Learning to Ask Questions for Zero-shot Dialogue State Tracking

Diogo Tavares
NOVA University of Lisbon
NOVA LINCS
Lisbon, Portugal
dc.tavares@campus.fct.unl.pt

David Semedo
NOVA University of Lisbon
NOVA LINCS
Lisbon, Portugal
df.semedo@fct.unl.pt

Alexander Rudnicky
Carnegie Mellon University
LTI
Pittsburgh, PA, USA
air@cs.cmu.edu

Joao Magalhaes
NOVA University of Lisbon
NOVA LINCS
Lisbon, Portugal
jmag@fct.unl.pt

ABSTRACT

We present a method for performing zero-shot Dialogue State Tracking (DST) by casting the task as a learning-to-ask-questions framework. The framework learns to pair the best question generation (QG) strategy with in-domain question answering (QA) methods to extract slot values from a dialogue without any human intervention. A novel self-supervised QA pretraining step using in-domain data is essential to learn the structure without requiring any slot-filling annotations. Moreover, we show that QG methods need to be aligned with the same grammatical person used in the dialogue. Empirical evaluation on the MultiWOZ 2.1 dataset demonstrates that our approach, when used alongside robust QA models, outperforms existing zero-shot methods in the challenging task of zero-shot cross domain adaptation—given a comparable amount of domain knowledge during data creation. Finally, we analyze the impact of the types of questions used, and demonstrate that the algorithmic approach outperforms template-based question generation.

CCS CONCEPTS

• **Information systems** → **Question answering**; • **Computing methodologies** → **Natural language processing**.

KEYWORDS

dialogue state tracking, zero-shot, question answering

ACM Reference Format:
Diogo Tavares, David Semedo, Alexander Rudnicky, and Joao Magalhaes. 2023. Learning to Ask Questions for Zero-shot Dialogue State Tracking. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3539618.3592010

1 INTRODUCTION

Task-oriented dialogue (TOD) systems interact with a user using natural language to achieve a specific goal, such as booking airline tickets or making a restaurant reservation. The state of a dialogue is defined by a set of *(key, value) pairs* which represent the information requested or accepted by the user, up to the current utterance. Knowing which important slots have and have not been *filled* allows a dialogue system to guide the conversation to quickly gather the information needed to complete the task at hand. We refer to assigning *values* to the set of *slot keys* in an utterance as *slot-filling*, and the task of maintaining this state throughout an entire dialogue as *Dialogue State Tracking (DST)*. A TOD system deployed in the wild should adapt to new functionalities without needing to be fully rebuilt. However, to train the system on the new information, it needs dialogue data and DST annotations. Circumventing this problem requires models with **zero-shot capabilities**; specifically, models which are robust not only to the addition of new slots, but also entirely new domains. When early open-vocabulary DST approaches [13, 23] tackled this task, they were faced with a typical difficulty of the zero-shot learning setting: how to best employ external textual knowledge, such as slot keys, to maximize performance in unseen domains [3]. More recent work [16, 26] uses textual descriptions of the domains and slots, which, though richer, are also noisier [3], and require human intervention to be created when the domain is novel. In an ideal zero-shot system, no human effort is needed to bootstrap—or enable—zero-shot capabilities.

To tackle these challenges, we begin by casting the DST task as a reading comprehension problem, allowing for explicit zero-shot DST support. As each slot will be directly linked to a question, we also propose ways to generate the set of questions pertaining to each slot. We propose a novel method which emphasizes the relationships between semantically similar slots, which are likely to share values. Lastly, we propose two self-supervised approaches to bootstrap DST models as QA, which address two typical situations: when in-domain dialogues exist, but no annotations are present, and when no in-domain dialogues exist. While we loosely follow the the approach presented by Honovich et al. [11], as far as we know, we are the first to use question generation techniques for pretraining in the DST task. In the latter, the larger and richer pool of data offered by the QA domain allows us to bootstrap models that can adapt to novel zero-shot classes.

Our contributions are twofold: first, we propose a self-supervised approach to pretrain DST models with in-domain and out-of-domain QA data, as a way to bootstrap DST models (Section 3.1). Second, we present a question generation strategy which emphasizes the relationships between semantically similar slots—slots which refer to similar types of values—as a way to maximize zero-shot DST performance (Section 3.2). [1]

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07.
https://doi.org/10.1145/3539618.3592010

[1] We release all sets of questions and prompts in https://github.com/d-c-t/learning-to-ask-questions-zero-shot-dst.

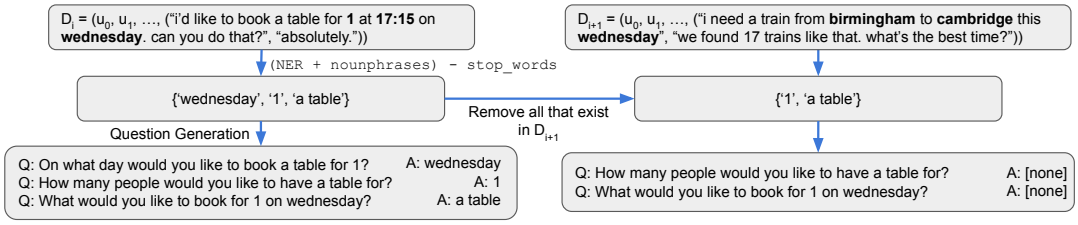

Figure 1: Step-by-step illustration of question generation for model pretraining. We emphasize (in bold) the slot annotations in u_i, to highlight how they overlap with the outputs of span extraction.

2 RELATED WORK

Early DST approaches [13, 23] heavily focused on not only the full-shot performance, but also on its behavior under low—and no—data situations, using the slot keys and RNN to generate the slot value. Rastogi et al. [22] and Ma et al. [17] perform slot-filling in zero-shot scenarios using Transformer encoders. Both models use extra information when encoding categorical slots: the former encoding all possible slot values; the latter performing feature engineering. Later works [16, 26] use the user intent and slot descriptions, which are richer than slot keys alone [3], as input to a T5 [20] model, and generate the dialogue state during the decoding step. Gupta et al. [9] and Hu et al. [12] use dialogue examples to prompt Large Language Models. Gupta et al. [9] use a single hand-crafted dialogue example per domain in conjunction with a T5 variant [20]. The performance increases proportionally with the required supervision. Hu et al. [12], rather than manually writing a dialogue, retrieve the most similar training example, and perform slot-filling with GPT-3 [1].

Some efforts have been made into mapping the QA domain to DST, following the success of approaches relying on span extraction [2, 24]. Gao et al. [7] explicitly model slot-filling as a series of questions about each slot and decodes the answers with a RNN. Zhou and Small [27] ask a question per slot, and propose a dynamic graph that attempts to model relations between semantically similar slots. Gao et al. [6] and Namazifar et al. [18] introduce one handcrafted question per slot key. The former explicitly separates non-categorical slots as span-based QA, and categorical slots as multiple choice QA, thus requiring all examples to be listed. The latter employs models previously trained on the SQuAD [21] dataset to perform few-shot DST. Li et al. [14] also uses handcrafted questions and a GPT-2 based decoder to generate slot values. Lin et al. [15] employ a T5 model trained on several QA datasets and shows that knowledge transfer between the QA to DST domains is feasible. Categorical slots are also handled as multiple choice QA. Rule-based question generation approaches for pretraining have been explored by Du et al. [4]. Pretraining is performed by transforming slot annotations from an unrelated dataset into a set of questions, using its slot ontology. BERT models pretrained in SQuAD have also been used with span-based slot-filling, using slot descriptions as auxiliary inputs [8]. Lastly, one of the baselines of Lin et al. [16] uses template-based question generation and a T5-small model fine-tuned for QA. While casting DST to QA has been shown to be promising [14, 15, 18], we focus on the underexplored problem of identifying the types of questions to be asked per slot. Furthermore, while in-domain and out-of-domain pretraining have been shown to improve performance [4, 8, 18], this avenue remains largely underexplored.

3 DST AS QUESTION ANSWERING

We cast DST as a reading comprehension problem by converting slot-filling annotations into question-answering. To do so, given a domain with a set of slot keys S, we create one natural language question $q_i \in Q_{slot}$ for each slot key $s_i \in S$, with $0 \leq i < |S|$. Then, for both user and agent utterances u_n at turn n of a dialogue D, so that $u_n \in D$, we input $((u_0, u_1, ..., u_n), Q_{slot})$ to a QA model. The dialogue state at turn n is the aggregation of all model responses to Q_{slot}, using all utterances until that point $(u_0, u_1, ..., u_n)$ as context. While some approaches to QA-based zero-shot slot-filling [15, 16, 26] follow a multiple-choice paradigm for categorical slots, we rely *only* on D and Q_{slot} for this task, meaning we rely on less data, and require less data to be manually created in real-world situations. Furthermore, we bootstrap our models by either pretraining them with in or out of domain data, without slot-filling annotations. Finally, we show our findings through two classes of QA model: *span-based*, which requires slot values to be explicitly mentioned in the text, and *generative*, which does not.

3.1 Self-Supervised Reading Comprehension Pretraining

To bootstrap our QA models for the DST task, we investigate two pretraining techniques: one focusing on out of domain QA data, and one on in-domain dialogues, without using DST annotations. Each strategy tackles a different zero-shot environment: the former, when no dialogues of the new domains exist; the latter, when dialogues in the novel domain already exist, but no DST annotations are present. We hypothesize that each pretraining method confers its own distinct advantages. Models pretrained on QA data interact with a larger range of domains and high-quality questions. Being trained on several domains may allow them to generalize in the zero-shot cross-domain adaptation, where the model is presented with multiple domains and must generalize to an unseen one. On the other hand, we posit that there are two main benefits to introducing a self-supervised, in-domain, pretraining step. First, by supplying models with highly varied albeit lower-quality in-domain examples, we increase their familiarity with language and domain characteristics. This allows the models to capture common language patterns specific to that domain. Second, it introduces a bias towards answering with named entities and noun phrases—which we posit that makes up most of the slot values.

Each initialization technique is based on the creation of a set of questions Q_{pre} and their answers A_{pre}. For the out-of-domain examples, we use the question and answer sets provided in SQuAD 2.0 [21], due to its high variety of domains and high quality questions. For the in-domain examples, we first create A_{pre} by extracting informative spans from the dialogues of the MultiWOZ training split, using spaCy [10] to tag named entities, noun phrases, and to remove stop-words from the extracted set. Then, for each element of A_{pre}, we use a T5 model, fine-tuned on the question generation task using SQuAD[2], to generate Q_{pre}, given elements of A_{pre} as the *answer* and the dialogue each element is extracted from as its *context* paragraph. We use greedy decoding to generate each question. QA models for DST require the ability to classify questions as unanswerable, as each utterance will generally only mention a small fraction of slots. This is crucial for maintaining good performance, as error propagation caused by misclassifications can lead to a rapid decrease in joint goal accuracy. In the out-of-domain pretraining step, we use the unanswerable SQuAD question set. In the in-domain step, we apply each question Q^i_{pre} to the next dialogue in the training set, if the dialogue does not explicitly contain the corresponding answer A^i_{pre}. We illustrate this strategy in Figure 1.

3.2 Slot-based Question Formulation (Q_{slot})

We will refer to any information in a slot-key that is not its domain as a *slot type*. For instance, given the slot-key *hotel-pricerange*, the domain is *hotel* and the type is *pricerange*. To study the impact of Q_{slot} on model performance, we experiment with its formulation in a number of ways.The strategies include *template-based*, *LLM-based*, and *handcrafted* approaches.

3.2.1 Template. This approach relies on the following templates: *"what is the <slot key>?"* and *"what is the value of the slot <slot key> ?"* [7]—referred to as *what-is* and *simple*, respectively.

3.2.2 LLM. We generate Q_{slot} by prompting a Large Language Model (LLM) so that a natural language question is composed for each slot type. This reduces the human bias inherent to manually generating slot questions [14] or descriptions [16, 26]. An immediate pitfall lies in language models not explicitly containing domain knowledge. This may cause the questions to misrepresent slot semantics when those are not immediately clear from the type. We define two LLM prompt strategies for question generation: *LLM*, which simply prompts all questions to be generated, and *Pronouns*, in which the LLM prompt requires the pronouns in the question to be in the third person, and refer to "the user", whenever needed.

3.2.3 Handcrafted. We also manually created a set of questions that contain explicit domain knowledge, alongside explicitly enforcing semantically similar slots to have similar questions. The resulting set is composed of questions less abstract and more meaningful to the dialogue than those in template-based approaches [6].

3.2.4 Implementation Details. When using **span-based** QA models, we prepend { Yes, No, Dontcare } to each question [18]. This allows the model to respond to questions pertaining to service-based slots (namely, *hotel-internet, hotel-parking*) with the required "yes" and "no". Furthermore, this facilitates assigning the commonly used "dontcare" value to questions. We initialize the embedding of the Dontcare token to the average of embeddings in "do not care". We format the input of **generative models** with the "*question:* q_i *context:* D" prompt.

[2]https://huggingface.co/mrm8488/t5-base-finetuned-question-generation-ap

3.3 Reading Comprehension Fine-Tuning

We use the MultiWOZ 2.1 [5] dataset for inference and fine-tuning. We fine-tune our models following the TRADE [23] pre and post-processing methodology, as recommended by the MultiWOZ authors, to ensure our results are directly comparable to the literature. We train and evaluate following a cross-domain adaptation setting, meaning we train on 4 dataset domains, and evaluate on the unseen domain data, with the proposed reading comprehension approach. Given that MultiWOZ is composed of 5 domains, we train a total of 5 instances of each model. We evaluate using the joint goal accuracy metric on the target slots, and use the average of the five runs as the main comparison metric. We use RoBERTa-base as our backbone span-based QA model and T5-base as our backbone generative model. We generate the *LLM* and *Pronouns* Q_{slot} using a single InstructGPT [19] input. Unless otherwise stated, we use the *LLM* Q_{slot} strategy in our experiments.

4 RESULTS AND DISCUSSION

4.1 Baselines

We compare our performance with the following approaches: **TRADE** [23] and **MA-DST** [13], which use RNNs and slot names as input; **Li et al. [14]** manually write questions and use a GPT-2 model for decoding the slot values. **T5-DST** [16], which uses slot descriptions with a T5-small model—**T5-DST (QA)**, one of the presented baselines in T5-DST, uses automatic question generation. This model is evaluated on MultiWOZ 2.0, which may not make it directly comparable to our work. **TransferQA** [15] pretrains a T5-large model using several QA datasets and applies it to the DST task, without any in-domain training; **D3ST** [26] uses slot descriptions and a T5 model. Finally, **SDT** [9] uses a T5-XXL model, and requires dialogue examples to be manually written from analyzing the data. SDT always contains the input of a handwritten example, making it explicitly a one-shot approach. Li et al. [14], T5-DST, D3ST, and SDT require more domain expertise and a higher developer workload to express new slots: not only do they require manually creating slot descriptions or examples, but also explicitly listing all values of categorical slots—which may not be trivial when adding new slots. While generating all the descriptions for D3ST and examples for SDT (in the SGD dataset [22]) takes 1.5 and 2 hours, respectively [9], we generate Q_{slot} automatically from only the slot keys, the minimal representation of a slot; thus requiring less data.

4.2 Zero-shot cross-domain adaptation

Table 1 presents the results for the zero-shot domain transfer task. Each column represents an instance of the model trained on all domains except one, then evaluated on that domain. The metric is the standard per-domain joint goal accuracy (JGA). We also present the average JGA over all domains, which we use as our main means of comparison. We obtain SOTA performance versus models with comparable amounts of information (slot keys only), consistently outperforming them in most domains, oftentimes by large margins.

Table 1: Zero-shot cross-domain adaptation results. The top approaches rely on simple features, such as slot names. The bottom approaches rely on more complex features, such as slot descriptions or full examples.

Model	Hotel	Rest.	Taxi	Attr.	Train	Avg.
Fully zero-shot						
TRADE [23]	13.7	11.5	60.6	19.9	22.4	25.6
MA-DST [13]	16.3	13.6	59.3	22.5	22.8	26.9
T5-DST (QA) [16]	19.8	21.8	64.4	32.5	32.6	34.3
TransferQA [15]	22.7	26.3	61.9	31.3	**36.7**	35.8
T5-self (ours)	**29.4**	**54.0**	**65.5**	**46.5**	35.6	**46.2**
Zero-shot with extra information						
Li et al. [14]	24.4	26.2	59.6	31.3	29.1	34.1
T5-DST [16]	21.2	21.7	64.6	33.1	35.4	35.2
D3ST [26]	21.8	38.2	78.4	56.4	38.7	46.7
SDT [9]	33.9	72.0	86.4	74.4	62.9	65.9

Table 2: We experiment with all classes of the models. Model nomenclature consists of the backbone model name (RoBERTa or T5), then the pretraining strategy.

Model	Hotel	Rest.	Taxi	Attr.	Train	Avg.
RoBERTa	27.8	30.0	63.1	38.1	24.3	36.7
RoBERTa-squad	29.4	35.5	61.6	38.5	23.7	37.7
RoBERTa-self	**29.8**	47.3	64.1	40.8	28.3	42.1
T5	27.5	35.4	**66.1**	43.6	38.9	42.3
T5-squad	28.8	37.7	65.3	44.1	**39.2**	43.0
T5-self	29.4	**54.0**	65.5	**46.5**	35.6	**46.2**

However, we under-perform when compared to D3ST and SDT. This is expected, as these models utilize descriptions or full examples, alongside listing all possible slot values when applicable. Our model obtains comparable performance while using substantially less information.

In Table 2, we show both classes of models and each pretraining step. Generally, the sequence-to-sequence models outperform encoder-based approaches. This is due in part to span-based models not supporting categorical slots directly, which make up a significant number of slots [25].

4.2.1 Impact of pretraining. Table 2 also displays the impact of different pretraining strategies. Our in-domain pretraining method outperforms models pretrained on SQuAD. Nevertheless, both strategies show strong improvements when compared to the vanilla backbone models. This indicates that pretraining in the QA task is useful for zero-shot slot-filling. Note that although our pretraining strategy doesn't use any DST annotations, it still interacts with dialogues whose domains we evaluate on the cross-domain adaptation task. This situation, however, can be reflected in the real world, where users of a dialogue agent may attempt to invoke features before they are present, and the features are later added. In this situation, our in-domain pretraining methodology is viable—the dialogues happen, but there are no annotations—so we argue that it is comparable to the other approaches. This may also indicate that more pretraining can be done: we can combine other QA datasets besides SQuAD [15], and even use other DST data when applying our pretraining step. This is left as future work.

Table 3: T5-self performance with different question generation strategies.

	Hotel	Rest.	Taxi	Attr.	Train	Avg.
Generative						
LLM	**29.4**	54.0	**65.5**	46.5	35.6	**46.2**
Pronouns	29.2	**54.3**	65.4	**47.5**	32.5	45.8
Template-based						
What-is	28.5	47.0	63.7	44.4	39.9	44.7
Simple	28.9	44.3	64.5	46.0	40.7	44.9
Manual	29.0	45.5	65.0	41.9	**41.9**	44.6

4.2.2 Impact of Q_{slot}. Table 3 displays the impact of different question generation approaches, with each row referring to a unique strategy. The performance improvement of *LLM* versus *Pronouns* highlights how small differences in questions can make a difference: using third person pronouns whenever possible contrasts with the questions present in both pretraining strategies. Questions with the natural pronouns of a two-person dialogue (i.e. "you" and "I") generally outperformed these. Forcing natural pronouns causes the performance of LLM-based Q_{slot} strategies to outperform manual question generation. On the other hand, we find that simpler, template-based approaches, perform comparably to question generation, without requiring access to a LLM or prompt. The semantic similarities between slots, while not as explicit, are still in display when using these questions. It's worth noting that the performance improvement of the LLM strategy isn't consistent across domains. We see a steep performance drop in the *train* domain, largely due to the LLM misrepresenting the slots *train-arriveby* and *train-leaveat*. The *train-leave day* question is also worth noting: although semantically correct, it is considerably different from the pretraining questions. Both these factors could explain its—and the *train* domain's—lower overall performance. In the *Manual* strategy, where *train-arriveby* and *leaveat* are fixed, the performance is considerably higher. This indicates that there is space for manual, domain-specific, tuning when generating Q_{slot}, which goes beyond our zero-shot DST premise.

5 CONCLUSIONS

We presented a zero-shot DST method that requires no human input or domain knowledge, through QA. The questions are posed by a generative LLM, and as such, new slots and domains can be easily added without any extra manual workload. We experimented with two pretraining methodologies, both of which outperform non-pretrained models, and as such, may be used to bootstrap models throughout the early data collection processes, to train more robust, full-shot, approaches. For future work, we plan to explore other approaches to automatically generate Q_{slot}, introduce pretraining methods which focus on more, varied, data, and explore methods for automatically correcting misrepresented slot questions.

6 ACKNOWLEDGEMENTS

This work was partially funded by the FCT Scholarship PRT/BD/152803 /2021, the NOVA LINCS project (UIDP/04516/2020), and the CMU Portugal project iFetch (LISBOA-01-0247-FEDER-045920).

REFERENCES

[1] 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*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901.

[2] Guan Lin Chao and Ian Lane. 2019. BERT-DST: Scalable End-to-End Dialogue State Tracking with Bidirectional Encoder Representations from Transformer. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* 2019-September (7 2019), 1468–1472. https://doi.org/10.48550/arxiv.1907.03040

[3] Jiaoyan Chen, Yuxia Geng, Zhuo Chen, Ian Horrocks, Jeff Z Pan, and Huajun Chen. 2021. Knowledge-aware Zero-Shot Learning: Survey and Perspective. (2021).

[4] Xinya Du, Luheng He, Qi Li, Dian Yu, Panupong Pasupat, and Yuan Zhang. 2021. QA-Driven Zero-shot Slot Filling with Weak Supervision Pretraining. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 654–664. https://doi.org/10.18653/v1/2021.acl-short.83

[5] Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tur. 2019. MultiWOZ 2.1: Multi-Domain Dialogue State Corrections and State Tracking Baselines. *arXiv preprint arXiv:1907.01669* (2019).

[6] Shuyang Gao, Sanchit Agarwal, Tagyoung Chung, Di Jin, and Dilek Hakkani-Tur. 2020. From Machine Reading Comprehension to Dialogue State Tracking: Bridging the Gap. (4 2020), 79–89. https://doi.org/10.48550/arxiv.2004.05827

[7] Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019. Dialog state tracking: A neural reading comprehension approach. *SIGDIAL 2019 - 20th Annual Meeting of the Special Interest Group Discourse Dialogue - Proceedings of the Conference*, 264–273. https://doi.org/10.18653/v1/w19-5932

[8] Pavel Gulyaev, Eugenia Elistratova, Vasily Konovalov, Yuri Kuratov, Leonid Pugachev, and Mikhail Burtsev. 2020. Goal-Oriented Multi-Task BERT-Based Dialogue State Tracker. (2 2020). https://doi.org/10.48550/arxiv.2002.02450

[9] Raghav Gupta, Harrison Lee, Jeffrey Zhao, Abhinav Rastogi, Yuan Cao, and Yonghui Wu. 2022. Show, Don't Tell: Demonstrations Outperform Descriptions for Schema-Guided Task-Oriented Dialogue. *NAACL 2022 - 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference* (4 2022), 4541–4549. https://doi.org/10.18653/v1/2022.naacl-main.336

[10] Matthew Honnibal and Mark Johnson. 2015. An Improved Non-monotonic Transition System for Dependency Parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 1373–1378. https://aclweb.org/anthology/D/D15/D15-11627D

[11] Or Honovich, Leshem Choshen, Roee Aharoni, Ella Neeman, Idan Szpektor, and Omri Abend. 2021. Q^2: Evaluating Factual Consistency in Knowledge-Grounded Dialogues via Question Generation and Question Answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 7856–7870. https://doi.org/10.18653/v1/2021.emnlp-main.619

[12] 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. (3 2022). https://doi.org/10.48550/arxiv.2203.08568

[13] Adarsh Kumar, Peter Ku, Anuj Goyal, Angeliki Metallinou, and Dilek Hakkani-Tur. 2020. MA-DST: Multi-Attention-Based Scalable Dialog State Tracking. *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (4 2020), 8107–8114. Issue 05. https://doi.org/10.1609/AAAI.V34I05.6322

[14] 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 *Conference of the European Chapter of the Association for Computational Linguistics*.

[15] Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Paul Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021. Zero-Shot Dialogue State Tracking via Cross-Task Transfer. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings* (9 2021), 7890–7900. https://doi.org/10.48550/arxiv.2109.04655

[16] Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Andrea Madotto, Eunjoon Cho, and Rajen Subba. 2021. Leveraging Slot Descriptions for Zero-Shot Cross-Domain Dialogue StateTracking. 5640–5648. https://doi.org/10.18653/v1/2021.naacl-main.448

[17] Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiying Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. 2019. An End-to-End Dialogue State Tracking System with Machine Reading Comprehension and Wide & Deep Classification. (12 2019). https://doi.org/10.48550/arxiv.1912.09297

[18] Mahdi Namazifar, Alexandros Papangelis, Gokhan Tur, and Dilek Hakkani-T¨ur. 2021. Language model is all you need: Natural language understanding as Question answering. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* 2021-June, 7803–7807. https://doi.org/10.1109/ICASSP39728.2021.9413810

[19] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 27730–27744. https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf

[20] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html.

[21] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)* 2 (6 2018), 784–789. https://doi.org/10.48550/arxiv.1806.03822

[22] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, 8689–8696. https://doi.org/10.1609/aaai.v34i05.6394

[23] 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 Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 808–819. https://doi.org/10.18653/v1/p19-1078

[24] Puyang Xu and Qi Hu. 2018. An End-to-end Approach for Handling Unknown Slot Values in Dialogue State Tracking. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)* 1 (5 2018), 1448–1457. https://doi.org/10.48550/arxiv.1805.01555

[25] Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. 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, 109–117. https://doi.org/10.18653/v1/2020.nlp4convai-1.13

[26] Jeffrey Zhao, Raghav Gupta, Yuan Cao, Dian Yu, Mingqiu Wang, Harrison Lee, Abhinav Rastogi, Izhak Shafran, Yonghui Wu, and Google Research. 2022. Description-Driven Task-Oriented Dialog Modeling. (1 2022). https://doi.org/10.48550/arxiv.2201.08904

[27] Li Zhou and Kevin Small. 2019. Multi-domain Dialogue State Tracking as Dynamic Knowledge Graph Enhanced Question Answering. (2019). http://arxiv.org/abs/1911.06192