST, the set of slots S relevant during inference and the set of slots S' seen during training of the model X_θ with parameters θ are disjoint, i.e., $S \cap S' = \emptyset$. Further, it may be $S' = \emptyset$, in which case θ is not specifically tuned towards solving DST. This is precisely the case for ChatGPT in our setup. Our approach to zero-shot DST with ChatGPT is formalized as follows. Let

$$\begin{aligned} A_1 &= P \oplus \text{"system"}{:}M_1 \oplus \text{"user"}{:}U_1, \\ A_t &= \text{"system"}{:}M_t \oplus \text{"user"}{:}U_t, \forall t \in [2, T], \end{aligned}$$

where P is the task description which provides the model with instructions for how to process a dialogue between a system M and a user U. A_1 is the initial prompt to ChatGPT. $A_{t\geq 2}$ are the follow-up prompts, only containing a single turn-pair of the dialogue of interest. ChatGPT is particularly suitable for this strategy due to its chat based interface. ChatGPT generates its next output B_t conditioned on the current prompt A_{t-1} , as well as all preceding user queries and system responses of the same chat. The dialogue state update $\widehat{DS}_t$ can be found in B_t, but may not be directly interpretable as such due to the diversity in the output surface forms. Thus, we require a normalization operation $\widehat{DS}_t = \text{normalize}(B_t)$. In contrast to (Hu et al., 2022), we do not condition B_t on DS_t. This renders the task even more challenging, as ChatGPT is forced to solve complex subtasks such as coreference resolution – the case where a newly encountered slot refers to the value of another slot – solely given the initial prompt and its own latent dialogue state given the dialogue history.

4 Experiments

At the time of conducting our experiments, ChatGPT is a proprietary research preview accessible for free via a web interface[1]. We used the `Jan 9` version of the model. We use a regular expression term to extract all parts that are JSON formatted. We form DS_t by accumulating all predicted updates up to turn t.

Evaluation. We evaluate on the 1000 dialogues of the MultiWOZ 2.1 (Eric et al., 2020) test split and use joint goal accuracy (JGA) to compare methods. For a fair judgement of the ChatGPT predictions, we follow the evaluation procedure of Heck

---

[1] `chat.openai.com` (accessed 6. Jan. to 20. Jan. 2023)

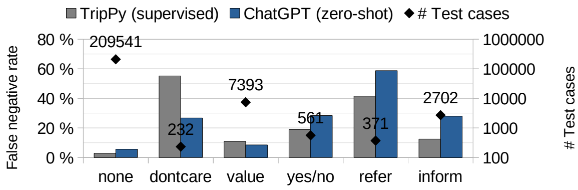


Figure 1: Missed predictions by type of value.

et al. (2020). We process each dialogue once and refrain from using ChatGPT's *regeneration* feature.

Prompt. We imposed restrictions that the task-defining prompt P be intuitive natural language and provides no formal schema. The crafting process involves simple trial-and-error on fewer than 10 held-out dialogues from the MultiWOZ training set. The design process was guided by the intention to imitate the behavior of a triple copy strategy (TripPy) DST (Heck et al., 2020). P consists of three parts. First, a list of names for detectable informable slots along with natural language descriptions. The slot names help us extract a $\widehat{DS}_t$ that is compatible with the dataset's labels. Second, a sparse list of slots that are categorical, along with their value candidates for (1) aiding normalization of values that are expected to show high variability in expression, and (2) modeling Boolean slots. Third, an informal task description.[2]

4.1 ChatGPT vs. Supervised SOTA

Comparing ChatGPT's performance to state-of-the-art *supervised* approaches that achieve close to 60% JGA is not a fair fight[3], and yet we observe an impressive 31.5% zero-shot JGA. This result is double-edged; on the one hand it is evidence that ChatGPT is capable of DST[4], and on the other hand is no match for specialized systems.

The comparison to TripPy, a SOTA supervised model, allows us a more fine-grained analysis. In Figure 1, slot filling performance is broken down into value types. We observed that ChatGPT underperforms in non-trivial cases, namely *refer*, where a newly encountered slot refers to the value of another slot, and *inform*, where a slot-value was mentioned by the system and confirmed by the user. ChatGPT shows slight underperformance for Boolean slots. Remarkably, performance for values that are extracted directly from user utterances – the most relevant category in terms of frequency – is exceeding the strong supervised baseline. Lastly, ChatGPT has a clear advantage in the underrepresented and therefore notoriously difficult *dontcare* cases, where a user is indifferent about a particular value for a slot.

Models	attr.	hotel	rest.	taxi	train	avg.
TRADE (2019; 2020)	22.8	19.5	16.4	59.2	22.9	28.16
TripPy-R (2022)	27.1	18.3	15.3	61.5	23.7	29.18
TransferQA (2021)	31.3	22.7	26.3	61.9	36.7	35.78
Li et al. (2021)	42.4	24.9	27.7	60.3	41.1	39.28
D3ST (2022)	**56.4**	21.8	38.2	**78.4**	38.7	46.70
Campagna et al. (2020)	52.8	36.3	45.3	62.6	46.7	48.74
ChatGPT	52.7	**42.0**	**55.8**	70.9	**60.8**	**56.44**
IC-DST[5] (2022)	60.0	46.7	57.3	71.4	49.4	56.96

Table 1: Performance comparison between state-of-the-art methods for zero-shot DST in per-domain JGA.

4.2 ChatGPT vs. Zero-shot SOTA

ChatGPT considerably outperforms previous approaches to zero-shot DST (see Table 1) and is more stable across domains than other methods. The model tends to handle challenging domains markedly better, while maintaining high performance on domains that are handled with relative ease by earlier approaches. Most approaches to zero-shot DST still employ supervised learning on a subset of domains and test on a held-out domain. Such methods struggle in domains with many slots never seen during training. This is evident for *hotel*, which has many unique slots and is the only domain with Boolean slots. ChatGPT can excel in such challenging scenarios by drawing from its general world knowledge to interpret concepts. *taxi* is challenging due to its frequent *refer* cases. Where most other methods fail, ChatGPT shows competency in resolving co-references in the zero-shot setting. Other models designed for DST rely on architectures that are not fundamentally different from the backbone model of ChatGPT. The reason for ChatGPT's superior abilities in conducting DST is likely found in its training scheme, particularly instruction tuning and alignment via reinforcement learning with human feedback (Ouyang et al., 2022; Ziegler et al., 2019), combined with its massive scale in terms of model and training data size. IC-DST (Hu et al., 2022) was the first successful attempt at pseudo[5] zero-shot DST via ICL. Our preliminary results with ChatGPT are on

---

[2] See Appendix A for the full prompt.

[3] `https://github.com/budzianowski/multiwoz`

[4] See Appendix B for an example dialogue.

[5] Hu et al. (2022) uses hand-crafted labeled examples for ICL even in the "zero-shot" case.

par, which is remarkable for the following reasons. (1) Our prompt is non-schematic and without examples, (2) our task-defining prompt is stated only once at the beginning of the chat, and (3) we do not maintain a DS to serve as additional input at each turn. The heightened zero-shot performance of IC-DST can be mainly attributed to these points.

4.3 Error Analysis

We identified a set of recurring errors that are likely caused by either the content of P or by the model's inherent properties. See Table 2 for examples. See also Appendix C lists for more detailed instances.

a) Failed carry-over of system-informed values. Our P does not explicitly instruct to resolve *inform* cases (see Section 4.1) cases. Nevertheless, ChatGPT handles the majority of cases correctly, failing to carry over only about 28% of system-informed values. Specifying the desired behavior in P may improve this ratio further.

b) Incomplete coreference resolution. Coreferences are usually detected – i.e., in about 65% of cases –, but often not resolved. Where a coference was detected, about 23% are not resolved correctly, and another 13% are incorrect due to other errors.

c) Overprediction of *dontcare*. The recall of ChatGPT for *dontcare* is considerably higher than for the supervised baseline, but precision is low. About 35% of *none* false negatives in Figure 1 can be attributed to overpredicting *dontcare* by ChatGPT, compared to 5% for the supervised baseline. This is likely caused by the formulation in P. Occasionally, the model interprets slots that are not specifically filled by the user as *dontcare*.

d) Ignoring value candidates. On rare occasions, ChatGPT ignores value candidates for categorical slots and picks variants from the dialogue context instead. We observed this error for 0.1% of all values to be extracted from the context.

e) Hallucinated slots. The model frequently hallucinates slots. About 90.4% of all of ChatGPT's slot predictions are MultiWOZ slots. Since we specifically prompt ChatGPT to fill slots requested by the user with "?", the vast majority of hallucinations – 8.6% of all slot predictions – are of the *requestable* type, which are not considered by the standard MultiWOZ evaluation and are therefore not listed in P. In fact, ChatGPT predicts all requestable slots appearing in the MultiWOZ dataset with an average recall of 61%. Rarely – in 0.3% of all cases – alternative names are hallucinated for slots listed in P. A further 0.6% are predictions for made up slots.

f) Arbitrary normalization. We observed that the model sometimes chooses to normalize predicted values. However, these normalizations are inconsistent across dialogues.

g) Predicting DS_t instead of $\widehat{DS_t}$. Despite explicitly requesting to predict DS updates, ChatGPT on rare occasions – in 0.2% of all processed dialogues – attempts to predict the full DS at each turn, which may cause other phenomena such as slot-value over-prediction.

5 Discussion: Solution or opportunity?[6]

ChatGPT is a sophisticated dialogue agent that, via alignment with human judgements, is capable of understanding context and intent of a multi-turn conversation far beyond the capacities of the previous generation of LLMs. This makes it well-suited for DST. Our results demonstrate that even with intuitive natural language prompts, a complex task such as DST can be solved exceedingly well without any form of additional learning.

While specialized systems can exert control over its input-processing and output-generation to arbitrary degrees, this is not the case for ChatGPT. Even with the most rigorous and schematic prompts, there can be no guarantee that the model interprets the input as intended or generates the output as required, which may lead to unexpected behavior. Furthermore, there is no guarantee that behavior is consistent across a series of similar inferences, such as in our experimental evaluation. In terms of deployment, the cost factor of building and running massive models may hinder their utilization as a plug-and-play module.

Despite impressive zero-shot and ICL results for general purpose models, specialist models still perform best on most tasks thanks to task-specific solutions via adequate engineering (Heck et al., 2020; Ye et al., 2021; Kim et al., 2020) and task-related data. However, the opportunities to improve dedicated systems with the help of general purpose models are plenty. Their predictive powers could be used for developing smaller, specialized, low inference cost models. Automatic labeling and data

---

[6] The model's own response on that matter is refreshingly balanced. See Appendix D for ChatGPT's response.

a)	PMUL4050	system: "I'd recommend the Autumn House. Would you like to make a booking?" user: "Yes please. I need the reservation to be for 8 people and 2 nights starting on Tuesday." **Prediction:** ... hotel-name: none **Label:** ..., hotel-name: autumn house
b)	PMUL0117	user: "Yes I also need a taxi that will get me to the restaurant by the booked time please." **Prediction:** taxi-destination: hotel, taxi-departure: restaurant **Label:** taxi-destination: the gonville hotel, taxi-departure: la mimosa
c)	SNG01873	user: "I need to be picked up from pizza hut city centre after 04:30" **Prediction:** ..., hotel-name: dontcare, ..., attraction-type: dontcare, ... **Label:** ...
d)	PMUL0599	user: "[...] Can you just help me find a high-end Mexican restaurant?" **Prediction:** ..., restaurant-pricerange: high-end **Label:** ..., restaurant-pricerange: expensive
e)	MUL2051	user: "Can I get address and postcode for the hotel?" **Prediction:** hotel-address: ?, hotel-postcode: ? **Label:** - system: "The address is 74 chesterton road, the postal code is cb41er, can I assist with anything else?" user: "That is all for now, goodbye." **Prediction:** hotel-address: 74 chesterton road, hotel-postcode: cb41er **Label:** -
f)	MUL0524	user: "I'm going to Cambridge on saturday and want to arrive by 14:15 please." **Prediction:** ..., train-day: Saturday **Label:** ..., train-day: saturday
g)	PMUL4246	user: "i need a place to go and should be a museum" **Prediction:** attraction-type: museum **Label:** attraction-type: museum system: "Okay! There are several museums in Cambridge. What part of town would you like to visit?" user: "How about ones in the centre, what's available?" **Prediction:** attraction-type: museum, attraction-area: centre **Label:** attraction-area: centre

Table 2: Examples of reoccurring errors as listed in Section 4.3.

augmentation are natural use cases for ChatGPT, as is evident from our experimental results; a per-domain JGA of 70% (see Section 4.2) is surely sufficient to generate additional mid- to high-quality training data for dedicated systems. Automatic labeling may be conducted on-line for on-the-fly adaptation of production systems or off-line for iterative learning.

Another way of harnessing general purpose models is the integration into dedicated systems as fallback options in case of out-of-domain or out-of-ontology requests. An integration via knowledge-seeking term detection (Gunasekara et al., 2020) could facilitate the ability to provide context-aware responses that go beyond the original scope of the specialized system. General purpose models may handle unseen domains in place of the main model.

While hallucinations may be an issue if not handled adequately, they also pose an opportunity to enable zero-shot concept detection. We observed that many slot hallucinations were sensible and pointed at elements that were meaningful to conversations. Zero-shot slot detection may be utilized to annotate and prepare unstructured data for model training, and to expand a system's capacities on-the-fly. Dialogue state trackers with dynamic dialogue states have the potential to expand a task-oriented dialogue system's conversational range seamlessly (Geishauser et al., 2022). A general purpose model that has the capacity to identify new concepts may be utilized to generate API calls and database queries that are unknown to the specialized system (OpenAI, 2023; Chase, 2023).

General purpose models may replace some components in a modular dialogue system (Zhu et al., 2022). It might still be beneficial to rely on specialized DST and a dedicated policy for particular tasks in order to maintain interpretability and a desired level of control over information flow. However, natural language understanding (NLU) and natural language generation (NLG) modules may be powered by generative large language model based systems such as ChatGPT in order to benefit from a heightened ability of semantic modeling and to facilitate more natural and diverse output, thus promoting more natural conversations with modular task-oriented dialogue systems.

6 Conclusion

This work is the first to investigate ChatGPT's capacities for zero-shot DST. Despite remarkable preliminary results that we achieved, we identified limitations rooted in inherent properties of general purpose models, preventing these to become holistic solutions to complex NLP problems without further research. We discussed opportunities provided by ChatGPT and similar models to advance the development of specialized systems. With our insights and discussion, we hope to stimulate research in similar directions.

Limitations

At the time of writing this work, ChatGPT is only available as a proprietary free research preview via a web interface. This is limiting in several ways. (1) Parts of our analysis are qualitative, as quantification is challenging due to limited accessability of the investigated model. (2) Some details about the investigated model are not yet disclosed. This is true for the model design as well as for the data used to train ChatGPT. MultiWOZ is a freely available and widely used dataset, therefore no guarantee can be given that ChatGPT has not been exposed to at least some meta details regarding this dataset. (3) Given the nature of the free research preview, exact reproducibility is not guaranteed, as the model may change any time. However, it is expected that any future version of ChatGPT retains its general abilities and behaviors.

Model-as-a-service. Building a general purpose model such as ChatGPT is extremely costly and an option only for few. However, once it exists, it may be utilized for a multitude of purposes. As a model, ChatGPT does not need to be built for DST in order to be useful for DST. With capable enough general purpose models, fine-tuning towards specific tasks may be avoided. Fine-tuning is challenging for multiple reasons such as the need for adequate data, computational costs, risk of over-fitting and catastrophic forgetting, among others.

Just like its sibling model, ChatGPT will become available as model-as-a-service. The advantage of this is that a massive LM such as this is usable independent of the user's hardware. But this advantage comes with the disadvantage that it will in all probability remain proprietary. In consequence, it will likely not be possible to ever run, adapt, train or modify ChatGPT on local machines.

ChatGPT as model-as-a-service is likely to remain a black box to customers and researchers, even if just in parts. The model may change any time. In fact, a model update during our experimental evaluation prompted us to re-process a few of our test dialogues. This property impedes backward compatibility and the ability to trust in familiar behavior.

A general purpose model may show too general behavior and converse about more than what is required or requested. This also poses vulnerabilities for adversarial attacks. To this end, models such as ChatGPT have been trained with human feedback to better handle malicious intent and abusive behaviors.

A model-as-a-service is a gated resource. As such, its indefinite availability cannot be guaranteed. Further, recurring costs for access may be too high for certain downstream tasks. As a hosted service, latency might become a bottleneck or hindrance for its use as a component in complex applications.

Ethics Statement

The disclaimer of ChatGPT states that the model may occasionally generate incorrect information and may occasionally produce harmful instructions or biased content. Models, code and datasets were used in accordance with their respective licenses, terms of use and intended use. We provide logs and code that we created for this work.[7] Data that we used and generated does not contain any information that names or uniquely identifies individual people or offensive content.

Acknowledgements

M. Heck, N. Lubis, S. Feng and C. van Niekerk are supported by funding provided by the Alexander von Humboldt Foundation in the framework of the Sofja Kovalevskaja Award endowed by the Federal Ministry of Education and Research, while C. Geishauser, H-C. Lin, B. Ruppik and R. Vukovic are supported by funds from the European Research Council (ERC) provided under the Horizon 2020 research and innovation programme (Grant agreement No. STG2018804636). We thank Girish Kulkarni and Annika Hennes for their help in processing MultiWOZ dialogues with ChatGPT.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

---

[7] `https://gitlab.cs.uni-duesseldorf.de/general/dsml/chatgpt-dst-public`

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Giovanni Campagna, Agata Foryciarz, Mehrad Moradshahi, and Monica Lam. 2020. Zero-shot transfer learning with synthesized data for multi-domain dialogue state tracking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 122–132, Online. Association for Computational Linguistics.

Harrison Chase. 2023. LangChain. Accessed 2023-05-25.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, pages 4299—-4307. Curran Associates, Inc.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tür. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428, Marseille, France. European Language Resources Association.

Shuyang Gao, Sanchit Agarwal, Di Jin, Tagyoung Chung, and Dilek Hakkani-Tur. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 79–89, Online. Association for Computational Linguistics.

Christian Geishauser, Carel van Niekerk, Hsien-chin Lin, Nurul Lubis, Michael Heck, Shutong Feng, and Milica Gašić. 2022. Dynamic dialogue policy for continual reinforcement learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 266–284, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. 2022. Improving alignment of dialogue agents via targeted human judgements.

R. Chulaka Gunasekara, Seokhwan Kim, Luis Fernando D'Haro, Abhinav Rastogi, Yun-Nung Chen, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, Dilek Hakkani-Tür, Jinchao Li, Qi Zhu, Lingxiao Luo, Lars Liden, Kaili Huang, Shahin Shayandeh, Runze Liang, Baolin Peng, Zheng Zhang, Swadheen Shukla, Minlie Huang, Jianfeng Gao, Shikib Mehri, Yulan Feng, Carla Gordon, Seyed Hossein Alavi, David R. Traum, Maxine Eskénazi, Ahmad Beirami, Eunjoon Cho, Paul A. Crook, Ankita De, Alborz Geramifard, Satwik Kottur, Seungwhan Moon, Shivani Poddar, and Rajen Subba. 2020. Overview of the ninth dialog system technology challenge: DSTC9. *CoRR*, abs/2011.06486.

Michael Heck, Nurul Lubis, Carel van Niekerk, Shutong Feng, Christian Geishauser, Hsien-Chin Lin, and Milica Gašić. 2022. Robust dialogue state tracking with weak supervision and sparse data. *Transactions of the Association for Computational Linguistics*, 10:1175–1192.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. *CoRR*, abs/2203.08568.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 567–582, Online. Association for Computational Linguistics.

Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. 2018. Scalable agent alignment via reward modeling: a research direction. *ArXiv*, abs/1811.07871.

Shuyang Li, Jin Cao, Mukund Sridhar, Henghui Zhu, Shang-Wen Li, Wael Hamza, and Julian McAuley. 2021. Zero-shot generalization in dialog state tracking through generative question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1063–1074, Online. Association for Computational Linguistics.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Zhenpeng Zhou, Paul Crook, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021. Zero-shot dialogue state tracking via cross-task transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7890–7900, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bing Liu and Ian Lane. 2017. An end-to-end trainable neural network model with belief tracking for task-oriented dialog. In *Proceedings of Interspeech 2017*, pages 2506–2510.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

OpenAI. 2022. ChatGPT: Optimizing language models for dialogue. Accessed 2023-01-13.

OpenAI. 2023. ChatGPT plugins. Accessed 2023-05-25.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Schema-guided dialogue state tracking task at DSTC8. *CoRR*, abs/2002.01359v1.

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2780–2786, Brussels, Belgium. Association for Computational Linguistics.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Hannaneh Hajishirzi, Noah A. Smith, and Daniel Khashabi. 2022. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *CoRR*, abs/2204.07705.

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

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021. Slot self-attentive dialogue state tracking. In *Proceedings of the Web Conference 2021*, pages 1598–1608.

Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

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

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467, Melbourne, Australia. Association for Computational Linguistics.

Qi Zhu, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, Baolin Peng, Zheng Zhang, Michael Heck, Nurul Lubis, Dazhen Wan, Xiaochen Zhu, Jianfeng Gao, Milica Gasic, and Minlie Huang. 2022. Convlab-3: A flexible dialogue system toolkit based on a unified data format. *CoRR*, abs/2211.17148.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593.

A Task-defining Prompt P

See Figure 2 for our complete prompt.

B Example Dialogue

See Figure 3 for an example dialogue that processes `MUL1350` for DST.

B.1 Reasoning about Numbers

From `MUL2122`:

```
"system": ""
"user": "Hello!  I am looking for train schedules from
      ↪ Leicester.  Can you help me with that?"

The following slots were updated by the user:
{
"train-departure": "Leicester",
"train-destination": "?",
"train-day": "?",
"train-leaveAt": "?",
"train-arriveBy": "?",
"train-book_people": "?"
}

"system": "Sure! There are several trains leaving from
      ↪ leicester. What destination, day and time would you
      ↪  like me to search?"
"user": "I like a train going to cambridge on tuesday please
      ↪ ."

The following slots were updated by the user:
{
"train-destination": "cambridge",
"train-day":"tuesday",
"train-leaveAt":"dontcare",
"train-arriveBy":"dontcare"
}

"system": "Okay.  It looks as if there are trains leaving
      ↪ Leicester for Cambridge every hour starting at 5:09
      ↪  and ending at 23:09 on Tuesday. "
"user": "Pick one. I need the travel time and arrival time.
      ↪ Any one after 20:15."

The following slots were updated by the user:
{
"train-leaveAt":"dontcare",
"train-arriveBy":"20:15"
}

"system": "You have one leaving at 21:09, what time do you
      ↪ want to arrive by?"
"user": "Thank you that will work fine for me and my husband
      ↪ "

The following slots were updated by the user:
{"train-book_people":"2"}

"system": "The booking was successful. Your reference number
      ↪  is RWRKP9L6"
"user": "I also need a hotel with free wifi and free parking
      ↪ . "

The following slots were updated by the user:
{
"hotel-parking":"yes",
"hotel-internet":"yes"
}

"system": "I have several excellent options. Did you have an
      ↪  area of town in mind?"
"user": "No, I just would like a guesthouse. "

The following slots were updated by the user:
{
"hotel-type":"guest house",
"hotel-area":"dontcare"
}

"system": "Sure, I'd suggest Acorn Guest House at 154
      ↪ Chesterton Road CB41DA. It's moderately priced with
      ↪  4 stars. Would that work for you?"
"user": "Can you please make a reservation for just me, for
      ↪ 4 nights starting tuesday?"

The following slots were updated by the user:
{
"hotel-name":"Acorn Guest House",
"hotel-book_day":"Tuesday",
"hotel-book_stay":"4",
"hotel-book_people":"1"
}

"system": "Your reference number is HYEDH9O5, anything else
      ↪ I can help you with?"
"user": "No thank you, you have been very helpful. Have a
      ↪ great day."

An empty JSON list, no new slot is updated by the user.
```

B.2 Explaining a Response Referring to P

From `MUL2405`:

```
"system": ""
"user": "Hi. I am having trouble finding a place to eat
      ↪ called prezzo. Can you locate it for me and tell me
      ↪  a little about it please?"
There were no "slots" updated in the "user"'s latest
      ↪ response to the "system". As per the provided user
      ↪ sentence, the user is seeking information and is
      ↪ not providing any details. Thus the response should
      ↪  be an empty JSON list {}

"system": "Prezzo is located at 21 - 24 Northampton Road, in
      ↪  the west. It's an italian restaurant. Moderate
      ↪ price range."
"user": "That sounds perfect!"
The "slots" updated by the "user" in this response are the
      ↪ following:

Copy code
{
    "restaurant-name": "prezzo",
    "restaurant-area": "west",
    "restaurant-food": "italian",
    "restaurant-pricerange": "moderate"
}
It's worth noting that these slots are updated only because
      ↪ system provided that information in the previous
      ↪ message, if the system have not provided this
      ↪ information, the updated slots should be filled by
      ↪ "?"

"system": "Would you like me to book you a table?"
"user": "Not this time but I would like to visit an
      ↪ attraction while in Centre, can you find one for me
      ↪ ?"
The "slots" updated by the "user" in this response are the
      ↪ following:

Copy code
{
    "restaurant-book_people": "dontcare",
    "restaurant-book_day": "dontcare",
    "restaurant-book_time": "dontcare",
    "attraction-area": "centre"
}
It's worth noting that restaurant-book_people,restaurant-
      ↪ book_day,restaurant-book_time are updated with "
      ↪ dontcare" since the user doesn't care about booking
      ↪  a table.

"system": "Okay, there's about 44 attractions in the centre
      ↪ of town.  Is there something in particular you'd
      ↪ like?"
"user": "no, but I will need the address."
The "slots" updated by the "user" in this response are the
      ↪ following:

Copy code
{
    "attraction-type": "?",
    "attraction-name": "?"
}
In this response the user didn't give any specific
      ↪ attraction details, so the "slots" of attraction-
      ↪ type and attraction-name are filled with "?" as the
      ↪  user didn't specify any preferences.

"system": "I think you would enjoy Old Schools in the centre
      ↪  of town.  It is an architectural attraction.  The
      ↪ address is trinity lane."
"user": "Great, that sounds perfect. Thanks!"
The "slots" updated by the "user" in this response are the
      ↪ following:

Copy code
{
    "attraction-type":"architectural",
    "attraction-name":"Old Schools",
    "attraction-address":"trinity lane"
}
The user is indicating that this attraction is the one they
      ↪ are interested in and thank the system, thus the
      ↪ slots are updated with the provided information.

"system": "Post code is cb21tt, Trinity lane. Is there
      ↪ anything else I can tell you?"
"user": "Not today , thank you so much for your help. I
      ↪ appreciate everything you do."
The "slots" updated by the user in this response are the
      ↪ following:

Copy code
{
    "attraction-postcode":"cb21tt"
}
The user is indicating the end of their conversation and
      ↪ expressing gratitude, so the system does not
      ↪ receive any new request.

"system": "Absolutely positively no problem at all! Thanks
      ↪ for using our service. Enjoy your time!"
```

```
"user": "Thanks, I'm all set here. You can end the
      ↪ conversation now."
There were no "slots" updated by the user in this response.
      ↪ The user is indicating the end of the conversation.
      ↪  Thus the response should be an empty JSON list {}
```

C Error Types

C.1 Failed carry-over of system-informed values

From `PMUL4050`:

```
"system": "I'd recommend the Autumn House. Would you like to
      ↪  make a booking?"
"user": "Yes please. I need the reservation to be for 8
      ↪ people and 2 nights starting on Tuesday. "
{
"hotel-book_people":"8",
"hotel-book_stay":"2",
"hotel-book_day":"Tuesday"
}

"system": "Great. I successfully booked your stay. Your
      ↪ reference number is 2OVWNFW1. Is there anything
      ↪ else I can help you with?"
"user": "I am looking for a particular attraction. Its name
      ↪ is called lynne strover gallery"
{
"attraction-name": "lynne strover gallery",
"attraction-type":"?"
}

"system": "The Lynne Strover Gallery is a wonderful museum
      ↪ in the West area of town, it is free to enter.
      ↪ Would you like the phone number and address perhaps
      ↪ ?"
"user": "Yes, I would also like to know what kind of
      ↪ attraction it is and the entrance fee."
{
"attraction-type": "?",
"attraction-fee":"?"
}

"system": "It is a museum and it is free entry."
"user": "What area is it in?"
{
"attraction-area":"west"
}

"system": "As stated previously, it is on the west side of
      ↪ town. Is there anything else I can do for you?"
"user": "No that's all the help I need. Thanks for your help
      ↪ . "
{}
```

C.2 Incomplete coreference resolution

From `MUL1228`:

```
"system": "I have one hotel in the moderate range, cityroomz
      ↪ . Would you like me to book it for you?"
"user": "Yes, please. Can you book a room for Friday for 1
      ↪ person, 3 nights?"
{
"hotel-name": "cityroomz",
"hotel-book_people": "1",
"hotel-book_day": "Friday",
"hotel-book_stay": "3"
}

"system": "Your reservation has been made. Reference number
      ↪ DTONYTIH. Is there anything else I can help you
      ↪ with?"
"user": "I need to book a taxi to get me to the restaurant
      ↪ by 21:30."
{
"taxi-destination": "?",
"taxi-arriveBy": "21:30",
"taxi-leaveAt": "dontcare",
"taxi-departure": "dontcare"
}

"system": "Will you be departing from your hotel?"
"user": "No, I'd like to leave the restaurant by 21:30, I
      ↪ need the taxi to take me to the hotel."
{
"taxi-destination": "hotel",
"taxi-leaveAt": "21:30",
"taxi-arriveBy": "dontcare",
"taxi-departure": "restaurant"
}
```

C.3 Overprediction of *dontcare*

From `SNG01873`:

```
"system": ""
"user": "I need to be picked up from pizza hut city centre
      ↪ after 04:30"

{
"taxi-leaveAt": "04:30",
"taxi-destination": "pizza hut city centre",
"taxi-departure": "?",
"taxi-arriveBy": "dontcare",
"restaurant-book_people": "dontcare",
"restaurant-book_day": "dontcare",
"restaurant-book_time": "dontcare",
"restaurant-food": "dontcare",
"restaurant-pricerange": "dontcare",
"restaurant-name": "dontcare",
"restaurant-area": "dontcare",
"hotel-book_people": "dontcare",
"hotel-book_day": "dontcare",
"hotel-book_stay": "dontcare",
"hotel-name": "dontcare",
"hotel-area": "dontcare",
"hotel-parking": "dontcare",
"hotel-pricerange": "dontcare",
"hotel-stars": "dontcare",
"hotel-internet": "dontcare",
"hotel-type": "dontcare",
"attraction-type": "dontcare",
"attraction-name": "dontcare",
"attraction-area": "dontcare",
"train-book_people": "dontcare",
"train-leaveAt": "dontcare",
"train-destination": "dontcare",
"train-day": "dontcare",
"train-arriveBy": "dontcare",
"train-departure": "dontcare"
}
```

C.4 Ignoring value candidates

From `MUL0540`:

```
"system": "Booking was successful, the total fee is 188.8
      ↪ GBP payable at the station. Reference number is: 31
      ↪ K4BK2E. Is there anything else you need?"
"user": "Yes, I am also looking for a multiple sports
      ↪ attraction in the city centre. Are you aware of any
      ↪ ?"
{
"attraction-type":"multiple sports",
"attraction-area":"city centre"
}

"system": "Unfortunately, there are no multiple sports
      ↪ attractions in the area you specified. Is there
      ↪ some other type of attraction or area I can look
      ↪ into for you?"
"user": "Alright, that's fine, how about colleges? Are there
      ↪  any of those in the city centre?"
{
"attraction-type":"colleges",
"attraction-area":"city centre"
}
```

C.5 Hallucinated slots

From `MUL0671`:

```
"system": "Cambridge has 8 moderately priced guesthouses in
      ↪ the north.  I would suggest Acorn Guest House, with
      ↪  a star rating of 4.  Would you like me to book it
      ↪ ?"
"user": "Yes please, I would like to book it for 7 people
      ↪ for 5 nights on Saturday, and I need a reference
      ↪ number."
{
"hotel-name":"Acorn Guest House",
"hotel-stars":"4",
"hotel-book_people":"7",
"hotel-book_day":"Saturday",
"hotel-book_stay":"5",
"hotel-reference_number":"?"
}

"system": "I'm very sorry I was unable to book at acorn
      ↪ guest house for 5 nights, would you like to try for
      ↪  a shorter stay?"
"user": "Yes, please try to book it for four nights instead
      ↪ ."
{
"hotel-book_stay":"4"
}
```

```
"system": "Your booking was successful, the reference number
      ↪  is WAEYAQ0M.  May I assist you with anything else
      ↪ today?"
"user": "No, that should be all for now. Thank you for the
      ↪ help!"
{
"hotel-reference_number":"WAEYAQ0M"
}
```

C.6 Arbitrary normalization

From `MUL0524`:

```
"system": "The cambridge and county folk museum is located
      ↪ at 2-3 castle street postcode cb30aq , it has an
      ↪ entrance fee of 3.50 pounds. "
"user": "Thanks.  Now I need to book a train from Norwich."
{
"train-departure": "Norwich",
"train-destination": "?",
"train-leaveAt": "?",
"train-day": "?",
"train-arriveBy": "?",
"train-book_people":"?"
}

"system": "There are 133 trains leaving norwich. What day
      ↪ and time are you interested in?"
"user": "I'm going to Cambridge on saturday and want to
      ↪ arrive by 14:15 please."
{
"train-destination": "Cambridge",
"train-day": "Saturday",
"train-arriveBy": "14:15"
}
```

C.7 Predicting DS_t instead of $\widehat{DS}_t$

From `MUL2116`:

```
"system": ""
"user": "Yes, can you tell me information on Cambridge
      ↪ lodgings?  We are looking for free wifi and a nice
      ↪ guesthouse."
{
"hotel-name": "?",
"hotel-area": "?",
"hotel-parking": "dontcare",
"hotel-pricerange": "dontcare",
"hotel-stars": "dontcare",
"hotel-internet": "yes",
"hotel-type": "guest house"
}

"system": "Sure. What area and price range are you looking
      ↪ for?"
"user": "I don't really care about area or price range, the
      ↪ most important thing for me is free wifi in a
      ↪ guesthouse that has free parking."
{
"hotel-name": "?",
"hotel-area": "dontcare",
"hotel-parking": "yes",
"hotel-pricerange": "dontcare",
"hotel-stars": "dontcare",
"hotel-internet": "yes",
"hotel-type": "guest house"
}

"system": "I've got over 20 guesthouses that will work with
      ↪ those requirements.  Would you like me to book a
      ↪ moderately priced 4 star? "
"user": "As long as it has free wifi and parking then yes."
{
"hotel-name": "?",
"hotel-area": "dontcare",
"hotel-parking": "yes",
"hotel-pricerange": "moderate",
"hotel-stars": "4",
"hotel-internet": "yes",
"hotel-type": "guest house"
}

"system": "I would recommend the Acorn Guest House on the
      ↪ north side of town.  It has 4-stars.  Should I book
      ↪  this?"
"user": "Yes. Please book it for 8 people for 5 nights
      ↪ starting from sunday."
{
"hotel-name": "Acorn Guest House",
"hotel-area": "north",
"hotel-parking": "yes",
"hotel-pricerange": "moderate",
"hotel-stars": "4",
"hotel-internet": "yes",
"hotel-type": "guest house",
"hotel-book_people": "8",
"hotel-book_day": "sunday",
"hotel-book_stay":"5"
}
```

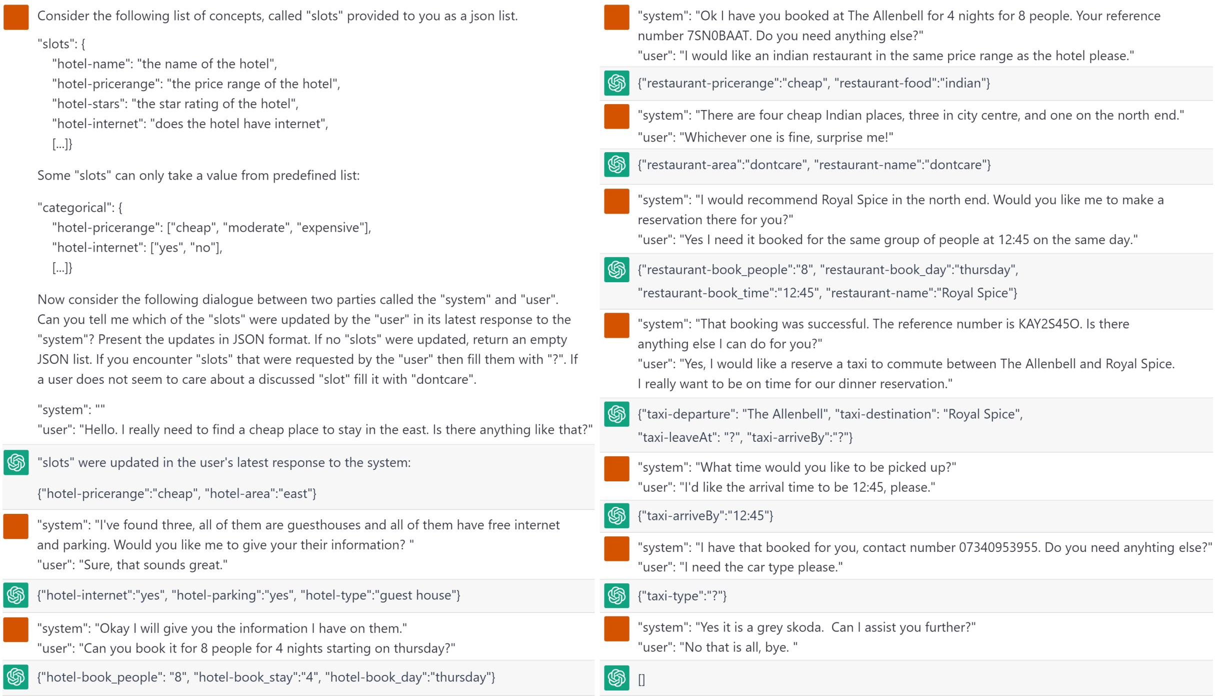
D “ChatGPT, are you a solution or an opportunity?”

We queried ChatGPT with our titular question and received the following response at the 5th regeneration.

```
ChatGPT for Zero-shot Dialogue State Tracking: A Solution or
      ↪  an Opportunity?

ChatGPT is a powerful language generation model that has the
      ↪  ability to generate human-like responses to a
      ↪ given prompt. It has been used in various natural
      ↪ language processing (NLP) tasks such as question
      ↪ answering, text summarization, and dialogue
      ↪ generation.

One potential application of ChatGPT is in zero-shot
      ↪ dialogue state tracking, which involves tracking
      ↪ the state of a conversation without any prior
      ↪ knowledge of the domain or the specific task. This
      ↪ could be useful in situations where the system
      ↪ needs to handle a wide range of topics and tasks,
      ↪ and where it is not feasible to manually annotate a
      ↪  large amount of training data.

However, there are also challenges with using ChatGPT for
      ↪ zero-shot dialogue state tracking. For example, it
      ↪ may not be able to accurately track the state of
      ↪ the conversation if the input is ambiguous or if
      ↪ the user's intent is not clear. Additionally, the
      ↪ model's responses may not always be appropriate or
      ↪ relevant to the current conversation.

Overall, while ChatGPT has the potential to be a solution
      ↪ for zero-shot dialogue state tracking, it also
      ↪ presents opportunities for further research and
      ↪ development in this area. It could be combined with
      ↪  other techniques such as reinforcement learning or
      ↪  active learning to improve its performance, or
      ↪ integrated with other NLP models such as dialogue
      ↪ state tracking models to provide more accurate and
      ↪ context-aware responses.
```

```
Consider the following list of concepts, called "slots" provided to you as a json list.

"slots": {
    "taxi-leaveAt": "the departure time of the taxi",
    "taxi-destination": "the destination of the taxi",
    "taxi-departure": "the departure of the taxi",
    "taxi-arriveBy": "the arrival time of the taxi",
    "restaurant-book_people": "the amount of people to book the restaurant for",
    "restaurant-book_day": "the day for which to book the restaurant",
    "restaurant-book_time": "the time for which to book the restaurant",
    "restaurant-food": "the food type of the restaurant",
    "restaurant-pricerange": "the price range of the restaurant",
    "restaurant-name": "the name of the restaurant",
    "restaurant-area": "the location of the restaurant",
    "hotel-book_people": "the amount of people to book the hotel for",
    "hotel-book_day": "the day for which to book the hotel",
    "hotel-book_stay": "the amount of nights to book the hotel for",
    "hotel-name": "the name of the hotel",
    "hotel-area": "the location of the hotel",
    "hotel-parking": "does the hotel have parking",
    "hotel-pricerange": "the price range of the hotel",
    "hotel-stars": "the star rating of the hotel",
    "hotel-internet": "does the hotel have internet",
    "hotel-type": "the type of the hotel",
    "attraction-type": "the type of the attraction",
    "attraction-name": "the name of the attraction",
    "attraction-area": "the area of the attraction",
    "train-book_people": "the amount of people to book the train for",
    "train-leaveAt": "the departure time of the train",
    "train-destination": "the destination of the train",
    "train-day": "the day for which to book the train",
    "train-arriveBy": "the arrival time of the train",
    "train-departure": "the departure of the train"
}

Some "slots" can only take a value from predefined list:

"categorical": {
    "hotel-pricerange": ["cheap", "moderate", "expensive"],
    "hotel-area": ["north", "south", "east", "west", "centre"],
    "hotel-parking": ["yes", "no"],
    "hotel-internet": ["yes", "no"],
    "hotel-type": ["hotel", "guest house"],
    "restaurant-pricerange": ["cheap", "moderate", "expensive"],
    "restaurant-area": ["north", "south", "east", "west", "centre"],
    "attraction-area": ["north", "south", "east", "west", "centre"]
}

Now consider the following dialogue between two parties called the "system" and "user". Can you tell me which of the "slots"
     ↪ were updated by the "user" in its latest response to the "system"? Present the updates in JSON format. If no "slots"
     ↪ were updated, return an empty JSON list. If you encounter "slots" that were requested by the "user" then fill them
     ↪ with "?". If a user does not seem to care about a discussed "slot" fill it with "dontcare".
```

Figure 2: Prompt P.



Consider the following list of concepts, called "slots" provided to you as a json list.

```
"slots": {
    "hotel-name": "the name of the hotel",
    "hotel-pricerange": "the price range of the hotel",
    "hotel-stars": "the star rating of the hotel",
    "hotel-internet": "does the hotel have internet",
    [...]}
```

Some "slots" can only take a value from predefined list:

```
"categorical": {
    "hotel-pricerange": ["cheap", "moderate", "expensive"],
    "hotel-internet": ["yes", "no"],
    [...]}
```

Now consider the following dialogue between two parties called the "system" and "user". Can you tell me which of the "slots" were updated by the "user" in its latest response to the "system"? Present the updates in JSON format. If no "slots" were updated, return an empty JSON list. If you encounter "slots" that were requested by the "user" then fill them with "?". If a user does not seem to care about a discussed "slot" fill it with "dontcare".

"system": ""
"user": "Hello. I really need to find a cheap place to stay in the east. Is there anything like that?"

"slots" were updated in the user's latest response to the system:

{"hotel-pricerange":"cheap", "hotel-area":"east"}

"system": "I've found three, all of them are guesthouses and all of them have free internet and parking. Would you like me to give your their information? "
"user": "Sure, that sounds great."

{"hotel-internet":"yes", "hotel-parking":"yes", "hotel-type":"guest house"}

"system": "Okay I will give you the information I have on them."
"user": "Can you book it for 8 people for 4 nights starting on thursday?"

{"hotel-book_people": "8", "hotel-book_stay":"4", "hotel-book_day":"thursday"}

"system": "Ok I have you booked at The Allenbell for 4 nights for 8 people. Your reference number 7SN0BAAT. Do you need anything else?"
"user": "I would like an indian restaurant in the same price range as the hotel please."

{"restaurant-pricerange":"cheap", "restaurant-food":"indian"}

"system": "There are four cheap Indian places, three in city centre, and one on the north end."
"user": "Whichever one is fine, surprise me!"

{"restaurant-area":"dontcare", "restaurant-name":"dontcare"}

"system": "I would recommend Royal Spice in the north end. Would you like me to make a reservation there for you?"
"user": "Yes I need it booked for the same group of people at 12:45 on the same day."

{"restaurant-book_people":"8", "restaurant-book_day":"thursday",
"restaurant-book_time":"12:45", "restaurant-name":"Royal Spice"}

"system": "That booking was successful. The reference number is KAY2S45O. Is there anything else I can do for you?"
"user": "Yes, I would like a reserve a taxi to commute between The Allenbell and Royal Spice. I really want to be on time for our dinner reservation."

{"taxi-departure": "The Allenbell", "taxi-destination": "Royal Spice",
"taxi-leaveAt": "?", "taxi-arriveBy":"?"}

"system": "What time would you like to be picked up?"
"user": "I'd like the arrival time to be 12:45, please."

{"taxi-arriveBy":"12:45"}

"system": "I have that booked for you, contact number 07340953955. Do you need anyhting else?"
"user": "I need the car type please."

{"taxi-type":"?"}

"system": "Yes it is a grey skoda. Can I assist you further?"
"user": "No that is all, bye. "

[]

Figure 3: Example of DST with ChatGPT. P is abridged for brevity. The sentence “"slots" were updated [...]” was generated at each turn. In turn 2, Boolean slots are predicted correctly, and a value variant (“guesthouse”) is mapped to the correct label. In turn 4 a coreference is resolved correctly. In turn 5, *dontcare* is handled correctly. In turn 7, the model assumes implicit requests. In turn 9, ChatGPT hallucinates a slot which is not listed in P, but sensible.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 5, Section "Limitations" (unnumbered)

☑ A2. Did you discuss any potential risks of your work?
Section 5, Section "Limitations" (unnumbered), Section "Ethics Statement" (unnumbered)

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract, Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Entirety of the paper

☑ B1. Did you cite the creators of artifacts you used?
Entirety of the paper

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section "Ethics Statement" (unnumbered)

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section "Ethics Statement" (unnumbered)

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Section "Ethics Statement" (unnumbered)

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Documentation of artifacts cited

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4

C ☑ Did you run computational experiments?

Section 4

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Model is proprietary and runs as black box.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.