INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Parameter-Efficient Low-Resource Dialogue State Tracking by Prompt Tuning

Mingyu Derek Ma[1], *Jiun-Yu Kao*[2], *Shuyang Gao*[§], *Arpit Gupta*[2], *Di Jin*[2], *Tagyoung Chung*[2], *Nanyun Peng*[1,2]

[1]University of California, Los Angeles [2]Amazon Alexa AI

{ma, violetpeng}@cs.ucla.edu,
{jiunyk, guparpit, djinamzn, tagyoung}@amazon.com, shuyangg@gmail.com

Abstract

Dialogue state tracking (DST) is an important step in dialogue management to keep track of users' beliefs. Existing works fine-tune all language model (LM) parameters to tackle the DST task, which requires significant data and computing resources for training and hosting. The cost grows exponentially in the real-world deployment where dozens of fine-tuned LM are used for different domains and tasks. To reduce parameter size and better utilize cross-task shared information, we propose to use soft prompt token embeddings to learn task properties. Without tuning LM parameters, our method drastically reduces the number of parameters needed to less than 0.5% of prior works while achieving better low-resource DST performance.

Index Terms: dialogue state tracking, prompt tuning

1. Introduction

Dialogue state tracking (DST) that extracts structured conversation progress in a list of slot-value pairs from unstructured dialogue utterances is an essential component of a dialogue system [1]. Unlike classification-based models that pick the slot value from given candidate [2, 3], recent works formulate DST as a conditional generation task [4, 5], where the concatenation of dialogue history and a slot-specific prompt are fed to generative models and the text generation output are decoded to predicted slot values [6, 7]. This formulation enjoys the benefit of generalizability to unseen domains and slot types beyond a defined dialogue ontology [8, 9].

General prompting methods use a textual prompt to provide task information to the LM [10, 11]. Prior works have variations that update different parameter combinations such as both LM and prompt token embeddings [12, 13, 14, 15], only the token embeddings of the LM [16], or only the prompt token embeddings [17, 18, 19].

While there are some existing prompt-based approaches for DST with different designs of prompts such as using slot name [20, 21, 22, 23], slot description [24], slot type [25], possible values [25], priming examples [26] and/or slot-specific question [4, 27, 28, 29, 8, 30] in prompt sentences, they all fine-tune the entire LM along with the prompt tokens for a new domain, which requires a significant amount of training time, system resources, and annotated data [31, 32]. The computing and data resource-hungry issues are more severe in the real-world deployment where LMs tuned for different domains and tasks need to be trained and hosted, and a typical dialogue system has to serve dozens of such LMs [33, 34, 35]. This leads to a high cost of the development and service of dialogue systems and constrains offline deployment. In addition, limited data is available for a new domain or task.

We propose a **parameter-efficient** and **data-efficient** DST model for **low-resource** settings, which only needs to update 0.08% of parameters compared with the previous best model, by keeping LM parameters frozen and introducing soft prompt tokens to represent task properties of different slots. Figure 1 gives an overview of our model. The only prior work we are aware of that only updates prompt token embeddings and thus parameter-efficient is [36], but it focuses on continual domain adaptation and with a significant amount of training data.

Our design introduces three techniques that are generalizable to other generative-based information extraction models. 1) **Task-specific parameters**: *task prompt tokens* are introduced to specifically learn domain, slot and slot type information so that the model behaves according to the task; *word-mapping prompt tokens* enable us to obtain task knowledge contained in natural language instruction and optimize human-created prompts with continuous embedding space. 2) **Task metadata in objective**: we introduce the reiteration technique in the target sequence in order to include explicit task signals in the text generation objective. 3) **Distinguishing segments**: segment embeddings help the model identify the prompt segment, dialogue speakers, and question partition. Our proposed method enables much more efficient dialogue system deployment as only one LM needs to be hosted and inference for different domains could be realized by feeding domain-specific prompt token embeddings into the transformer stack.

Experiments on MultiWOZ 2.0 show that our method achieves better performance on low-resource DST with orders of magnitude fewer parameters. We further conduct ablation studies, error analysis, and examine the semantic information shown in the prompt tokens. We observe that our model is more specialized in predicting categorical slot values, is more conservative for slots with free output space and introduces more hallucination errors for categorical slots.

2. Method

We introduce task definition (Section 2.1), overall framework (Section 2.2) and soft prompt designs (Section 2.3).

2.1. Task definition

The goal is to construct a belief state with $|S|$ pairs of slot and value at a certain turn in a multi-turn conversation. All the turns up to the query turn are dialogue history, and slot-specific information (*i.e.* name, description, value candidates, question and type of the slot) is provided. There are 5 slot types, *i.e.* CATEGORICAL, DAY, NUMBER, OPEN and TIME. Questions are from [28], slot descriptions are from MultiWOZ 2.2 dataset [37], and

[§]Work done while at Amazon.

10.21437/Interspeech.2023-2238

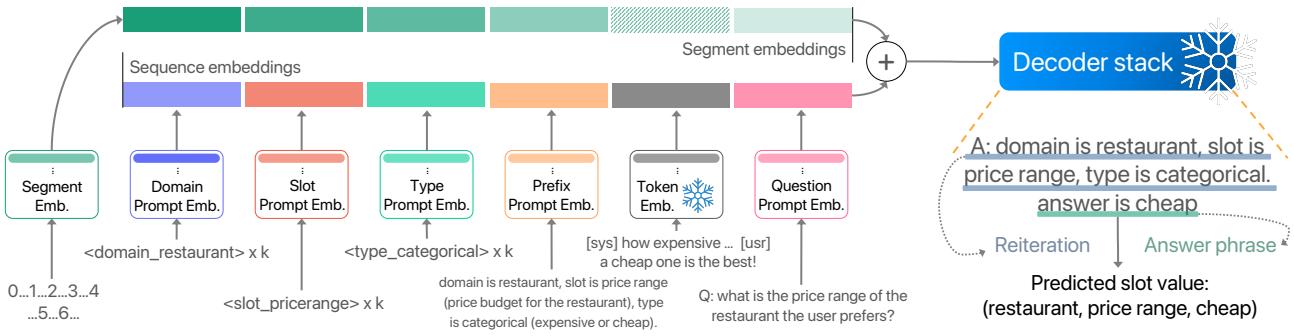

Figure 1: *Model design. The snow icon indicates non-trainable parameters. Absolute positional embeddings are added together with segment embeddings and sequence embeddings, we omit it for simplicity in the illustration.*

value candidates are from dialogue ontology.

2.2. Generative seq2seq framework

We use a decoder-only pre-trained language model (PLM) GPT-2 [38] as the backbone to provide language and commonsense knowledge, rather than an encoder-decoder model because of its superior performance [8]. To get a belief state at a certain turn, we create $|S|$ data instances to predict the slot value for each slot. Figure 1 demonstrates the design and a sample query.

Input sequence. We construct the input sequence by concatenating the following segments: 1) *Task prompt tokens for domain, slot and type*, each has k prompt tokens and they are shared among instances with the same domain, slot or type; 2) *Prefix*, a short sentence containing slot description, names of domain, slot, and type, and all possible candidates if the query slot is categorical; 3) *Dialogue history*, in which `[sys]` and `[usr]` tokens are used to indicate the speaker; and 4) *Question*, human-written question about the slot.

Target sequence and reiteration. We introduce the reiteration technique in the target sequence as shown in Figure 1 and generate task information before the answer phrase. We include the verbalized slot information as a "domain is `domain name`, slot is `slot name`, type is `type name`" phrase in the expected output sequence. By doing so, we require the model to optimize to remember the task information explicitly before generating the answer phrase, while using a consistent text generation cross-entropy loss. This technique allows the model to optimize upon both the answer and the sentence containing slot metadata, and explicitly learn the task information.

Segment embeddings. The input sequence contains segments with diverse formats and they are quite different from the format used in the pre-training phase of the LM. We divide the input sequence into segments, including five prompt segments, the system turns, the user turns and the answer segment. Tokens within a specific segment are assigned the same segment ID, and each segment ID maps to a unique segment embedding. Segment embeddings, which have the same length as the input sequence, are added with sequence embeddings and positional embeddings. We randomly initialize the embeddings of segment IDs and update them during training.

Training and inference. We pass the combined embeddings to the decoder stack to calculate the likelihood over the vocabulary. We use the cross-entropy loss with a regularization term to constrain the scale of prompt token embeddings following $L = CE + \lambda \|PE' - PE\|_2^2$ where λ is a weighting factor, and PE' and PE are updated and initialized prompt token embeddings [39]. Parameters of the PLM are frozen, and only prompt and segment embeddings are updated with Adam optimizer. During inference, we generate the output autoregressively with greedy decoding, and extract the answer with a rule-based function. For example, we extract predicted slot value "cheap" from free-form generation output "answer is cheap".

2.3. Soft prompt tokens

Prompt segments. We use two kinds of prompt tokens. *Task prompt tokens* are chosen according to the task's metadata, and used in the domain, slot and type prompt segments. *Word-mapping prompt tokens* are mapped from existing tokens in the prefix and question parts and used to replace normal tokens. In other words, task and word-mapping prompt tokens are shared across instances with the same task and instances using the same words respectively. We concatenate embeddings of each prompt segment (obtained by separate embedding matrices) with dialogue history embeddings (obtained by the frozen token embedding matrix) to form sequence embeddings.

Prompt initialization. To boost the performance in the low-resource setting, we use the pre-trained token embeddings to initialize the soft prompt token embeddings. The token embeddings from PLM are used to represent word semantics for language understanding, while the soft prompt tokens are used to represent task information initialized by task-related semantic meanings. We initialize a task prompt token by embedding of a randomly chosen token from its domain, slot or slot type name. Word-mapping prompt tokens are initialized with the embedding of the mapped word.

3. Experimental setup

Settings. We experiment on dialogues of five domains (*i.e.* attraction, hotel, restaurant, train, taxi) in MultiWOZ 2.0 [40] using the single-domain low-resource few-shot DST task. We take 5, 10, 20, 1%, 5% and 10% of training conversations of a particular domain to train, and evaluate on the full test set of the domain.

Evaluation metrics. Joint Goal Accuracy (JGA) represents the proportion of *turns* with *all* slots predicted correctly, and Slot Accuracy (SA) reflects the proportion of correct *slots*. If a slot is empty at a certain turn (for example, no related information is mentioned), the model needs to predict "none". A slot value is only correct if it matches exactly with the ground-truth value.

Implementation details. We use different prompt embeddings and learning rate schedules for the parameters of each prompt segment, meaning even if the same token appears in the prefix and question segments during initialization, it maps to different prompt embeddings for a larger optimization space. We

Table 1: *Overall performance. We report Joint Goal Accuracy (JGA, %), which is higher the better. We report the numbers from the paper (†), reproduction using author's codebase (‡), or our re-implementation (§). 271K is the average parameter count across domains. Detailed parameter counts are shown in Appendix A.1.*

Model	Params#	5	10	20	1%	5%	10%	5	10	20	1%	5%	10%	5	10	20	1%	5%	10%
		Attraction (3 slots, 1% = 27 conv.)						Hotel (10 slots, 1% = 33 conv.)						Restaurant (7 slots, 1% = 38 conv.)					
TRADE†		—	—	—	—	52.19	58.46	—	—	—	—	31.93	41.29	—	—	—	—	47.31	53.65
DSTQA†		—	—	—	—	51.58	61.77	—	—	—	—	33.08	**49.69**	—	—	—	—	35.33	54.27
T5DST‡	60M	4.77	21.93	30.57	40.68	52.12	60.13	8.19	13.46	17.94	18.63	38.76	46.13	13.80	19.51	22.79	29.47	**53.32**	58.44
Lee et al. [22] §	60M	6.33	19.12	34.53	37.56	54.34	58.75	9.31	15.76	22.07	24.41	**40.11**	42.98	15.87	19.66	22.15	30.96	48.94	**58.59**
Li et al. [8] §	335M	7.90	27.09	35.63	42.18	49.13	60.85	12.49	15.15	19.44	24.04	37.88	46.47	17.27	22.30	25.68	30.70	49.75	58.50
Ours	271K	**33.56**	**39.41**	**45.75**	**47.28**	**56.99**	**63.61**	**15.63**	**18.18**	**22.50**	**33.01**	38.24	45.60	**19.76**	**25.72**	**27.65**	**34.40**	50.81	55.79
		Taxi (4 slots, 1% = 15 conv.)						Train (6 slots, 1% = 29 conv.)						Average					
TRADE†		—	—	—	—	59.03	60.51	—	—	—	—	48.82	59.65	—	—	—	—	47.86	54.71
DSTQA†		—	—	—	—	58.25	59.35	—	—	—	—	50.36	61.28	—	—	—	—	45.72	57.27
T5DST‡	60M	48.22	53.74	58.27	58.19	59.23	69.03	12.31	21.93	36.45	43.93	69.27	69.48	17.46	26.11	33.20	38.18	54.54	60.64
Lee et al. [22] §	60M	45.32	49.93	58.58	58.52	60.77	**71.23**	13.57	25.02	38.52	50.26	69.32	69.72	18.08	25.90	35.17	40.34	54.70	60.25
Li et al. [8] §	335M	50.99	57.47	58.49	58.26	**61.68**	69.23	17.56	27.42	39.27	45.32	**71.69**	73.45	21.24	29.89	35.70	40.10	54.03	**61.70**
Ours	271K	**51.11**	**59.63**	**60.89**	**60.33**	61.63	63.00	**18.95**	**30.95**	**50.34**	**52.05**	69.51	**75.00**	**27.80**	**34.78**	**41.43**	**45.41**	**55.44**	60.60

use GPT-2 medium with 1024 hidden states as our backbone model with a maximum output length of 20. We choose the best epoch by monitoring the JGA of the development set. We report the averaged result for three runs with different random seeds for each experiment. All the models are trained on a single NVIDIA A6000 GPU on a Ubuntu 20.04.2 OS. The implementations of the transformer-based models are extended from the Huggingface codebase [41]. We use a 1e-3 learning rate.

Baseline models. We compare with the following works.[1] 1) TRADE [42]: GRU-based model with copy mechanism; 2) DSTQA [27]: QA-style model using ELMo representation; 3) T5DST [25]: T5-based generative model with slot type as prompt; 4) Lee et al. (2021) [22]: T5-based generative model with slot description and possible slot values as prompt; 5) Li et al. (2021) [8]: GPT-2 based QA-style generative model with manually created questions. The entire language model is updated for T5DST, Lee et al. and Li et al., and they represent the performance of prompt-based DST works. For Li et al., we use GPT-2 medium as the backbone PLM and do not use DSTC8 for transfer learning as it would introduce additional data resources and make the comparison not fair. For T5DST and Lee et al., we use the T5-small PLM with 60M parameters.

4. Experimental results

4.1. Overall results.

We show the overall few-shot experimental results in Table 1. Although our model uses only 0.08% and 0.45% of parameters compared with baselines, it still achieves higher JGA than all baseline models when using 1% or less training data across all domains. Especially we observe around 5, and 9 points JGA increases for the `attraction` and `hotel` domains compared with existing best models with 1% training data. In the `attraction` domain with 3 unique slots, our model trained using 5 dialogues performs on par with the previous best model using 20 dialogues. Our model shows its superiority especially when the amount of unique tasks is small. Using 5% and 10% data, our model performs comparably with existing best models with small gaps. Our model outperforms the frozen LM version of the baseline with even larger gaps as shown in Table 2.

[1]We are not comparing with prompt-based DST works that jointly train with other tasks for a fair comparison.

Table 2: *Comparison with the frozen LM variation of the baseline using 1% training data for each domain (JGA, %).*

Model	Attr.	Hotel	Rest.	Taxi	Train
Li et al. [8] (frozen LM)	29.16	14.81	15.14	47.56	35.77
Li et al. [8]	42.18	24.04	30.70	58.26	45.32
Ours	47.28	33.01	34.40	60.33	52.05

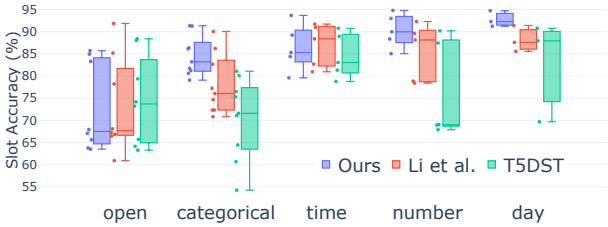

Figure 2: *Slot accuracy across slot types using 1% training data, each dot represents a unique slot.*

We demonstrate the performance of slots with different types across all five domains in Figure 2 compared with two generative baselines Li et al. [8] and T5DST [25]. We observe the worst performance in OPEN slots, which could be explained by the larger output candidate space. Breaking down slot type to more fine-grained type leads to a better result (considering DAY as a separate type rather than CATEGORICAL type, NUMBER and TIME as separate types rather than OPEN type). Compared with baselines, our model performs comparably on OPEN and TIME slots, but is more superior for CATEGORICAL, NUMBER and DAY slots.

We investigate the relationship between performance and the number of unique candidate answers (ontology size) using 1% target domain training data and Figure 3 demonstrates the result with trendlines created by expanding average algorithm for each model. We also show the performance of two generative baseline models for comparison. We observe that the

performance of all three models drops when the ontology size grows. For most ontology sizes, our model outperforms Li et al. [8] and T5DST [25].

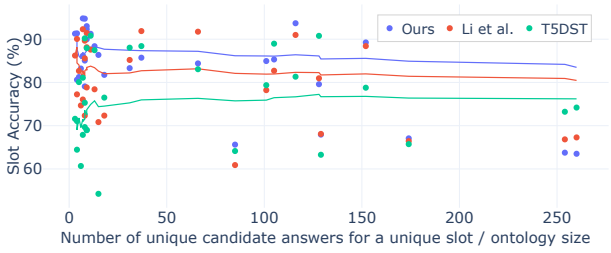

Figure 3: *Performance for slots with different ontology sizes*

4.2. Ablation study.

In Table 3, removing the slot segment (Line 2) leads to the largest performance drop among the three task prompt segments (L1-3), as slot is the most fine-grained task categorization. Prefix (L5) is more important than the question prompt (L4), which contains more metadata and parameters. The model without segment embedding (L6) has on average 7.8 points JGA drop, indicating the effectiveness of the segment embedding.

We further examine the effectiveness of the reiteration technique in Table 4. We observe a significant JGA drop without reiteration (11 points JGA drop for 10 training dialogues) especially when we have fewer training dialogues, which shows the helpfulness of including explicit task information in the learning objective. When there are limited training data, reiteration can help the model learn task boundaries among each slot faster and better. Note that even without reiteration, our model performs better than all baselines using 1% training data.

Table 3: *Ablation study using 1% training data (JGA, %).*

#	Model	Attr.	Hotel	Rest.	Taxi	Train	Avg
1	w/o domain	44.22	28.16	29.78	60.27	50.01	42.49
2	w/o slot	46.64	26.55	24.35	51.11	45.11	38.75
3	w/o type	45.30	25.26	33.65	59.89	51.91	43.20
4	w/o question	45.08	32.26	33.30	59.63	51.60	44.37
5	w/o prefix	42.98	28.78	31.54	57.72	47.00	41.60
6	w/o segment emb.	34.35	23.18	27.33	59.69	43.30	37.57
8	Full model	47.28	33.01	34.40	60.33	52.05	45.41

Table 4: *Ablation study for the reiteration technique (JGA, %).*

Few-shot	Model	Attr.	Hotel	Rest.	Taxi	Train	Avg
5	w/o reit.	22.16	12.09	16.67	47.68	4.97	20.71
	w/ reit.	33.56	15.63	19.76	51.11	18.95	27.80
10	w/o reit.	23.08	12.39	13.75	56.39	9.26	22.97
	w/ reit.	39.41	18.18	24.72	59.63	30.95	34.58
1%	w/o reit.	45.08	27.57	33.48	59.89	51.08	43.42
	w/ reit.	47.28	33.01	34.40	60.33	52.05	45.41

4.3. Error and qualitative analysis.

We categorize error cases as: 1) hallucination: predicting value for an empty slot; 2) omission: predicting "none" for a non-empty slot; 3) wrong value: predicting wrong real value for a non-empty slot [28]. Figure 4 shows the error distribution in terms of the proportion of each error category. The general OPEN slots (including TIME and NUMBER) have relatively more omission errors, while the general CATEGORICAL slots have relatively more hallucination errors. Our model is more conservative for OPEN slots compared with Li et al. [8].

Figure 4: *Error distribution across slot types*

We then investigate semantic information contained in the learned prompt tokens by selecting the most changed prompt tokens and producing the closest tokens with the smallest cosine similarity between the learned prompt token embedding and frozen token embeddings of the PLM. We show the result for the `attraction` domain in Table 5. The closest tokens are mostly variations or semantically similar tokens of the expected meanings of prompt tokens.

Table 5: *Closest tokens for the most changed prompt tokens in five prompt segments for the attraction domain.*

Prompt token	Closest tokens
<domain_attraction_4>	raction; ractions; racted
<slot_name_2>	name; Name; names
<type_open_3>	open; Open; opened
special	special; Special; statistical
Q	answer; Answer; answered

5. Conclusion and future work

We propose a parameter-efficient DST model using prompt tuning, and it represents tasks with soft prompt tokens with segment awareness and reiteration. Our model achieves state-of-the-art low-resource DST performance with less than 0.5% parameters compared with fine-tuning LM. We plan to further investigate the effects of prompt tuning on domain adaptation and prompt aggregation.

6. Acknowledgements

Many thanks to Sidi Lu, Tanmay Parekh, and Sarik Ghazarian for internal reviews, to members at Amazon Alexa AI, PLUS lab and UCLA-NLP for suggestions, and to the anonymous reviewers for their feedback.

A. Appendix

A.1. Detailed Parameter Count

Table 6: *The number of prompt tokens and parameters needed.*

	Attr.	Hotel	Rest.	Taxi	Train
Domain	5	20	20	10	10
Slot	15	200	140	40	60
Type	10	80	100	20	40
Question	20	46	36	19	27
Prefix	60	117	84	29	76
All prompt tokens #	110	463	380	118	213
Params #	120832	482304	397312	129024	226304

B. References

[1] Z. Wang and O. Lemon, "A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information," in *SIGDIAL*, Aug. 2013.

[2] F. Ye, J. Manotumruksa, Q. Zhang, S. Li, and E. Yilmaz, "Slot self-attentive dialogue state tracking," in *WWW*, 2021.

[3] L. Chen, B. Lv, C. Wang, S. Zhu, B. Tan, and K. Yu, "Schema-guided multi-domain dialogue state tracking with graph attention neural networks," in *AAAI*, 2020.

[4] S. Gao, A. Sethi, S. Agarwal, T. Chung, and D. Hakkani-Tur, "Dialog state tracking: A neural reading comprehension approach," in *SIGdial Meeting on Discourse and Dialogue*, Sep. 2019.

[5] Z. Lin, A. Madotto, G. I. Winata, and P. Fung, "MinTL: Minimalist transfer learning for task-oriented dialogue systems," in *EMNLP*, Nov. 2020.

[6] D. Ham, J.-G. Lee, Y. Jang, and K.-E. Kim, "End-to-end neural pipeline for goal-oriented dialogue systems using GPT-2," in *ACL*, Jul. 2020.

[7] E. Hosseini-Asl, B. McCann, C.-S. Wu, S. Yavuz, and R. Socher, "A simple language model for task-oriented dialogue," *Advances in Neural Information Processing Systems*, 2020.

[8] S. Li, J. Cao, M. Sridhar, H. Zhu, S.-W. Li, W. Hamza, and J. McAuley, "Zero-shot generalization in dialog state tracking through generative question answering," in *EACL*, Apr. 2021.

[9] B. Peng, C. Li, J. Li, S. Shayandeh, L. Liden, and J. Gao, "Soloist: Building task bots at scale with transfer learning and machine teaching," *TACL*, vol. 9, 2021.

[10] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *arXiv preprint arXiv:2107.13586*, 2021.

[11] M. D. Ma, X. Wang, P.-N. Kung, P. J. Brantingham, N. Peng, and W. Wang, "STAR: Boosting low-resource event extraction by structure-to-text data generation with large language models," *arXiv preprint arXiv:2305.15090*, May 2023.

[12] T. Gao, A. Fisch, and D. Chen, "Making pre-trained language models better few-shot learners," in *ACL-IJCNLP*, Aug. 2021.

[13] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," in *ACL-IJCNLP*, Aug. 2021.

[14] M. D. Ma, A. K. Taylor, W. Wang, and N. Peng, "DICE: Data-efficient clinical event extraction with generative models," in *ACL*, Jul. 2023.

[15] J. Xu, M. D. Ma, and M. Chen, "Can NLI provide proper indirect supervision for low-resource biomedical relation extraction?" in *ACL*, Jul. 2023.

[16] Y. Zhu, J. Feng, C. Zhao, M. Wang, and L. Li, "Counter-interference adapter for multilingual machine translation," in *Findings of ACL: EMNLP*, Nov. 2021.

[17] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *EMNLP*, Nov. 2021.

[18] Y. Gu, X. Han, Z. Liu, and M. Huang, "PPT: Pre-trained prompt tuning for few-shot learning," in *ACL*, May 2022.

[19] T. Vu, B. Lester, N. Constant, R. Al-Rfou', and D. Cer, "SPoT: Better frozen model adaptation through soft prompt transfer," in *ACL*, May 2022.

[20] S. Lee and R. Jha, "Zero-shot adaptive transfer for conversational language understanding," in *AAAI*, 2019.

[21] J. Zhao, M. Mahdieh, Y. Zhang, Y. Cao, and Y. Wu, "Effective sequence-to-sequence dialogue state tracking," in *EMNLP*, 2021.

[22] C.-H. Lee, H. Cheng, and M. Ostendorf, "Dialogue state tracking with a language model using schema-driven prompting," in *EMNLP*, Nov. 2021.

[23] Y. Su, L. Shu, E. Mansimov, A. Gupta, D. Cai, Y.-A. Lai, and Y. Zhang, "Multi-task pre-training for plug-and-play task-oriented dialogue system," in *ACL*, 2022.

[24] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan, "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset," in *AAAI*, 2020.

[25] Z. Lin, B. Liu, S. Moon, P. Crook, Z. Zhou, Z. Wang, Z. Yu, A. Madotto, E. Cho, and R. Subba, "Leveraging slot descriptions for zero-shot cross-domain dialogue StateTracking," in *NAACL*, Jun. 2021.

[26] R. Gupta, H. Lee, J. Zhao, Y. Cao, A. Rastogi, and Y. Wu, "Show, don't tell: Demonstrations outperform descriptions for schema-guided task-oriented dialogue," in *NAACL*, Jul. 2022.

[27] L. Zhou and K. Small, "Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering," *ArXiv*, vol. abs/1911.06192, 2019.

[28] S. Gao, S. Agarwal, D. Jin, T. Chung, and D. Hakkani-Tur, "From machine reading comprehension to dialogue state tracking: Bridging the gap," in *Workshop on NLP for Conversational AI*, 2020.

[29] Z. Lin, B. Liu, A. Madotto, S. Moon, Z. Zhou, P. Crook, Z. Wang, Z. Yu, E. Cho, R. Subba, and P. Fung, "Zero-shot dialogue state tracking via cross-task transfer," in *EMNLP*, Nov. 2021.

[30] Y. Wu, H. Wang, D. Zhang, G. Chen, and H. Zhang, "Incorporating instructional prompts into a unified generative framework for joint multiple intent detection and slot filling," in *Proceedings of the 29th International Conference on Computational Linguistics*. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 7203–7208. [Online]. Available: https://aclanthology.org/2022.coling-1.631

[31] C. Clarke, J. Peper, K. Krishnamurthy, W. Talamonti, K. Leach, W. Lasecki, Y. Kang, L. Tang, and J. Mars, "One agent to rule them all: Towards multi-agent conversational AI," in *Findings of ACL: ACL*, May 2022.

[32] A. Sauer, S. Asaadi, and F. Küch, "Knowledge distillation meets few-shot learning: An approach for few-shot intent classification within and across domains," in *Workshop on NLP for Conversational AI*, 2022.

[33] A. Maronikolakis and H. Schütze, "Multidomain pretrained language models for green NLP," in *Workshop on Domain Adaptation for NLP*, 2021.

[34] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," in *ACL*, Jul. 2019.

[35] A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres, "Quantifying the carbon emissions of machine learning," *arXiv preprint arXiv:1910.09700*, 2019.

[36] Q. Zhu, B. Li, F. Mi, X. Zhu, and M. Huang, "Continual prompt tuning for dialog state tracking," in *ACL*, May 2022.

[37] X. Zang, A. Rastogi, S. Sunkara, R. Gupta, J. Zhang, and J. Chen, "MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines," in *Workshop on NLP for Conversational AI*, 2020.

[38] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[39] T. Müller, G. Pérez-Torró, and M. Franco-Salvador, "Few-shot learning with Siamese networks and label tuning," in *ACL*, 2022.

[40] P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gašić, "MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling," in *EMNLP*, 2018.

[41] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-art natural language processing," in *EMNLP: Demos*, 2020.

[42] C.-S. Wu, A. Madotto, E. Hosseini-Asl, C. Xiong, R. Socher, and P. Fung, "Transferable multi-domain state generator for task-oriented dialogue systems," in *ACL*, Jul. 2019.