Prompter: Zero-shot Adaptive Prefixes for Dialogue State Tracking Domain Adaptation

Taha Aksu[†‡*], **Min-Yen Kan**[†], **Nancy F. Chen**[‡]
† National University of Singapore (NUS), Singapore
‡ Institute for Infocomm Research (I^2R), A*STAR, Singapore
*taksu@u.nus.edu

Abstract

A challenge in the Dialogue State Tracking (DST) field is adapting models to new domains without using any supervised data — zero-shot domain adaptation. Parameter-Efficient Transfer Learning (PETL) has the potential to address this problem due to its robustness. However, it has yet to be applied to the zero-shot scenarios, as it is not clear how to apply it unsupervisedly.

Our method, Prompter, uses descriptions of target domain slots to generate dynamic prefixes that are concatenated to the key and values at each layer's self-attention mechanism. This allows for the use of prefix-tuning in zero-shot. Prompter outperforms previous methods on both the MultiWOZ and SGD benchmarks. In generating prefixes, our analyses find that Prompter not only utilizes the semantics of slot descriptions but also how often the slots appear together in conversation. Moreover, Prompter's gains are due to its improved ability to distinguish "none"-valued dialogue slots, compared against baselines.

1 Introduction

Task-oriented dialogue (TOD) systems serve users through several tasks, such as booking a table in a restaurant or suggesting tourist attractions. One crucial component of these systems, Dialogue State Tracking (DST), is responsible for extracting users' preferences (*i.e.* slot-values) over key attributes (*i.e.* slot-labels) of their service (Wu et al., 2019). DST has a significant role in TOD systems as it ensures that both the action taken in the back-end and the responses returned to the users are aligned with the preferences that the users indicate.

A challenging task in this field is to adapt an existing DST model to a new domain it has not seen before without using any supervised data, *i.e.* in the zero-shot scenario. This is important, as in many new scenarios, it is hard to collect data, let alone annotate it. Yet it is still an essential need for a TOD system to appropriately answer such queries in new contexts. The challenge arises from the differences in dialogue context, slot values, and slot labels among different domains. For example, a model could be trained on the 'taxi-booking' domain and thus capable of extracting the destination for a taxi; but when deployed to the 'train-booking' domain, the range of slot-values changes, resulting in a higher probability of a mistaken inference. We show an example (Figure 1), where due to the superficial connections a baseline T5 model forms, it

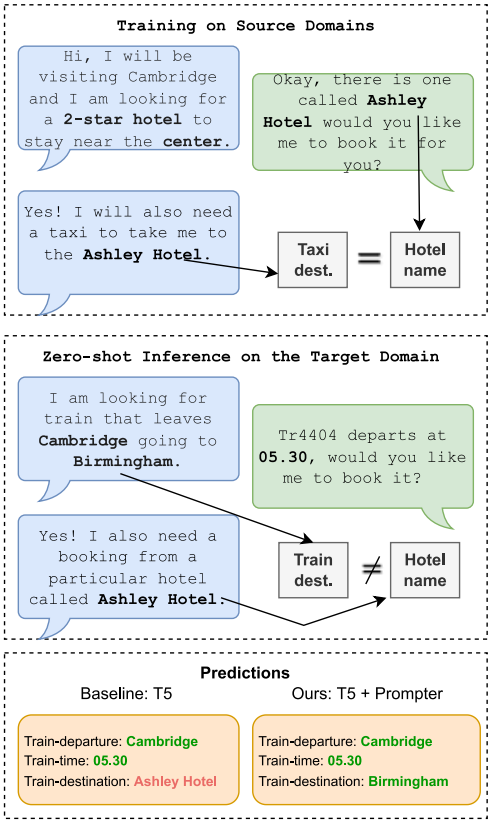


Figure 1: Zero-shot domain adaptation. The model is trained on four source domains and tested on the train-booking domain without any supervised training. Bottom-left: T5 baseline predictions, Bottom-right: Prompter predictions. (Correct, incorrect) predictions are colored (green, red), respectively.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4588–4603

incorrectly predicts 'Ashley Hotel' as the train destination (bottom left). In many dialogue contexts, a large number of slots are unspecified. These are known as "none"-valued slots. In cases where the model is adapting to a new domain without any prior training, it often incorrectly predicts none values. This makes it even more important to address the problem of domain shift.

Lin et al. (2021b) proposed to address this domain shift challenge via the language model's intrinsic ability to reason over prompts. Specifically, they concatenate the description of each slot as a hard prompt into the dialogue context and then generate the answers using the T5 model. While it does well for a naive baseline, it makes mistakes due to its superficial understanding of slot labels.

Meanwhile, another line of study has shown that Parameter-efficient Transfer Learning (PETL) methods are effective training methods to address domain shift. Due to the small number of parameters it introduces per task/instance, it overcomes overfitting in few-shot scenarios, outperforming earlier baselines. There have been various attempts to use these methods for DST tasks within a few-shot, continual learning setting (Zhu et al., 2022; Madotto et al., 2021). However, a significant barrier to adopting PETL is that such methods cannot be directly applied in zero-shot, as they all require some form of supervised training.

In this study, we propose a new method to use prefix-tuning under a zero-shot scenario to benefit from the gains it brings for robustness, even without supervised data. Rather than fine-tuning the prefixes during training, we add a new mechanism into the T5 architecture called Prompter[1]. Prompter simply takes the description of the slot and then generates the prefixes on the fly. We then append these prefixes at each layer of the encoder to represent the dialogue from the perspective of the subject slot label. This method makes minimal changes to LM parameters while generating unsupervised prefixes. This ensures both the preservation of general-purpose traits and extrapolation to new domains.

We conduct experiments with the MultiWOZ 2.1 and SGD datasets. Prompter improves average JGA results across domains by 1.7 for MultiWOZ, and 9.1 points for the SGD dataset (considering 4 domains reported in prior studies) compared to the strongest baseline. This shows that PETL methods' robustness advantage is also favorable for unsupervised domain adaptation scenarios. To the best of our knowledge, these are the highest results achieved so far using a small language model. Through further analysis, we have discovered that Prompter not only considers the semantic similarities of slot descriptions but also the frequencies in which slots co-appear in the dialogue context. Furthermore, Prompter proves to be more effective in identifying slots that have no value within a conversation in comparison to previous methods.

[1]Implementation available at https://github.com/cuthalionn/Prompter

2 Related Work

Dialogue State Tracking. DST has a long history of models working with a static, ontology-based problem definition (*i.e.* slot-values are fixed) (Balaraman et al., 2021). The static-ontology DST is a simplified classification problem where the model selects a value from each slot's value pool. (Zhang et al., 2020; Lee et al., 2019; Rastogi et al., 2017; Zhong et al., 2018). Recently interest in *dynamic* ontologies have received attention, adding flexibility at inference time (Wu et al., 2019; Rastogi et al., 2019; Heck et al., 2020).

Low-resource Domain Adaptation. Dynamic ontology introduces slot-value level flexibility, but its ability to work with new slot-labels is limited. Domain adaptation of DST systems aims to make the model adaptable to new domains/slot-labels. Few studies have attempted to utilize language models' intrinsic reasoning abilities by mapping DST as a question–answering task (Lin et al., 2020; Zhou and Small, 2019). Shin et al. (2022), on the other hand, map DST to a dialogue summarization task, and Xie et al. (2022) map it to a structured-knowledge grounding task. Many use data augmentation to address the lack of supervision in the target domain (Qiu et al., 2022; Mi et al., 2021; Gritta et al., 2021; Aksu et al., 2022; Li et al., 2020). Finally, remaining studies focus on improving the model's architecture and training strategies for robustness toward domain changes. (Feng et al., 2022; Balaraman and Magnini, 2020; Madotto and Liu, 2020; Huang et al., 2020; Coope et al., 2020; Wu et al., 2019; Lei et al., 2018; Lin et al., 2021b; Yang et al., 2022). Wang et al. (2022) have a similar goal as our own, but they use a different method. They create cross-slot dependency by combining multiple slot prompts to create a final prompt, which encourages the model to apply what it has learned

in one slot to other slots.

PETL for DST Domain Adaptation. Parameter Efficient Transfer Learning (PETL) is a recently trending set of methods that aims to adapt models more efficiently by significantly reducing the number of parameters that need to be fine-tuned. (Pfeiffer et al., 2020; Lester et al., 2021; Liu et al., 2022; Li and Liang, 2021; Houlsby et al., 2019). Many studies have found that PETL is advantageous for low-resource domain adaptation settings due to its efficient parameter training scheme. This scheme minimizes changes in LM parameters and thus believed to prevent over-fitting (Li and Liang, 2021; Liu et al., 2022). However, He et al. (2022) argues that tuning the entire language model does not negatively impact its robustness advantage. Researchers in the DST field have also utilized PETL methods for their robust capabilities. In their work, Zhu et al. (2022) employed soft prompts and fine-tuned them for each domain in a continual learning setting, utilizing validation sets from target domains to decide which previous prompts to use for initialization. Madotto et al. (2021) also tackled the problem of continual learning, using unique adapters for each domain and relying on a classifier to select which adapter to use during inference. Both studies only explored the use of PETL methods for DST with few-shot availability. In contrast, this study aims to investigate a well-known PETL method, prefix-tuning (Li and Liang, 2021), for zero-shot domain adaptation of DST models.

3 Background

3.1 Dialogue State Tracking Task

A task-oriented dialogue consists of a number of consecutive system and user utterances, together referred to as a turn, $t_i = (s_i, u_i)$. Each turn is annotated with a belief state that shows the user's preferences over a number of attributes from various domains up to and including that turn, $B_i = (D_0, D_1, ..., D_K)$ where D_j is the belief state for domain j, and K is the total number of domains. The belief state for each domain is made up of a list of slot-label (*e.g.* 'restaurant-area') and slot-value pairs (*e.g.* 'center'), $D_j = \{s_0 : v_0, s_1 : v_1, ..., s_N : v_N\}$, where N is the number of slots within domain j. Each s_i is further annotated with a description that explains the attribute in the context of the domain (*e.g.* 'restaurant-area':'The area of the city where the restaurant is located.'). For each v_i, if s_i is not discussed in the dialogue context, v_i is set to 'none'. Otherwise, v_i is a sequence of tokens. The task of DST is to predict the belief state B_i for a given dialogue context DC, *i.e.* dialogue turn history up to and including turn i, $DC = (t_0, t_1, ..., t_i)$.

3.2 Prefix-Tuning

Prefix-tuning is a parameter-efficient alternative to fine-tuning which optimizes a small continuous task-specific vector called the prefix for each new task. These tunable prefix vectors are prepended to the keys and values of the multi-head attention at every layer of the transformer (Li and Liang, 2021; He et al., 2021). Li and Liang (2021) also report that prefix-tuning also improves extrapolation to unseen tasks in few-shot settings. However, there is no straightforward way to use this method for the zero-shot setting, as it requires supervision to fine-tune the prefixes.

4 Method

We propose to add a new mechanism into the T5 architecture (Raffel et al., 2019), called Prompter, to take advantage of prefix-tuning's extrapolation capabilities without requiring supervision. Instead of fine-tuning the prefixes with source domain data, we generate them on the fly for each slot. However, we need a way to condition Prompter for a new domain without any supervised data. Task-oriented dialogue schemas provide a solution by annotating the slot descriptions for each slot-label. Using these slot descriptions Prompter can generate domain-specific prefixes which allow it to adapt to any domain without the need for supervised data. We can summarize the Prompter pipeline in three key parts: (1) Slot Prompt Generation, (2) Prefix Generation, and (3) Multi-head Self Attention.

Slot Prompt Generation. is responsible for generating a prompt that is specific to each slot, using its unique description. Previous approaches to this problem, such as simply concatenating the description to the input, result in only a superficial understanding of the slots in zero-shot settings (Lin et al., 2021b). Additionally, using slot embeddings as soft prompts can cause unstable training and hinder zero-shot adaptation due to changes in the descriptions. Instead, we propose using a global prompt that is modified according to each slot's description. This modification is applied through a cross-attention mechanism that attends the global

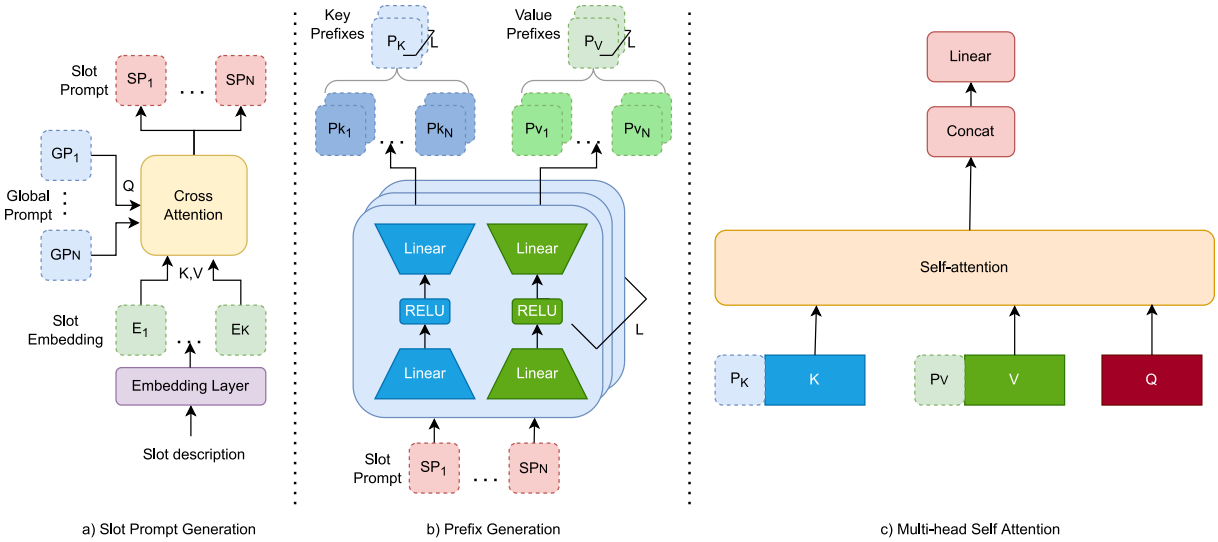


Figure 2: The architecture of our proposed method, Prompter. Prompter leverages the prefix-tuning method to enable zero-shot learning without the need for supervised data and it is composed of three parts: (a) Slot Prompt Generation where the information from the description is fused with some global prompt to generate slot-specific prompts, (b) Prefix Generation which feeds slot prompts across two linear layers and an activation function to generate per-layer key and value prefixes, (c) Finally these prefixes are concatenated to keys and values at every layer of the T5 encoder.

prompt to the slot description's embedding, *c.f.* Figure 2a. This approach ensures that each slot prompt shares the same initialization addressing unstable training, and the modifications reflect changes in the slot-label addressing domain shift. It also has the advantage of making the final prompt's length fixed, regardless of the length of the description. The slot prompt is calculated as follows:

$$S = ((GW_q)(EW_k)^\top)(EW_v) \quad (1)$$

where W_q,W_k, and $W_v \in \mathbb{R}^{d\times d}$ are query, key, and value weights for the cross attention mechanism and d is the model dimension, $G \in \mathbb{R}^{N\times d}$ is the global prompt[2], $E \in \mathbb{R}^{K\times d}$ is the slot embedding, K is the length of slot description, and $S \in \mathbb{R}^{N\times d}$ is the slot prompt.

Prefix generation. For the DST task, the dialogue context can make up the majority of the language model input (*i.e.* 100–400 tokens long dialogue context compared to 10–15 tokens long slot description), this results in challenges with the prompt-tuning method because the prompt's impact can vanish easily before the decoding starts. This is why we opt for prefix-tuning because it ingests prompts at each layer and thus the generated value will have higher exposure to the prompt.

[2]For N we try different values from [1,100] range and empirically found 10 to work best. Thus we set N=10 throughout conducted experiments.

So following the generation of slot prompts the next step is to generate key and value prefixes for each layer. For this step, we have tried several different architectural designs such as a simple MLP or a whole transformer block. We empirically observed that while the former lags behind due to the small number of parameters the latter results in overfitting. Thus, inspired by He et al. (2022) we use a sequence of down and up projections separated by an activation function as prefix generators, *c.f.* Figure2b. Note that each transformer layer has a pair of dedicated prefix generators to generate key and value prefixes:

$$K_i = RELU(SWk_{down_i})Wk_{up_i} \quad (2)$$

$$V_i = RELU(SWv_{down_i})Wv_{up_i} \quad (3)$$

where K_i, and $V_i \in \mathbb{R}^{N\times d}$ are key and value prefixes for the i^{th} layer; $Wk_{down_i}, Wv_{down_i} \in \mathbb{R}^{d\times r}$, Wk_{up_i} and $Wv_{up_i} \in \mathbb{R}^{r\times d}$ are the respective down and up projectors for the i^{th} layer; r is the bottleneck dimension. r is set to $d/4$ throughout our experiments.

Multi-head Self Attention. After we get K_i and V_i for each layer i we split them to N_h head vectors K_i^j and $V_i^j \in \mathbb{R}^{N\times d_h}$ for each head j, where $d_h = d/N_h$ is the dimension per head. Finally, we concatenate these key and value prefixes into the self-attention mechanism at each layer of the

transformer encoder completing our modifications to the original T5 architecture, *c.f.* Figure 2c.

$$head_i^j = (h_i W_{q_i}^j [K_i^j, h_i W_{k_i}^j]^\top)[V_i^j, h_i W_{v_i}^j] \quad (4)$$

where $head_i^j$ is the output from the j^{th} head of self-attention mechanism at layer i; $W_{q_i}^j$, $W_{k_i}^j$, and $W_{v_i}^j \in \mathbb{R}^{d \times d_h}$ are query, key and value weight matrices of the j^{th} head in the ith layer; and h_i is the input to the i^{th} layer. The final output of the multi-head self-attention at layer i is calculated as:

$$MSA(h, i) = [head_i^0, head_i^1, ..., head_i^{N_h}] W_{o_i} \quad (5)$$

where $W_{o_i} \in \mathbb{R}^{d \times d}$.

5 Experimental Setup

5.1 Datasets

We conduct experiments with two well-known DST benchmarks: MultiWOZ and SGD (Budzianowski et al., 2018; Rastogi et al., 2019). MultiWOZ is a task-oriented dialogue dataset collected in a wizard of oz setting using human speakers. It has 10k dialogues that span over 7 domains. It provides turn-level annotations and descriptions of each slot label. In line with previous studies, we limited our experiments to only 5 domains because the police and hospital domains do not have a sufficient number of examples in the test set. We use MultiWOZ version 2.1 which addresses the noisy state annotations within the original dataset (Eric et al., 2020). Similar to MultiWOZ, the SGD dataset also has turn-level annotations and descriptions, *i.e.* schema, for each domain and slot. It has over 20k annotated conversations between a human and a virtual assistant. These span over 20 domains. Besides, the SGD dataset has unseen domains in the test set specifically formed to evaluate zero-shot performance.

5.2 Baseline Models

We compare our method with a range of DST models from the past as well as the recent state of the art. The only models we utilize that do not depend on a language model are **TRADE** (Wu et al., 2019) and **MA-DST** (Kumar et al., 2020). The former introduces the copy mechanism to ease predicting slots not seen during training, whereas the latter adds cross-attention to model relationships between the context and slots at different semantic levels and self-attention to resolve cross-domain coreferences to a base RNN layer. **SUMBT** by Lee et al. (2019) is built with BERT and again uses an attention mechanism to learn relations between domains and slots. **SGD-baseline** (Rastogi et al., 2019) feeds slots, domains, and value embeddings into a BERT encoder to create schema embedding and uses it to predict dialog state in the target domain under zero-shot. **Seq2seq-DU** (Feng et al., 2021) formalizes DST as a sequence-to-sequence task where the dialog history is transformed directly into semantic frames. Li et al. (2021) on the other hand use GPT-2 and define DST as a generative question-answering approach. **TransferQA** builds on a similar motivation but combines both extractive and multi-choice QA enabling tracking categorical and non-categorical slots simultaneously (Lin et al., 2021a). **T5DST** (Lin et al., 2021b) and **Wang et al. (2022)** both use the T5 architecture. The former concatenates slot descriptions with dialogue context and generates slot values in an auto-regressive manner. Whereas the latter proposes a unique design that models cross-slot dependency by composing multiple slots as the final prompt so that the model is forced to learn the relations among each slot.

5.3 Training Details

For all experiments, we used a Tesla-V100 GPU. We use the small-sized PPTOD (Su et al., 2022) built on the T5 architecture for the T5DST baseline and our own Prompter. We empirically found PPTOD to be more suitable for prompt-tuning tasks most probably due to the nature of its pretraining tasks. We set the batch size to 8 with gradient accumulation every 8 steps. We use AdamW optimizer (Loshchilov and Hutter, 2017) for training and set the initial learning rate to $1e-4$.

Semi-frozen Training Scheme Contrary to what is typically recommended for limited data scenarios by traditional PETL techniques, we discovered that freezing LM parameters does not improve performance in the zero-shot scenario. This is in line with what He et al. (2022) suggests. However, we also find that tuning all parameters is imperfect. In search for a better strategy we experiment with different combinations of frozen layers and compare the results for zero-shot train domain performance. We found that the best strategy is a semi-frozen (S.F.) training scheme, where all LM parameters are trained for 1k steps and then all layers of the T5 model are frozen except the first and last layers of the encoder and decoder (*c.f.* Appendix B for

Model	Lang. Model	Attraction	Hotel	Restaurant	Taxi	Train	Avg
TRADE	-	20.06	14.20	12.59	59.21	22.39	25.69
MA-DST	-	22.46	16.28	13.56	59.27	22.76	26.87
SUMBT	BERT-b	22.60	19.08	16.50	59.50	22.50	28.18
Li et al.	GPT2	23.67	18.54	21.05	59.1	24.34	29.34
T5DST	T5-s	31.92	**20.72**	20.09	64.12	28.83	33.56
Wang et al.	T5-s	33.92	18.85	20.75	66.25	36.96	35.55
T5DST*	PPTOD-s	$35.5_{\pm 1.7}$	$20_{\pm 0.9}$	$25.3_{\pm 0.8}$	$65.6_{\pm 0.6}$	$35.3_{\pm 1.0}$	$36.4_{\pm 6.9}$
Prompter*	PPTOD-s	$\mathbf{35.8}_{\pm 0.7}$	$19.2_{\pm 0.8}$	$\mathbf{26}_{\pm 0.7}$	$\mathbf{66.3}_{\pm 0.2}$	$\mathbf{39}_{\pm 0.5}$	$\mathbf{37.2}_{\pm 7}$

Table 1: Zero-shot join-goal accuracy(%) results on MultiWOZ 2.1 dataset. Results for all baselines are reported from original papers. Models with * trained using the semi-frozen training scheme. For our trained models the results are averaged over three runs. The best results on each column are **bold**.

JGA	Buses	Messaging	Trains	Payment	Media	Events	Unseen
SGD-baseline	9.7	10.2	13.6	11.5	18.0	23.5	-
Seq2seq-DU	16.8	4.9	16.8	7.2	-	-	-
Transfer-QA	15.9	13.3	17.4	**24.7**	-	-	-
Wang et al.	43.9	36.6	46.7	16.5	-	-	-
T5DST*	$46.8_{\pm 2.2}$	$54_{\pm 2.8}$	$\mathbf{53}_{\pm 0.4}$	$23.3_{\pm 3.8}$	$55.5_{\pm 3.3}$	$48.8_{\pm 2.5}$	$48.0_{\pm 0.8}$
Prompter*	$\mathbf{48.4}_{\pm 2.1}$	$\mathbf{59.2}_{\pm 1.3}$	$50.8_{\pm 0.9}$	$21.9_{\pm 4.6}$	$\mathbf{65.3}_{\pm 3.8}$	$\mathbf{51.5}_{\pm 0.4}$	$\mathbf{49.4}_{\pm 0.4}$

Table 2: Zero-shot joint-goal accuracy (%) results on SGD dataset. Results for all baselines are reported from original papers. Models with * trained using the semi-frozen training scheme. For our trained models the results are averaged over three runs. The final column shows the average JGA on all unseen slots. The best results on each column are **bold**.

more details). Thus for the experiments conducted in this section, we employ this strategy to train the models.

5.4 Evaluation

We evaluate the performance of all models using Joint Goal Accuracy (JGA) following prior studies. For MultiWOZ, a zero-shot setting is used where training occurs on four domains and the remaining domain is used for testing. For SGD, results are reported on domains that are not included in both the training and validation sets, as they have already been included in the PPTOD pretraining. We modified the official SGD evaluation script to reflect this change. Therefore, in our evaluation settings, unseen domains refer only to domains in the test data, contrary to the original definition by Rastogi et al. (2019) which considers domains only showing up in the validation data unseen as well.

6 Results and Analysis

In MultiWOZ (Table 1), our addition of Prompter shows improvements in all domains except Hotel, boosting the average JGA by 1.7 points, compared to the state-of-the-art model by Wang et al. (2022). We believe the lack of improvements in the hotel domain for Prompter is due to it having many unique slots (*i.e.* 'hotel-internet', 'hotel-parking', 'hotel-type', *etc.*). This makes it harder to take advantage of earlier domains as they lack similar slots. This is also in line with the results from Wang et al. (2022), as their cross-slot dependency design also lags behind for hotel domain results.

We also present the results on the SGD dataset in Table 2, where Prompter shows improvements on average. We share results over 6 representative domains along with results for official unseen domain performance. Once more, Prompter demonstrates superior performance on average in unfamiliar domains. Compared to the results reported in the original paper by Wang et al. (2022) for four domains (Columns 1 through 4 of Table Table 2), Prompter shows an average improvement of 9.1 in JGA. The Alarm domain is excluded from the comparison as PPTOD has been pretrained on it.

Model	Train	Rest	Hotel	Taxi	Attr
T5DST	28.83	20.09	20.72	64.12	31.92
+ S.F.	29.3	24.4	**22.3**	65.6	34.76
+ PPTOD	35.3	25.3	20	65.6	35.5
+ Prompter	**39**	**26**	19.2	**66.3**	**35.8**

Table 3: Ablation results on the test set of MultiWOZ 2.1. We cumulatively add semi-frozen (S.F.) training, PPTOD, and Prompter to the T5DST baseline and report results. The best results along each column are **bold**.

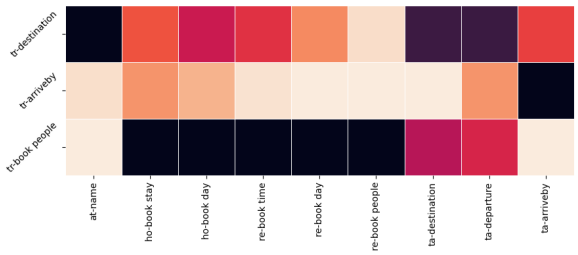
6.1 Ablation Study

We further conducted ablation to analyze the contribution of Prompter's components (Table 3). Adding the S.F. training scheme (second row) to the T5DST baseline introduces performance increase across all domains. This demonstrates that this training scheme plays a significant role in the robustness of the model. If we switch the pre-trained model from T5 to PPTOD (third row), we see another round of improvement but it is inconsistent across domains. Finally, it is evident from the final row that adding the Prompter increases the results by another margin, clearly showing its contribution.

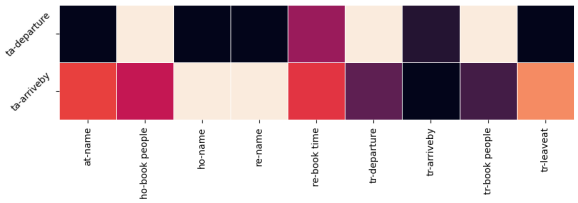
6.2 Fine Grained Analysis

How does Prompter improve results? We define two new metrics to better understand Prompter's improvements: *Miss-prediction* (MP), where the model fails to correctly identify a gold slot-label, mistakenly labeling it as 'none' instead; and *Over-prediction* (OP), where the model incorrectly predicts a 'none' valued slot-label as something else. We then combine these metrics in *None Accuracy*, a metric that measures the accuracy of the model's predictions regarding the "activeness" of a slot-label. In other words, it measures how often the model correctly predicts whether a slot-label has the value 'none' or not. The results over all 5 domains can be found in Table 4. It is evident that Prompter's improvement comes from the None accuracy measure as its results are in line with the change in JGA (*i.e.* improvements across all domains except the Hotel domain). Moreover, we find that this is mostly due to the reduction of over-prediction mistakes — Prompter decreases this class of error in every domain.

How does Prompter connect slots? To better understand the benefits of using Prompter, we look at how it connects target domain slots with source domain slots. This is done by aggregating the key



(a) Source domain slots in close proximity to 'Train-destination', 'Train-arriveby', and 'Train-bookpeople' slots, according to generated prefix similarities.



(b) Source domain slots close to 'Taxi-departure' and 'Taxi-arriveby' slots, according to generated prefix similarities.

Figure 3: Heatmaps depicting the similarity of selected source and target domain slots. The generated prefixes are aggregated and compared with cosine similarity, where darker colors indicate higher similarity.

prefixes across each layer and attention head for every slot and then comparing them to the source domain slot prefixes from the training set using cosine similarity.

Figure 3 highlights important similarities among some of the taxi and train domain slots (*c.f.* Appendix A for a comprehensive version that includes all domains and slots). Figure 3a shows that 'train-destination' has a high similarity with 'taxi-departure' and 'destination', as well as the 'attraction-name' slots. The first two connections are expected, but the latter is also relevant because the 'attraction-name' often appears as the 'taxi-destination' in training. This indicates that the model finds that the 'destination' slots can often contain named entities (such as locations) within the dialogue. For 'train-arriveby', the most similar slot is also the semantically closest: 'taxi-arriveby'. Finally, for the 'train-bookpeople' slot, the most similar slots are those related to booking from the hotel and restaurant domains, which makes sense as these often co-occur in the training data.

Figure 3b shows the results of adapting in the taxi domain. The similarity between the 'taxi-arriveby' slot and its train domain counterpart, 'train-arriveby', is high as expected. Moreover, for the 'taxi-departure' slot, the generated prefixes are most similar to slots for attraction, restaurant,

	Attraction			Hotel			Restaurant			Taxi			Train		
	MP↓	OP↓	None Acc↑	MP↓	OP↓	None Acc↑	MP↓	OP↓	None Acc↑	MP↓	OP↓	None Acc↑	MP↓	OP↓	None Acc↑
T5DST	76.58	13.78	65.96	**59.51**	24.03	**76.30**	35.98	14.25	79.15	**53.57**	10.96	83.94	**3.93**	12.81	90.91
T5DST+ Prompter	**69.80**	**11.16**	**68.81**	65.77	**21.66**	75.34	**29.14**	**13.40**	**82.07**	54.40	**8.92**	**84.05**	7.66	**9.08**	**91.50**

Table 4: Fine Grained Analysis over MultiWOZ 2.1 dataset. MP and OP stand for miss-prediction and over-prediction respectively. Down arrow (↓) means lower the better, whereas up arrow (↑) means higher the better. The best results among each column are **bold**.

Dial	Conversation Details	
1	U0	I am looking for a train that leaves on Wednesday, going to Bishops Stortford.
	S1	tr4404 departs **Cambridge** at 05:29 and arrives at 06:07 in Bishops Stortford. Would you like me to book it?
	U1	Thats pretty early. Would there be a later train that arrives by 15:00?
	S2	tr3844 departs Cambridge Wednesday at 13:29 and arrives in Bishops Stortford by 14:07. Would you like to reserve a seat?
	U2	No. How long will the train take?
	S3	That train ride will take approximately 38 minutes.
	U3	Thanks. I also need a particular hotel. Its name is **Ashley hotel**.
	GT	{train-destination: Bishops Stortford, train-day: Wednesday, train-arriveby: 15:00, train-departure: **Cambridge**}
	T5DST	{train-destination: Bishops Stortford, train-day: Wednesday, train-departure: **Ashley hotel**, train-arriveby: 15:00}
	T5DST+ Prompter	{train-destination: Bishops Stortford, train-day: Wednesday, train-arriveby: 15:00, train-departure: **Cambridge**}
2	U0	I am coming to Cambridge and would like to see some **architecture**. Do you have any located in the **centre**?
	S1	Yes, there are 5 places located in the centre. I recommend the All Saint Church on Jesus Lane.
	U1	Thanks! What is the entrance fee?
	S2	...
	GT	{}
	T5DST	{hotel-type: **architecture**, hotel-area: **centre**}
	T5DST+ Prompter	{}
3	U0	Hello, I am looking for places to go in the centre?
	S1	There are many attractions in the centre like museums, architecture, boating, and concert halls. What are you interested in?
	U1	How about a boating attraction?
	S2	There are 2 in the centre of town. Scudamores punting co., and the cambridge punter. Would either of those interest you?
	U2	Could you give me the address for the Cambridge punter, please? I also need a place to stay, preferably somewhere **cheap**.
	GT	{hotel-pricerange: **cheap**}
	T5DST	{hotel-pricerange: cheap}
	T5DST+ Prompter	{hotel-pricerange: **cheap**, hotel-type: **cheap**, hotel-internet: **cheap**}

Table 5: Three example dialogues from the MultiWOZ 2.1 test set. Each dialogue consists of user and system turns, ground truth dialogue state (GT). We show a pair of predictions by the T5DST baseline, and our Prompter.

and hotel names. This is likely because the 'train-departure' slot also has named entities as values. The findings show that Prompter not only utilizes slots with similar descriptions to create prefixes, but also accounts for other slots that co-occur in the same conversation with a similar source slot. This is important as slots may have different descriptions but exhibit significant semantic overlap (e.g., 'taxi-departure' and 'hotel-name' having location named entities as values).

6.3 Case study

We use three dialogues from the MultiWOZ test set to demonstrate some of the phenomena observed in previous analysis studies (Table 5). The first example shows how the T5DST baseline is susceptible to overgeneralization from training data. When the T5DST model encounters a hotel name during zero-shot inference on the train domain, it mistakenly assumes that the hotel is the departure for the train because it has been trained to associate location names with taxi departure/destination. Prompter avoids this mistake through its deeper understanding of cross-slot relations. In the second case, the model has made predictions for the hotel type and area even though the dialogue does not mention a hotel. This happens because the model has learned to predict the same type of slots for the attraction domain and has overfitted them during training. In contrast, Prompter ameliorates this form of over-prediction (§6.2).

Our model has a weakness when it comes to dealing with slots that are unique and do not have similar slots in the source domain. In the third case, the model struggles to accurately predict the 'hotel-type' and 'hotel-internet' slots because they

	T5DST*	T5DST* + Prompt-tuning
Attraction	31.68	31.68
Hotel	18.51	15.4
Restaurant	19.66	18.23
Taxi	64.77	64.71
Train	33.5	35.4

Table 6: Zero-shot joint-goal accuracy (%) results on MultiWOZ 2.1 dataset comparing T5DST baseline with prompt tuning version of our approach. Both models are trained using T5-small model and semi-frozen training scheme. The results are averaged over three runs.

are dissimilar to all slots in the source domain.

6.4 Why Prefix-Tuning?

We also try implementing Prompter using soft prompt-tuning rather than prefix-tuning. Under this setting, the learned prompts are fed directly at the input layer instead of as prefixes to the attention mechanism at each layer. We compare the performance of this method with the baseline T5DST, using T5-small as the language model. We find that prompt-tuning is not even comparable to the fine-tuning baseline let alone to prefix-tuning, *c.f.* Table 6. We believe this difference is due to the fact that prompts fed in the initial layer of the transformer have a diminishing effect on the output of the decoder. This is also evident in the original prefix-tuning paper where Li and Liang (2021) claim it performs better compared to prompt-tuning when it comes to generation tasks.

7 Conclusion

Parameter Efficient Transfer Learning methods have been frequently used for their strong robust features under a low-resource setting. However, there is no straightforward way to take advantage of these features under a zero-shot setting because they require at least some supervised data during adaptation. The dialogue state tracking (DST) task, on the other hand, has just the right annotation for this scenario as it contains schema annotations with slot label descriptions. We propose Prompter, which uses these descriptions to enable prefix-tuning, a well-known PETL method, for use under a zero-shot domain adaptation setting.

We show through experiments that this method improves the JGA metric for the two most common DST benchmarks. We further explain through analyses and a case study that the reason behind the Prompter's power is two-fold. (1) It has better capability to distinguish 'none' valued slots within the dialogue and (2) it can digest the frequency of slots co-occurrences within the dialogue context into the prefix generation process. We believe that this study shows PETL's hidden potential for DST domain adaptation under a zero-shot setting.

8 Acknowledgements

This research was supported by the SINGA scholarship from A*STAR. We would like to thank anonymous reviewers for their insightful feedback on how to improve the paper.

9 Limitations

One limitation of our study is that we only evaluated our method on the T5 architecture. Further experiments on other architectures could be useful to determine the generalizability of our findings. Additionally, as in previous SOTA, our model also did not produce better results for the hotel domain, even though it did improve performance in general. We have attempted to explain why this domain is more difficult, but more research is needed to fully understand the reasons for this variability and to create methods that can improve performance across all domains.

References

Ibrahim Aksu, Zhengyuan Liu, Min-Yen Kan, and Nancy Chen. 2022. N-shot learning for augmenting task-oriented dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1659–1671, Dublin, Ireland. Association for Computational Linguistics.

Vevake Balaraman and Bernardo Magnini. 2020. Domain-aware dialogue state tracker for multi-domain dialogue systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:866–873.

Vevake Balaraman, Seyedmostafa Sheikhalishahi, and Bernardo Magnini. 2021. Recent neural methods on dialogue state tracking for task-oriented dialogue systems: A survey. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 239–251, Singapore and Online. Association for Computational Linguistics.

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, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Samuel Coope, Tyler Farghly, Daniela Gerz, Ivan Vulić, and Matthew Henderson. 2020. Span-ConveRT: Few-shot span extraction for dialog with pretrained conversational representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 107–121, Online. Association for Computational Linguistics.

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

Yue Feng, Aldo Lipani, Fanghua Ye, Qiang Zhang, and Emine Yilmaz. 2022. Dynamic schema graph fusion network for multi-domain dialogue state tracking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 115–126, Dublin, Ireland. Association for Computational Linguistics.

Yue Feng, Yang Wang, and Hang Li. 2021. A sequence-to-sequence approach to dialogue state tracking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1714–1725, Online. Association for Computational Linguistics.

Milan Gritta, Gerasimos Lampouras, and Ignacio Iacobacci. 2021. Conversation graph: Data augmentation, training, and evaluation for non-deterministic dialogue management. *Transactions of the Association for Computational Linguistics*, 9:36–52.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *ArXiv*, abs/2110.04366.

Yun He, Huaixiu Zheng, Yi Tay, Jai Gupta, Yu Du, V. Aribandi, Zhe Zhao, Yaguang Li, Zhaoji Chen, Donald Metzler, Heng-Tze Cheng, and Ed H. Chi. 2022. Hyperprompt: Prompt-based task-conditioning of transformers. *ArXiv*, abs/2203.00759.

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.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*.

Yi Huang, Junlan Feng, Min Hu, Xiaoting Wu, Xiaoyu Du, and Shuo Ma. 2020. Meta-reinforced multi-domain state generator for dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7109–7118, Online. Association for Computational Linguistics.

Adarsh Kumar, Peter Ku, Anuj Kumar Goyal, Angeliki Metallinou, and Dilek Z. Hakkani-Tür. 2020. Ma-dst: Multi-attention based scalable dialog state tracking. In *AAAI Conference on Artificial Intelligence*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483, Florence, Italy. Association for Computational Linguistics.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447, Melbourne, Australia. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Fatema Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2020. Coco: Controllable counterfactuals for evaluating dialogue state trackers. *CoRR*, abs/2010.12850.

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.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 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. 2021a. 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.

Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Andrea Madotto, Eunjoon Cho, and Rajen Subba. 2021b. Leveraging slot descriptions for zero-shot cross-domain dialogue StateTracking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5640–5648, Online. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. MinTL: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3391–3405, Online. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7452–7467, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Andrea Madotto and Zihan Liu. 2020. Language models as few-shot learner for task-oriented dialogue systems. *ArXiv*, abs/2008.06239.

Fei Mi, Wanhao Zhou, Lingjing Kong, Fengyu Cai, Minlie Huang, and Boi Faltings. 2021. Self-training improves pre-training for few-shot learning in task-oriented dialog systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1887–1898, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Liang Qiu, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. Structure extraction in task-oriented dialogues with slot clustering. *ArXiv*, abs/2203.00073.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Abhinav Rastogi, Dilek Z. Hakkani-Tür, and Larry P. Heck. 2017. Scalable multi-domain dialogue state tracking. *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 561–568.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *AAAI Conference on Artificial Intelligence*.

Jamin Shin, Hangyeol Yu, Hyeongdon Moon, Andrea Madotto, and Juneyoung Park. 2022. Dialogue summaries as dialogue states (DS2), template-guided summarization for few-shot dialogue state tracking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3824–3846, Dublin, Ireland. Association for Computational Linguistics.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676, Dublin, Ireland. Association for Computational Linguistics.

Qingyue Wang, Yanan Cao, Piji Li, Yanhe Fu, Zheng Lin, and Li Guo. 2022. Slot dependency modeling for zero-shot cross-domain dialogue state tracking. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 510–520, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

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.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle,

Ansong Ni, Ziyu Yao, Dragomir R. Radev, Caiming Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *ArXiv*, abs/2201.05966.

Yuting Yang, Wenqiang Lei, Juan Cao, Jintao Li, and Tat-Seng Chua. 2022. Prompt learning for few-shot dialogue state tracking. *ArXiv*, abs/2201.05780.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, Philip Yu, Richard Socher, and Caiming Xiong. 2020. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 154–167, Barcelona, Spain (Online). Association for Computational Linguistics.

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.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *ArXiv*, abs/1911.06192.

Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. 2022. Continual prompt tuning for dialog state tracking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1124–1137, Dublin, Ireland. Association for Computational Linguistics.

A Prefix Heatmaps

Figures 4 to 8 depict cosine similarity heatmaps between target and source domain slot prefixes for every domain in Multiwoz 2.1 dataset.

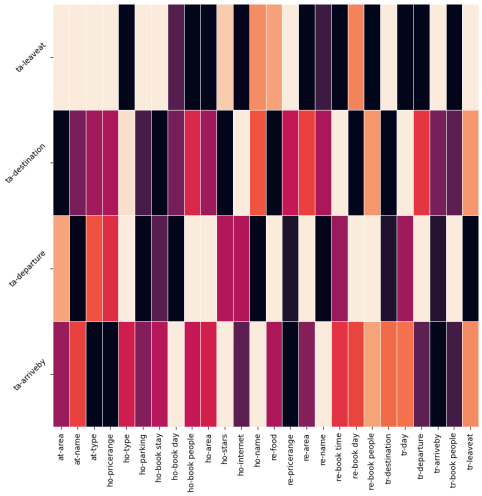


Figure 4: Heatmap for Taxi domain slots.

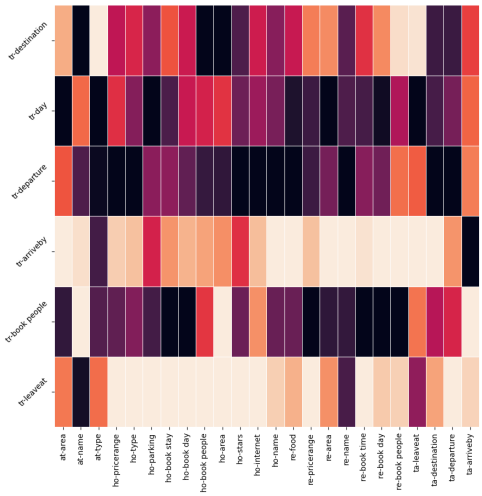


Figure 5: Heatmap for Train domain slots.

B Semi-Frozen Training

After discovering that completely freezing the parameters of the Language Model (LM) does not lead to improved performance in zero-shot adaptation, we conducted a series of initial experiments to determine the most effective configuration. These preliminary experiments focused on the train domain of MultiWOZ 2.1. Each experiment involved training all parameters for 1,000 steps, which consistently showed benefits. We then selectively froze layers, with the specific layers varying for each row in Table 7. For example, in the first row, we froze

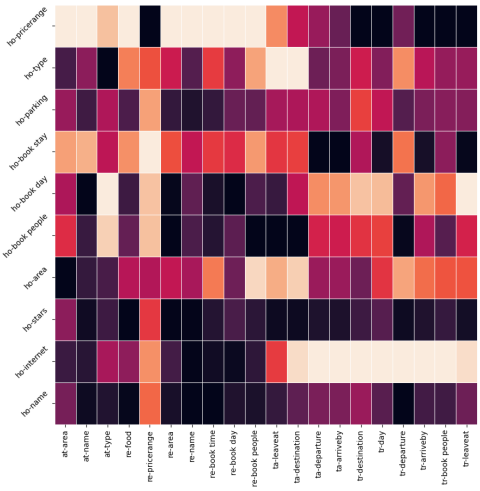


Figure 6: Heatmap for Hotel domain slots.

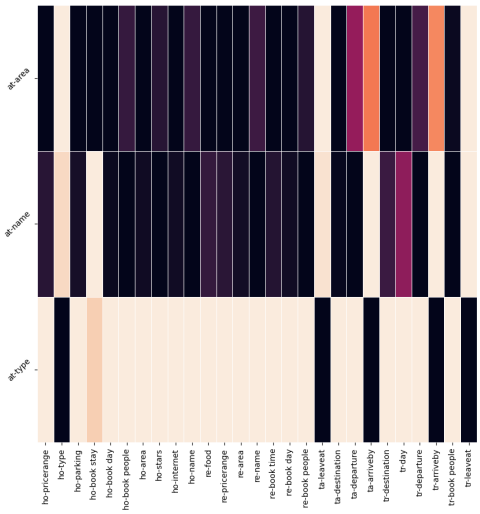


Figure 7: Heatmap for Attraction domain slots.

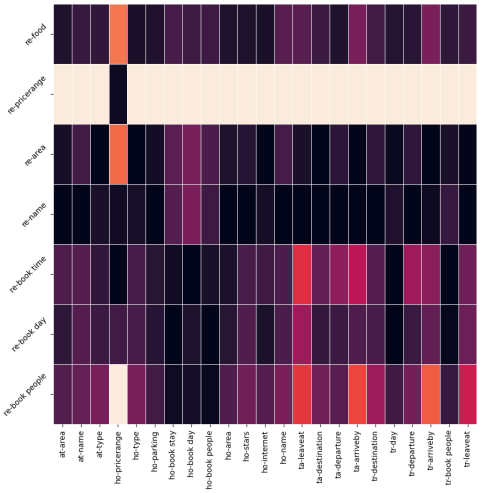


Figure 8: Heatmap for Restaurant domain slots.

Setting	JGA
Unfreeze all up to 2nd layer	37.9
Unfreeze all up to 3rd layer	37.7
Unfreeze all up to 4th layer	38.8
Unfreeze all up to 5th layer	34.5
Unfreeze all up to 6th layer	39.1
Unfreeze the first and last layers (ours)	**39.7**
Unfreeze the first two and last two layers	30.2

Table 7: Zero-shot joint-goal accuracy (%) results on MultiWOZ 2.1 dataset, train domain using Prompter. Each row uses a different configuration of the semi-frozen training scheme.

all layers except the first layer of the encoder and decoder after the initial 1,000 steps. Our findings revealed that the optimal approach is to freeze all layers except the first and last layers of both the encoder and decoder after 1,000 steps.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations (Section 8)

☒ A2. Did you discuss any potential risks of your work?
It is unlikely that research on zero-shot dialogue state tracking poses any foreseeable risks.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Introduction (Section 1). Both summarize the contribution and findings that we describe throughout the paper.

☑ A4. Have you used AI writing assistants when working on this paper?
We used ChatGPT to assist us purely with the language of the paper. Specifically, we used it to make certain discussions in the Introduction (Section 1) and Results and Analysis (Section 6) clearer and more succinct. In doing so, we never used the model to generate sentences from scratch. Instead, we used a drafted sentence and asked the assisting model to paraphrase it such that it will be easier to read and/or more succinct. Moreover, for each generated sentence we further edited it to address the errors and keep the flow with the rest of the paragraph intact.

B ☑ Did you use or create scientific artifacts?

Related Work (Section 2), Method (Section 3).

☑ B1. Did you cite the creators of artifacts you used?
Related Work (Section 2), Method (Section 3).

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Experiment and Analysis (Sections 6). We reuse the codebase and artifacts from Lin et al. (2021) and include their license in our codebase, which is uploaded within the submission and will be published upon acceptance.

☒ 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)?
The original codebase was licensed by Attribution-NonCommercial-ShareAlike 4.0 International. Which requires its users to give appropriate credit, not use it for any commercial purposes and share any new changes made. Our use case is in line with all three and we believe there is no need for further discussion within the paper.

☒ 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?
We use two well-known DST benchmarks which are collected in an anonymized environment through a WOZ setting. Their collection process is publicly available in their original manuscripts.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Both benchmarks are well studied and there are already publicly available manuscripts that document these artifacts.

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

☒ 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.
Both benchmarks are well studied and there are already publicly available manuscripts that document these statistics.

C ☑ Did you run computational experiments?

Results and Analysis (Section 6).

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Training Details (Section 5.3)

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Training Details (Section 5.3)

☑ 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?
We report mean results averaged over a number of runs. This is explained in the captions of both main experiment tables. Results and Analysis (Section 6).

☑ 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.)?
Evaluation (Section 5.4)

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.