Zero-shot Cross-domain Dialogue State Tracking via Context-aware Auto-prompting and Instruction-following Contrastive Decoding

Xiaoyu Dong*, **Yujie Feng***, **Zexin Lu, Guangyuan Shi, Xiao-Ming Wu**†
Department of Computing, The Hong Kong Polytechnic University, Hong Kong S.A.R.
xiaoyu.dong@connect.polyu.hk, xiao-ming.wu@polyu.edu.hk

Abstract

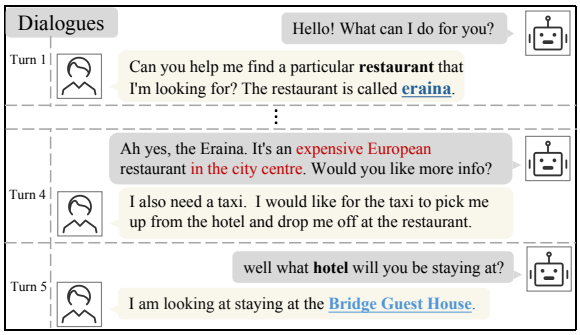
Zero-shot cross-domain dialogue state tracking (DST) enables us to manage task-oriented dialogues in new, unseen domains without the cost of collecting in-domain data. Previous studies have implemented slot-based input improvements, such as schema-driven descriptions and question-answering formats, but still suffer from negative transfer for seen slots and inefficient transfer for unseen slots due to the significant source-target domain gap. To address these issues, we introduce a novel framework called Context-aware Auto-prompting and Instruction-following Contrastive Decoding (CAPID). This framework generates dynamic, context-aware slot queries, effectively improving the model's transferability. Our context-aware auto-prompting approach tailors slot queries to the current dialogue context, increasing flexibility and reducing ambiguities. Additionally, an instruction-following contrastive decoding strategy helps reduce errors related to off-topic slots by penalizing deviations from the provided instructions. Extensive experiments on two datasets, with varying model sizes (from 60M to 7B), demonstrate the superior performance of CAPID. The source code[1] is provided for reproducibility.

1 Introduction

Task-oriented dialogue systems are powerful tools for assisting users in accomplishing a wide range of tasks, e.g. booking a train or making restaurant reservations (Huang et al., 2020). Dialogue State Tracking (DST), a crucial component of these systems, is responsible for identifying domain-slot pairs (e.g., <Restaurant-Name>) and extracting the corresponding values (e.g., "Eraina") from the conversation (Peng et al., 2020; Lin et al., 2020).

In industrial applications, these systems frequently need to incorporate new domains based on user requirements. However, collecting extensive data for each new domain is both costly and inefficient. Consequently, there is a substantial need for **zero-shot cross-domain DST**, which requires a DST model to adapt to entirely new target domains without retraining, leveraging knowledge already acquired in previous source domains (Lin et al., 2021b). This transferability is essential for developing flexible and scalable dialogue systems that can operate effectively across diverse domains without exhaustive, domain-specific training.

With the impressive performance demonstrated by large-scale pretrained neural language models



Value of the slot <Restaurant-Name> for different slot formats	
Conventional Pre-defined Slot:	<Restaurant-Name>
Schema-driven Prompting:	name of the restaurant
Question-answering:	What is the name of the restaurant?
Their result:	Bridge Guest House ✗
Context-aware Slot Query:	Given the conversation, what is the name of the specific expensive European restaurant the user is looking for?
Our Result:	eraina ✓

Figure 1: An example illustrating negative transfer in Zero-Shot Cross-Domain DST. After training on source domains—hotel, taxi, and others—the model is tested on the restaurant domain to evaluate its cross-domain transfer ability. Previous methods using static slot descriptions erroneously assign the value "Bridge Guest House," which belongs to the previously trained "Hotel" domain, to the <Restaurant-Name> slot. Conversely, our context-aware slot query approach allows the model to accurately identify "eraina" as the correct value.

* Equal contribution
† Corresponding author.
[1] https://github.com/dong7313/CAPID_

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 8527–8540
November 12-16, 2024

(LMs), recent studies have focused on using slot-related descriptions as input to enhance slot understanding (Lee et al., 2021; Feng et al., 2024b; Aksu et al., 2023). For example, in order to mitigate the confusion arising from abbreviated slots (e.g., "ArriveBy," which does not clearly indicate whether it refers to a time or a method of arrival), Lin et al. (2021b) used schema-driven prompting by incorporating natural language schema descriptions, resulting in remarkable performance improvement. Additionally, Lin et al. (2021a) and Zhou et al. (2022a) proposed to convert conventional slot descriptions into question-answering formats, further enhancing clarity and understanding. Building on these recent advancements, our investigation of current works on zero-shot cross-domain DST (see Section 2 for an in-depth analysis) has uncovered the following previously unreported challenge:

Challenge 1: Negative transfer for seen slots (Tavares et al., 2023). This occurs when a slot in the target domain has already been encountered in the previously trained source domain. As illustrated in Figure 1, the <Name> slot seen in the "Hotel" domain during training. When tracking its value in the target "Restaurant" domain, previous methods erroneously apply learned patterns from previously trained domains, resulting in incorrect responses.

Challenge 2: Inefficient transfer for unseen slots. This arises when the model encounters a slot that has never been seen in the source domain. Due to the significant domain gap, the model struggles to transfer existing knowledge effectively.

These challenges arise because of the fixed descriptions for each slot used by traditional methods, which are collected by crowd-sourced human annotators and might be inconsistent across different domains. Moreover, these slot-based enhancements lead the model to learn fixed slot-value pairs, lacking the contextual diversity and nuanced understanding achieved from a broader contextual perspective. For instance, contexts such as "expensive" and "European" in Figure 1 would help the model better differentiate between domains.

To address these challenges, we propose Context-aware Auto-prompting and Instruction-following Contrastive Decoding (CAPID), a ***context-aware*** slot query generation approach through auto-prompting to align the gap between source and target domains. Rather than generating prompts solely related to slots, our method is the first to dynamically creates customized prompts for each sample by also incorporating contextual information from the dialogue. The detailed comparisons between our method and the previous DST methods can be shown in Table 1.

We utilize auto-prompting capability of GPT-4 (Achiam et al., 2023) to generate context-aware slots by incorporating dialogue contexts and domain-specific attributes. What's more, when training with these context-aware slots changing dynamically with the dialogue, the model accumulate the knowledge of analyzing dialogue contexts to extract slot values, which is more tranferrable to unseen slots.

While GPT-4 demonstrates remarkable capabilities, its usage limits and data privacy issues necessitate a different strategy for large-scale slot query generation. We address this by training a smaller student model to emulate GPT-4's context-aware slot query generation. Furthermore, to ensure the student model adheres to instructions when handling off-topic slots, we developed a novel instruction-following contrastive decoding method. This method maximizes the difference between expert log-probabilities from the original prompt and amateur log-probabilities from the repurposed prompt, restricting the search space to high-probability tokens that are semantically consistent with the original prompt, thereby producing coherent results.

To summarize, our main contributions include:

- We propose CAPID, a novel framework for zero-shot cross-domain DST, which effectively addresses the challenges of negative transfer for seen slots and inefficient transfer for unseen slots.
- We conduct comprehensive experiments to evaluate CAPID, demonstrating its remarkable capability to improve the performance of zero-shot cross-domain DST. Specifically, CAPID achieves a 16.8% increase JGA on MultiWOZ 2.1 and a 29.34% increase on MultiWOZ 2.4 compared to the previous baseline.

2 Motivation

We categorize slots into two types: unseen slots, which lack counterparts in the source domains (e.g., "<Hotel_Stay>" vs. "<Restaurant_Food>"), and seen slots, which exhibit similarities across domains (e.g., "<Hotel_Name>" and "<Restaurant_Name").

Method	Slot Prompting Approach	Dynamic Prompt	Context-aware	Example
LDST, FNCTOD	Pre-defined Slot	✗	✗	<Restaurant-Name>
Prompter, D3ST, T5DST	Schema-driven Prompting	✗	✗	name of the restaurant
TransferQA	Question-answering	✗	✗	What is the name of the restaurant?
CAPID (ours)	Context-aware Slot Prompting	✓	✓	What is the name of the specific expensive European restaurant the user is looking for?

Table 1: Comparisons between previous methods and CAPID.

2.1 Minimizing Ineffective Knowledge Transfer for Unseen Slots

Conventional pre-defined slot often lead the model to focus primarily on learning slot-value pairs. This type of knowledge, while effective within known domains, proves difficult to transfer when the system encounters slots that were unseen during training. The limited ability of these conventional systems to adapt to new or varied dialogue contexts often results in ineffective knowledge transfer, manifesting as frequent "None" responses.

To tackle this, we employ context-aware slots that enable the model to learn how to extract slot values based on varying dialogue contexts. This approach enhances the model's knowledge to analyze dialogues, whichs is more adaptable and transferrable to unseen slots in new domains. Additionally, it increases the similarity between slots and the dialogue, which also helps to prevent the common issue of "None" responses in unseen slots. For example, as illustrated in our study shown in Figure 2, we tailor slot terms to better align with user-specific language. Unlike previous methods that rigidly apply the term "hotel" across different dialogues, our context-aware approach dynamically adapts, using terms like "guesthouse" or "hotel" depending on the specific context. Additionally, it captures varied user expressions such as "book a table" or "in Cambridge" instead of merely requesting the restaurant's name.

2.2 Addressing Negative Transfer for Seen Slots

Negative transfer occurs when a model trained in one domain inappropriately applies learned patterns to a different, unseen domain, leading to errors in understanding and processing. This phenomenon is particularly prevalent with seen slots in trained domains in zero-shot learning. For example, as is shown in Figure 1, a DST system trained on "hotel_name" might mistakenly identify "restaurant_name" as "hotel_name" in the restaurant domain, assuming similarities between the domains due to its training. As depicted in Figure 3, the significant impact of negative transfer is highlighted,

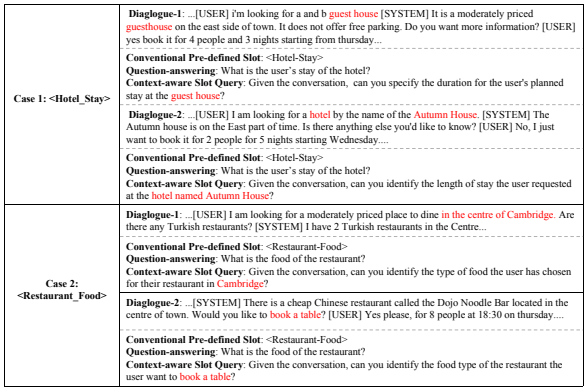


Figure 2: Context-aware slots help minimize ineffective knowledge transfer for unseen slots, e.g. "<Hotel_Stay>" vs "<Restaurant_Food>". The two cases show how to integrate useful dialogue context with slots, where previous methods keep slot descriptions fixed across various dialogues.

with 88.31% of errors attributed to this issue. Negative transfer ratio is calculated by dividing the number of incorrect predictions where slot values are predicted as those seen during training by the total number of incorrect predictions.

Our context-aware slot representation addresses this challenge by incorporating adjectives and terminologies specific to the current domain, e.g. "expensive" and "European", thus strengthening the slot descriptions. This adaptation enables the model to better differentiate between domains and accurately interpret user intents, significantly reducing the risk of negative transfer and enhancing accuracy across multiple domains.

3 Proposed Method: CAPID

Problem Formulation A task-oriented dialogue consists of a sequence of utterances alternating between the assistant and user, $\{A_1, U_1, ..., A_t, U_t\}$, where A and U represent the assistant and user utterances, respectively. A predefined slot set[2] $\mathcal{S} = \{S^j\}_{j=1}^{J}$ is provided, where J is the total number of slots. Given the dialogue context $C = \{(A_1, U_1), (A_2, U_2) \ldots, (A_t, U_t)\}$, the task

[2]To specify the domain to which a slot belongs, a slot is defined as the concatenation of the specific domain and the slot name, e.g., "<Restaurant_Area>".

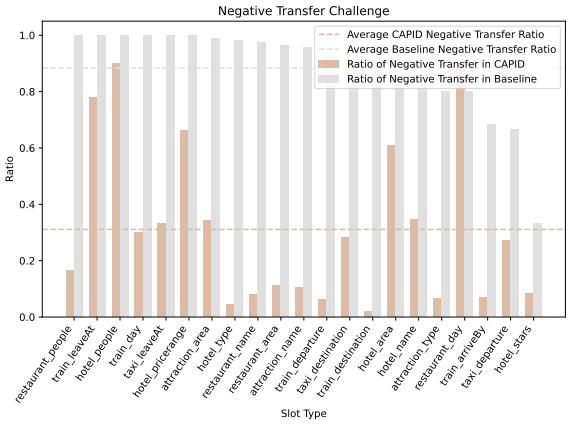


Figure 3: Mitigation of negative transfer shown by decreased average negative transfer ratio in dashed lines. This trend is particularly pronounced in seen slots, notably those ending in "name".

of DST is to predict the dialogue state (S_t^j, V_t^j) at each turn t, where (S_t^j, V_t^j) is a set of slot-value pair. Our CAPID framework involves training two separate models. The first model is tasked with training a student model to learn a context-aware slot query function, represented as $f_s : S_t^{'j} = \phi(C_t, S_t^j)$. The model is first trained on source domains D_{source} using dialogue contexts C_t and the context-aware slot $S_t^{'j}$ generated by GPT-4, then apply the trained model f to a new, unseen domain D_{target} to perform inference. The second model involves training a DST model, denoted as $f_d : \mathcal{C}_t \oplus S_t^{'j} \rightarrow V_t^j$ to predict the slot values based on the enhanced slot descriptions provided by f_s, where $\oplus$ denotes simple text concatenation.

Overview Our CAPID framework leverages LLMs to generate context-aware slots using an auto-prompting strategy. These regenerated slots are then structured into datasets of paired prompt instructions to facilitate instruction-based training for smaller student models. The student model is then deployed for large-scale slot generation with the innovative instruction-follwing contrastive decoding, as detailed in Figure 4.

3.1 Context-aware Auto-prompting for Slot Query Generation via LLMs

We utilize GPT-4 (Achiam et al., 2023) to learn the function $\phi(C_t, S_t^j)$ through auto-prompting, which aims to transform the conventional slots into new questions by enhancing three key features. The first is replacing conventional slots with synonyms found within the dialogue. The second is integrating contextual information related to the slot value, including domain-specific adjectives and user-described needs, to formulate a new question. The third is enhancing the fluency of the new question.

Specifically, we use a "crossover" operation that cross the dialogue with the slot to generate a context-aware slot query, fulfilling the second step. This operation, adapted from general genetic algorithms (Schmitt, 2004), swaps segments between two prompts and acts as a global search mechanism to broaden the search scope. Detailed instructions are in Appendix A. After this process, we have the new context-aware slot query $S^{'}$.

3.2 Training a Student Model for Context-aware Auto-prompting

Despite GPT-4's impressive performance, its closed-source nature and request restrictions are significant limitations. Therefore, we train a student model for large-scale $S^{'}$ generation. We instruct the student model to generate questions faithful to the pre-defined slots and to incorporate additional dialogue information into the questions. Training involves instruction tuning on paired prompts, each consisting of a pre-defined slot and its corresponding context-aware slot.

However, in DST tasks, slots often contain domains not present in the dialogue, which we refer to as off-topic slots. This mismatch may cause the student model to incorrectly follow instructions by substituting the domain of the predefined slots with those appearing in the dialogue.

3.3 Instruction-following Contrastive Decoding for Handling Off-topic Slots

To address the mismatch domain challenge in 3.2, we propose the Instruction-following Contrastive Decoding, as illustrated in Figure 5. This innovative approach ensures that the student model generates slot queries which adhere to the original domain specified in the slots, mitigating the risk of deviating from our two primary instructions.

Inspired by contrastive decoding(CD) (Li et al., 2023), our method adapts this concept to suit slot-verified generation process. We also observe diminished confidence in the model's outputs under scenarios where the slot domain mismatches the dialogue domain. Specifically, the model often assigns lower probabilities to output tokens, with only a slight discrepancy between erroneous and correct tokens. Therefore we use an "repurposed

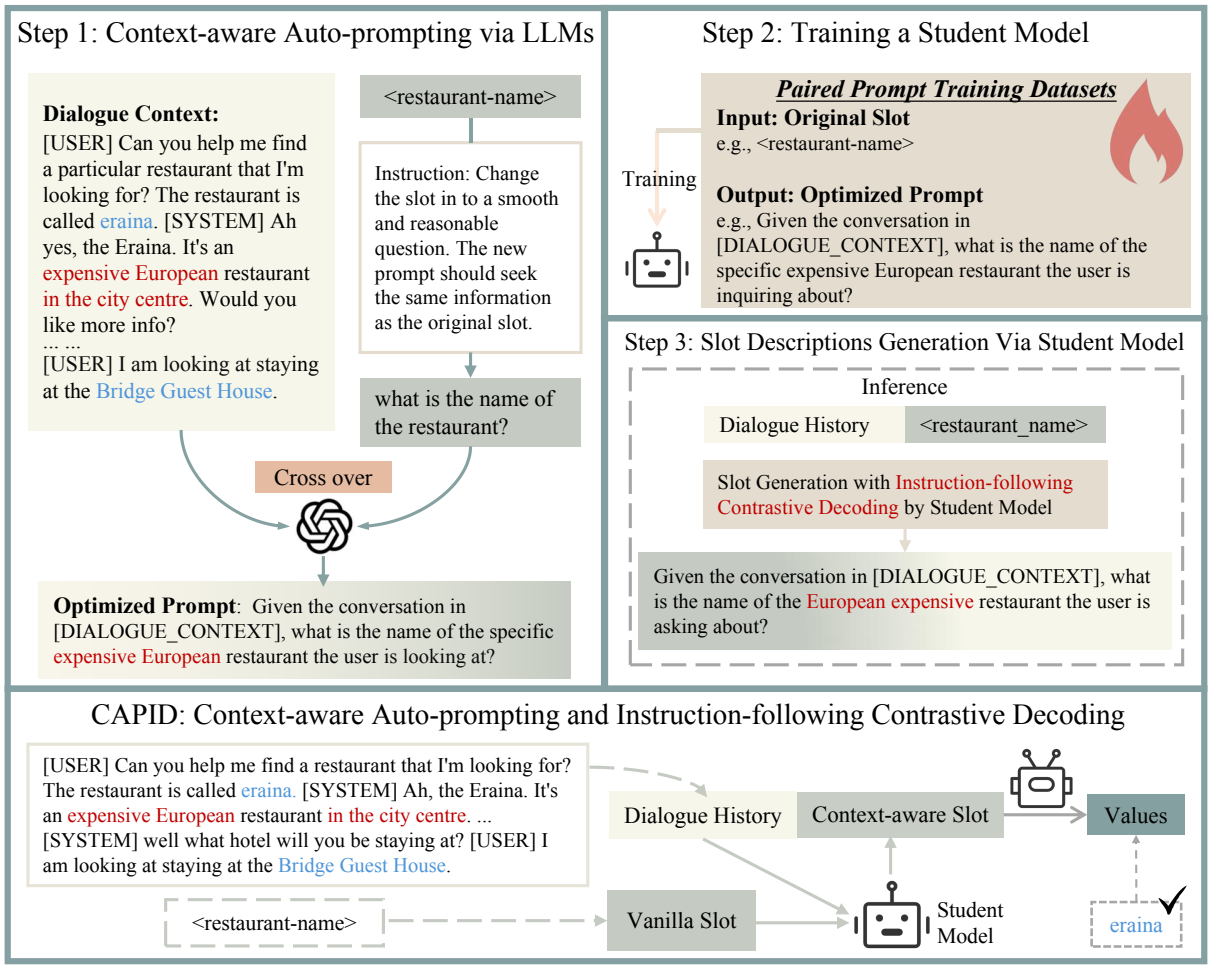


Figure 4: Overview of our CAPID framework. The upper panel delineates our three-stages process for generating context-aware slots. Initially, we harness LLMs to craft context-aware slot query. These refined slots are formatted into datasets featuring paired prompts for instruction-tuning in student model, then the student model is deployed extensively for slot generation with instruction-following contrastive decoding method. The lower panel of the figure illustrates how the CAPID framework combines with the DST task.

prompt", which deliberately instructs the model not to generate responses that adhere strictly to the original slot but to produce entities that actually exist in the dialogue. Then we use the output of the "repurposed prompt" $\mathbf{x}_r$ as a penalty to the origin prompt. Details on the implementation of the original and repurposed prompts for tuning the student model's instructions are provided in Appendix A.2.

Specifically, the original prompt of length n is denoted as $\mathbf{x}_o = x_1 ... x_n$, the "repurposed prompt" of length m is denoted as $\mathbf{x}_r = x_1 ... x_m$. The distribution of the next token during the contrastive decoding can be formulated as:

$$L_t = \log s(x_t|\mathbf{x}_o x_{<t}; \theta) - \beta \log s(x_t|\mathbf{x}_r x_{<t}; \theta), \quad (1)$$

$$p(x_t|\mathbf{x}_o x_{<t}; \theta) := \text{softmax}(L_t), \text{ s.t. } x_t \in \mathbf{V}_{\text{valid}}, \quad (2)$$

where θ symbolizes the parameters of the actual student model, and s represents the logits produced by the model. In Eq. (1), we introduce an repurposed prompt $\mathbf{x}_r$ as a penalty to suppress the original likelihood of tokens that may lead to hallucinations. The authentic distribution of the next token is defined in Eq. (2).

We introduce a trick termed adaptive plausibility constraint (i.e., $x_t \in \mathbf{V}_{\text{valid}}$) to select a subset valid of tokens for a penalty as follows:

$$\mathbf{V}_{\text{valid}} = \{x_t \in \mathbf{V} : p(x_t|\mathbf{x}_o x_{<t}; \theta) \geq \alpha \max_w p(w|\mathbf{x}_o x_{<t}; \theta)\}, \quad (3)$$

where $\mathbf{V}$ is the vocabulary set. This constraint is strategically implemented to mitigate the effects of contrastive decoding when our model demonstrates high confidence in its predictions. Specifically, if the genuine model assigns a probability greater than 0.9 to a particular token, suggesting that this token is the dominant choice post-softmax, the contrastive decoding is designed to refrain from

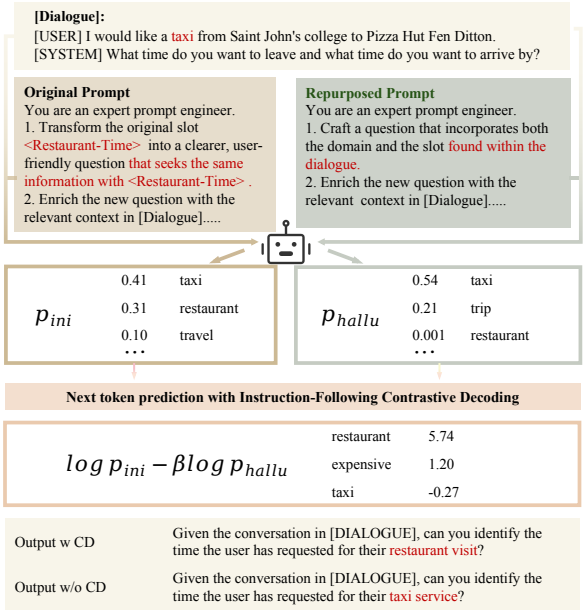


Figure 5: Instruction-following Contrastive Decoding. This figure presents an off-topic scenario. The model receives two instructions and might overlook the first. We design a repurposed prompt to deliberately generate entities from the dialogue, which is penalized during decoding to ensure the first instruction is followed.

intervening. This ensures that the model's high-confidence outputs are preserved without undue influence from the contrastive mechanism.

4 Experiments

4.1 Experiment Setup

Dataset Our experiments use the MultiWOZ 2.1 (Eric et al., 2019) and MultiWOZ 2.4 (Ye et al., 2021) datasets, widely utilized in previous cross-domain research. MultiWOZ 2.4 builds on MultiWOZ 2.1 with improved DST evaluation and reannotated validation and test sets. We use MultiWOZ 2.4 as a reliable dataset for testing and MultiWOZ 2.1 to assess model robustness and its ability to handle annotation noise. Both datasets span five domains, with each conversation potentially covering multiple domains.

Evaluation Metrics We evaluate DST performance using Joint Goal Accuracy (JGA) and Average Goal Accuracy (AGA), consistent with previous works (Feng et al., 2023). JGA calculates the proportion of dialogue turns where the entire state is accurately predicted, while AGA measures the mean accuracy across active slots per turn. A slot is active if its value is mentioned in the current turn and is not inherited from previous turns.

Baselines We compare our approach against existing cross-domain zero-shot approaches, which train on MultiWOZ with one domain excluded and then evaluate on the held-out domain. These methods include TRADE (Wu et al., 2019), TransferQA (Lin et al., 2021a), T5DST (Lin et al., 2021b), D3ST (Zhao et al., 2022). TRADE and MADST use RNNs and slot names as input, where T5DST uses slot descriptions with a T5-small model. D3ST uses slot descriptions and a T5 model. As for MultiWOZ 2.4, we select recent baseline models including IC-DST (Hu et al., 2022), ParsingDST (Wu et al., 2023), RefPyDST (King and Flanigan, 2023), D0T (Finch et al., 2024).

Training Details We employ the CAPID method in a cross-domain setting, training on four domains and evaluating on the unseen domain. We train the student model with a T5-base model using a balanced dataset of five thousand samples, equally distributed between "None" and other specified values, over five epochs at a $3e-4$ learning rate.

The primary reason for selecting the T5-base model as our student model is its efficiency. After testing the inference time for 100 samples on an NVIDIA 3090 GPU, we find that the T5-base model completes the task in approximately 27 seconds, making it roughly four times faster than the LLaMA-2-7B model, as is shown in Table 4. Additionally, our experiments reveal no significant differences in the quality of context-aware slot queries generated by either model. Therefore, we conclude that the T5-based model possesses adequate capabilities for this task.

To ensure that student models learn from a diverse corpus, we develop a sampling strategy by maximizing slot variations and dialogue context coverage. This involves uniformly sampling across all slot types and randomly selecting 5,000 dialogues, ensuring a broad spectrum of dialogue contexts for training. The model, trained exclusively on known domains, is not exposed to the held-out domain. We assess our method's generalization using various backbones, including LLaMa2-CHAT-7B (Touvron et al., 2023), T5-small (60M) (Raffel et al., 2020), and T5-base (220M). For instruction-following contrastive decoding, we set the β at 0.9, α at 0.1.

4.2 Main Results

Context-aware slots descriptions effectively boost DST Performance. We conducted com-

Method	Backbone	MultiWoz 2.1											
		Hotel		Restaurant		Taxi		Attraction		Train		Average	
		JGA	AGA	JGA	AGA	JGA	AGA	JGA	AGA	JGA	AGA	JGA	AGA
TRADE	ELMo	13.70	65.32	11.52	53.43	60.58	73.92	19.87	55.53	22.37	49.31	25.61	59.50
MA-DST	ELMo	16.28	-	13.56	-	59.27	-	22.46	-	22.76	-	26.87	-
TransferQA	T5-large	22.72	77.84	26.28	81.73	61.87	86.48	31.25	60.62	36.72	87.21	35.77	78.78
T5DST	T5-small	21.21	-	21.65	-	64.62	-	33.09	-	35.43	-	35.2	-
D3ST	T5-base	21.80	-	38.20	-	78.40	-	56.40	-	37.70	-	46.5	-
Prompter	PPTOD-s	19.20	-	26.00	-	66.30	-	35.80	-	39.00	-	37.26	-
FNCTOD	LLaMa2-13B	46.83	-	60.27	-	67.48	-	62.24	-	60.90	-	59.54	-
LDST	LLaMa2-7B	63.32	-	73.72	-	91.47	-	75.61	-	75.03	-	75.83	-
CAPID	T5-small	31.10	72.56	31.64	69.06	65.41	83.75	40.88	68.99	34.26	65.93	40.66	72.06
CAPID	T5-base	43.53	83.33	37.08	75.24	87.06	92.02	33.33	64.42	49.46	73.42	50.10	77.69
CAPID	LLaMa2-7B	**71.64**	**94.24**	**77.51**	**95.26**	**91.21**	**95.99**	**83.63**	**92.59**	**89.97**	**97.77**	**82.79**	**95.17**

Table 2: Overall performance on the MultiWOZ 2.1 dataset across various backbones, evaluated using JGA and AGA. Our CAPID framework outperforms the previous best method, LDST, with a 9.17% increase with LLaMa2-7B in JGA. Additionally, CAPID surpasses baseline models with other backbones, such as T5-small (with an 5.46% absolute improvement in JGA) and T5-base (with an 3.6% absolute improvement in JGA).

Method	Backbone	MultiWoz 2.4											
		Hotel		Restaurant		Taxi		Attraction		Train		Average	
		JGA	AGA	JGA	AGA	JGA	AGA	JGA	AGA	JGA	AGA	JGA	AGA
IC-DST	Codex	46.69	-	57.28	-	71.35	-	59.97	-	49.37	-	56.93	-
ParsingDST	Gpt-3.5-turbo-0301	46.76	-	67.67	-	80.58	-	65.63	-	62.59	-	64.65	-
RefPyDST	LLaMa2-13B	51.20	-	65.60	-	67.10	-	70.90	-	69.20	-	64.70	-
D0T	T5-11B	32.00	-	72.30	-	50.60	-	68.10	-	55.80	-	55.70	-
D0T	Llama2-13B	56.40	-	78.80	-	54.70	-	76.80	-	76.10	-	68.60	-
CAPID	T5-small	38.73	77.05	29.35	67.92	73.35	88.25	47.87	74.32	47.88	74.43	47.43	76.39
CAPID	T5-base	31.34	79.05	39.14	78.68	89.28	93.75	22.816	59.33	56.71	76.86	47.86	77.53
CAPID	LLaMa2-7B	**71.30**	**94.55**	**79.06**	**95.44**	**91.62**	**96.04**	**84.40**	**93.13**	**89.56**	**97.62**	**83.19**	**95.35**

Table 3: Overall performance on the MultiWOZ 2.4 dataset across various backbones, evaluated using JGA and AGA. Our CAPID framework outperforms the previous method, D0T, by achieving a 21.26% increase on JGA.

Backbone Model	Total Inference Time for 100 Samples
LLaMA-2-7B	113.84 s
T5-base	27.59 s

Table 4: We choose T5-base as the student model, since it completes the task roughly four times faster than LLaMA-2-7B.

prehensive evaluations of the CAPID method, detailed in Table 2 for MultiWOZ 2.1 and Table 3 for MultiWOZ 2.4. Our findings confirm that context-aware slot descriptions substantially improve DST performance. For MultiWOZ 2.1, integrating this method with advanced models like T5-small, T5-base, and LLAMa-7B results in significant performance boosts across domains: T5-small reaches 40.66%, T5-base 50.10%, and LLaMa2-7B achieves an exceptional 82.79%. For MultiWOZ 2.4, T5-small reaches 47.43%, T5-base 47.86%, and LLaMa2-7B achieves 83.19%. Moreover, the consistent performance gains across diverse datasets suggest that our approach is robust and versatile.

Context-aware slots significantly minimize the ineffective knowledge transfer and negative transfer problems in zero-shot cross-domain DST. Figure 3 shows that context-aware slots have significantly lowered the average negative transfer ratio, from 88.31% to 33.10%, particularly in slots like those ending in "name" seen during training, highlighting their effectiveness in clarifying ambiguities across various domains. Figure 6 showcases the CAPID's superior performance in minimizing ineffective knowledge transfer challenge.

We use misclassified off-topic slot values to measure its degree, which refers to the number of instances where a slot in the conversation has a value, but the model fails to extract it, incorrectly predicting it as "none". CAPID consistently exhibits fewer misclassifications (illustrated by yellow bars)

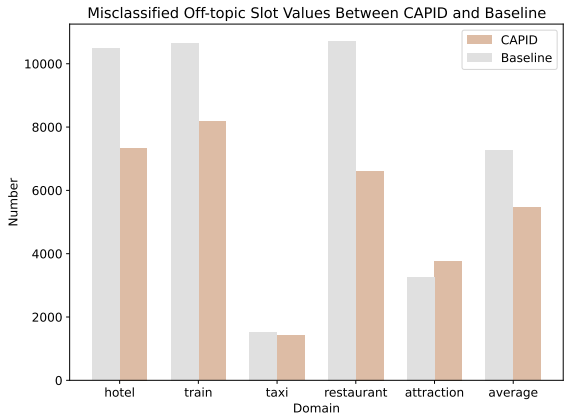


Figure 6: CAPID minimizes ineffective knowledge transfer, as demonstrated by a significant reduction in the number of slots misclassified as "none" across most domains, compared to the baseline using standard pre-defined slots. Both CAPID and the baseline utilize the same backbone network.

than the baseline model (grey bars), with an overall decrease in errors across domains. This improvement underscores the CAPID's enhanced accuracy and consistency in diverse settings.

4.3 Ablation Study

Effectiveness of context-aware slot prompting. Our initial experiment focuses on the impact of context-aware slot prompting. In previous studies, there are three types of slot forms: *(i)* conventional pre-defined slots, e.g., "<Restaurant_Area>". *(ii)* Question-answering which transform the slot into a question using hand-made craft "what is the <slot> of the <domain>", e.g. "what is the area of the restaurant". *(iii)* Schema-driven Prompting leveraging the official slot description offered by MultiWOZ datasets, e.g. "area or place of the restaurant". The JGA scores for the models using T5-small as the backbone are presented in Table 5, while the scores for the models utilizing LLaMA2-7B as the backbone, which were trained on only 5% of the training data, can be found in Table 6.

The CAPID model outperforms the baseline approaches across all domains. In contrast, the conventional pre-defined slots shows the lowest performance on average. The Question-answering and Schema-driven Prompting methods demonstrate moderate improvements.

Effectiveness of instruction-following contrastive decoding. Our second investigation focuses on the influence of the Instruction-following Contrastive Decoding method. In this ablation study, we evaluate the effectiveness of instruction-following contrastive decoding within the context-aware auto-prompting framework. The results, depicted in Table 7, suggest that instruction-following contrastive decoding plays a significant role in enhancing model performance. Without instruction-following CD, there is a notable degradation in how accurately the model adheres to the intended slot meanings, often substituting them with unrelated elements from the dialogue. The significant decrease in average JGA from 43.73% to 37.22% across all domains quantitatively demonstrates that instruction-following contrastive decoding crucially enhances the model's capacity to discern relevant context and accurately generate slot values.

	Hotel	Restaurant	Taxi	Attraction	Train	Average
Vanilla Pre-defined Slots	16.28	13.56	59.27	22.46	22.76	26.87
Question-answering	22.72	26.28	61.87	31.25	36.72	28.18
Schema-driven Prompting	21.21	21.65	64.62	33.09	35.43	35.2
CAPID	33.33	48.11	59.60	34.50	43.12	43.73

Table 5: Ablation study on the effectiveness of context-aware slot query using T5-small as the backbone.

Slot Prompting Approach	Hotel	Restaurant	Taxi	Attraction	Train	Average
Vanilla Pre-defined Slots	29.67	31.24	48.77	37.43	42.40	37.90
Schema-driven Prompting	43.22	50.19	64.15	56.58	50.62	52.95
CAPID	45.19	58.70	91.24	63.34	69.02	65.60

Table 6: Ablation study on the effectiveness of context-aware slot query using LLaMa2-7B as the backbone.

Impact of training data size. Under the context of LLMs, models have become huge and difficult to train or even obtain. Table 8 illustrates the impact of varying training data sizes on the performance of a model utilizing the LLaMa2-7B architecture. The data sizes are represented as percentages (5%, 8%, and 10%) of the total available training data. From the data, there is a clear trend: as the percentage of utilized training data increases, so does the performance across all domains. This analysis indicates that even with 8% of the training data, the model achieves high performance, demonstrating the efficiency of the LLaMa2-7B architecture in leveraging data for improved outcomes.

5 Related Work

5.1 Zero-shot Cross-domain DST

Dialogue State Tracking (DST) is a critical component of Task-Oriented Dialogue Systems (TODS) (Lee et al., 2019; Heck et al., 2020; Hosseini-Asl et al., 2020; Peng et al., 2020; Feng

	Hotel	Restaurant	Taxi	Attraction	Train	Average
CAPID w/o CD	24.69	27.42	50.92	47.10	35.99	37.22
CAPID w CD	33.33	48.11	59.60	34.50	43.12	43.73

Table 7: Ablation study on the effectiveness of instruction-following contrastive decoding. Instruction-following contrastive decoding is essential for improving model accuracy, as demonstrated by a significant drop in JGA from 43.73% to 37.22% when it's absent.

	Hotel	Restaurant	Taxi	Attraction	Train	Average
w 5%	45.19	58.70	91.24	63.34	69.02	65.60
w 8%	71.64	77.51	91.21	83.63	89.97	82.79
w 10%	76.89	82.92	93.10	88.84	89.05	86.16

Table 8: Impact of training data size on performance. Increased data usage enhances performance across all domains.

et al., 2024a; Zhang et al., 2022a, 2023a), which are designed to assist users by understanding their needs and providing relevant suggestions throughout a conversation (Lu et al., 2021), effectively offering personalized recommendations (Liu et al., 2022, 2024a). However, collecting training data for new domains is costly, making it crucial to address zero-shot cross-domain DST for unseen domains.

Prior strategies for zero-shot cross-domain DST typically involve using auxiliary dialogue corpora (Su et al., 2022; Lin et al., 2021a) or reformatting DST tasks into analogous formats, such as question answering (Lin et al., 2021b) or summarization (Shin et al., 2022). Despite these efforts, the zero-shot performance of these methods remains unsatisfactory.

With the rise of large language models (LLMs), significant progress has been made across a wide range of domains due to their ability to generalize from vast and diverse data (Zhao et al., 2024; Liu et al., 2024b). LLMs have also shown promise in DST tasks (Feng et al., 2023, 2024a; Li et al., 2024; Heck et al., 2023). However, despite the strong performance of recent LLM-based DST methods, their zero-shot DST capabilities remain unsatisfactory (Hu et al., 2022; Bang et al., 2023; Hudeček and Dušek, 2023; Heck et al., 2023; Zhang et al., 2023b; Chung et al., 2023). These methods often rely on crafted prompt templates that can introduce ambiguity and fail to capture nuanced dialogue information. The underlying reasons for their effectiveness in redefining slots also remain unclear and difficult to interpret.

5.2 Automatic Prompt Optimization

Prompt learning and optimization have recently become key research areas, as they allow models to adapt to specific tasks with minimal additional training, enhancing flexibility and efficiency in both NLP and CV applications (Shin et al., 2020; Lee et al., 2023; Wang et al., 2024). Optimizing prompts for LLMs is also essential for boosting performance.

Previous researches suggest that an automated self-improvement process can optimize hard prompts more effectively for specific domains, as first introduced by AutoPrompt (Shin et al., 2020). The advent of LLMs has enabled automated techniques for prompt engineering. Zhang et al. (2022b) automatically identify reasoning chains for few-shot Chain of Thought applications. Automatic Prompt Engineer (Zhou et al., 2022b) generates prompt candidates through an automatic model and then uses another model to mutate them. Cheng et al. (2023) develop an automatic prompt optimizer that refines human-generated prompts to better convey human intent.

6 Conclusion

We have introduced CAPID, a novel framework that creates dynamic, context-aware slot prompts for zero-shot cross-domain DST. By utilizing these context-aware slot prompts, we have substantially reduced the complexity of DST, enhancing the model's comprehension and adaptability to new domains. The combination of a student model trained on context-aware slot prompts with our instruction-following contrastive decoding method has significantly reduced errors on off-topic slots. Extensive experiments on two datasets demonstrate the superior performance of the CAPID framework, highlighting its robustness and versatility across various language models.

Limitations

The current auto-prompting strategy employed by our CAPID framework is relatively basic. The robustness and effectiveness of our auto-prompting process can potentially be enhanced through the implementation of multi-step optimization. These enhancements could increase contextual sensitivity of the new prompts, thus boosting the accuracy and reliability of the DST process.

Acknowledgements

We thank the anonymous reviewers for their valuable feedback. This research was partially supported by the grant of HK ITF ITS/359/21FP.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Taha Aksu, Min-Yen Kan, and Nancy F Chen. 2023. Prompter: Zero-shot adaptive prefixes for dialogue state tracking domain adaptation. *arXiv preprint arXiv:2306.04724*.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Jiale Cheng, Xiao Liu, Kehan Zheng, Pei Ke, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. 2023. Black-box prompt optimization: Aligning large language models without model training. *arXiv preprint arXiv:2311.04155*.

Willy Chung, Samuel Cahyawijaya, Bryan Wilie, Holy Lovenia, and Pascale Fung. 2023. Instructtods: Large language models for end-to-end task-oriented dialogue systems. *arXiv preprint arXiv:2310.08885*.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Yujie Feng, Xu Chu, Yongxin Xu, Guangyuan Shi, Bo Liu, and Xiao-Ming Wu. 2024a. Tasl: Continual dialog state tracking via task skill localization and consolidation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1266–1279.

Yujie Feng, Bo Liu, Xiaoyu Dong, Zexin Lu, Li-Ming Zhan, Xiao-Ming Wu, and Albert Lam. 2024b. Continual dialogue state tracking via reason-of-select distillation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7075–7087.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. 2023. Towards llm-driven dialogue state tracking. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 739–755.

James D Finch, Boxin Zhao, and Jinho D Choi. 2024. Leveraging diverse data generation for adaptable zero-shot dialogue state tracking. *arXiv preprint arXiv:2405.12468*.

Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, and Milica Gašić. 2023. Chatgpt for zero-shot dialogue state tracking: A solution or an opportunity? *arXiv preprint arXiv:2306.01386*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. *arXiv preprint arXiv:2005.02877*.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

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. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2627–2643.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Vojtěch Hudeček and Ondřej Dušek. 2023. Are llms all you need for task-oriented dialogue? *arXiv preprint arXiv:2304.06556*.

Brendan King and Jeffrey Flanigan. 2023. Diverse retrieval-augmented in-context learning for dialogue state tracking. *arXiv preprint arXiv:2307.01453*.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4937–4949.

Dongjun Lee, Seokwon Song, Jihee Suh, Joonmyeong Choi, Sanghyeok Lee, and Hyunwoo J Kim. 2023. Read-only prompt optimization for vision-language few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1401–1411.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. *arXiv preprint arXiv:1907.07421*.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023. Contrastive decoding:

Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312.

Zekun Li, Zhiyu Zoey Chen, Mike Ross, Patrick Huber, Seungwhan Moon, Zhaojiang Lin, Xin Luna Dong, Adithya Sagar, Xifeng Yan, and Paul A Crook. 2024. Large language models as zero-shot dialogue state tracker through function calling. *arXiv preprint arXiv:2402.10466*.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Zhenpeng Zhou, Paul A Crook, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, et al. 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.

Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul A 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.

Zhaojiang Lin, Andrea Madotto, Genta Indra Winata, and Pascale Fung. 2020. Mintl: Minimalist transfer learning for task-oriented dialogue systems. *arXiv preprint arXiv:2009.12005*.

Qijiong Liu, Xiaoyu Dong, Jiaren Xiao, Nuo Chen, Hengchang Hu, Jieming Zhu, Chenxu Zhu, Tetsuya Sakai, and Xiao-Ming Wu. 2024a. Vector quantization for recommender systems: A review and outlook. *arXiv preprint arXiv:2405.03110*.

Qijiong Liu, Jieming Zhu, Quanyu Dai, and Xiao-Ming Wu. 2022. Boosting deep ctr prediction with a plug-and-play pre-trainer for news recommendation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2823–2833.

Qijiong Liu, Jieming Zhu, Yanting Yang, Quanyu Dai, Zhaocheng Du, Xiao-Ming Wu, Zhou Zhao, Rui Zhang, and Zhenhua Dong. 2024b. Multimodal pre-training, adaptation, and generation for recommendation: A survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6566–6576.

Zexin Lu, Jing Li, Yingyi Zhang, and Haisong Zhang. 2021. Getting your conversation on track: Estimation of residual life for conversations. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 1036–1043. IEEE.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. *arXiv preprint arXiv:2005.05298*, 3.

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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Lothar M Schmitt. 2004. Theory of genetic algorithms ii: models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling. *Theoretical Computer Science*, 310(1-3):181–231.

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.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

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.

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*, pages 2118–2122.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Cong Wang, Jinshan Pan, Wanyu Lin, Jiangxin Dong, Wei Wang, and Xiao-Ming Wu. 2024. Self-promer: Self-prompt dehazing transformers with depth-consistency. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 5327–5335.

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. *arXiv preprint arXiv:1905.08743*.

Yuxiang Wu, Guanting Dong, and Weiran Xu. 2023. Semantic parsing by large language models for intricate updating strategies of zero-shot dialogue state tracking. *arXiv preprint arXiv:2310.10520*.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2021. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *arXiv preprint arXiv:2104.00773*.

Haode Zhang, Haowen Liang, Liming Zhan, Xiao-Ming Wu, and Albert Lam. 2023a. Revisit few-shot intent classification with plms: Direct fine-tuning vs. continual pre-training. *arXiv preprint arXiv:2306.05278*.

Haode Zhang, Haowen Liang, Yuwei Zhang, Liming Zhan, Xiao-Ming Wu, Xiaolei Lu, and Albert Lam. 2022a. Fine-tuning pre-trained language models for few-shot intent detection: Supervised pre-training and isotropization. *arXiv preprint arXiv:2205.07208*.

Xiaoying Zhang, Baolin Peng, Kun Li, Jingyan Zhou, and Helen Meng. 2023b. Sgp-tod: Building task bots effortlessly via schema-guided llm prompting. *arXiv preprint arXiv:2305.09067*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022b. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*.

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

Xiangyu Zhao, Bo Liu, Qijiong Liu, Guangyuan Shi, and Xiao-Ming Wu. 2024. Easygen: Easing multimodal generation with bidiffuser and llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1351–1370.

Han Zhou, Ignacio Iacobacci, and Pasquale Minervini. 2022a. Xqa-dst: Multi-domain and multilingual dialogue state tracking. *arXiv preprint arXiv:2204.05895*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022b. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Prompt Templates

A.1 Prompt Templates for Generating Context-aware Slot via LLMs.

Below, we present a specific example of the instruction used to elicit context-aware slot generation from ChatGPT.

{
  Instruction: "{}"
  Dialogue: "{}"
  Groundtruth:"{}"
  Model Output:"{}"

  Be an expert prompt engineer and enhance the original slot "<domain_slot>" by transforming it into a question that effectively solicits the ground truth, utilizing the context information related to the "groundtruth"
  Pay attention to:
1.Replace the slot "<domain_slot>" with synonyms in the dialogue, if any.
2.If the slot "<domain_slot>" is mentioned in the dialogue, cross over the question with the dialogue context information. Ensure that the question includes adjectival phrases from the dialogue that describe the "groundtruth". It is crucial that these adjectival phrases are accurately incorporated into the question to maintain context relevance and accuracy. If not, do nothing.
3.Replace the whole dialuge with a token:"[DIALOGE_CONTEXT]".
4.You should never generate a response to the original instruction!
Output with the following format:
  Valuable context information from the dialogue: xxx [END]
  Optimized slot description: xxx [END]
}

A.2 Prompt Templates for Student Model Instruction Tuning

Below, we provide a specific example of the original instruction and the evil instruction used for fine-tuning the student model.

The original instruction:

{
  You are an expert prompt engineer.
1.Transform the original slot <Restaurant-Time> into a clearer, user-friendly question that seeks the same information with <Restaurant-Time>.
  2.Enrich the new question with the relevant context in [Dialogue], like adjectival phrases for the domain or synonyms for the slot.
  [Dialogue]:[USER] I would like a taxi from Saint John's college to Pizza Hut Fen Ditton. [SYSTEM] What time do you want to leave and what time do you want to arrive by?]
}

The repurposed instruction:

```
You are an expert prompt engineer.
{
1.Craft a question that incorporates both the domain and the slot found within the dialogue.
2.Enrich the new question with the relevant context in [Dialogue], like adjectival phrases for the domain or synonyms for the slot.
[Dialogue]:[USER] I would like a taxi from Saint John's college to Pizza Hut Fen Ditton. [SYSTEM] What time do you want to leave and what time do you want to arrive by? ]
}
```

B Implementation Details

B.1 Instruction-following Contrastive Decoding

In this section, we clearly describe the contrastive decoding methods utilized in different backbone architectures (i.e., Encoder-decoder and Decoder-only), and outline how inputs are handled and how outputs are generated in each scenario.

- **Decoder-only backbone.** In a decoder-only architecture utilizing *causal decoding*, the model iteratively generates each token by considering the previously generated sequence and contrasting different prompts. The process can be formulated as:
  1. Generating token probabilities for the next position based on the concatenated sequence of the selected prompt $\mathbf{x}_i$ and the previously generated output $x_{<t}$, where $\mathbf{x}_i \in (\mathbf{x}_{original}, \mathbf{x}_{repurposed})$.
  2. Comparing these probabilities for $p(x_t|\mathbf{x}_{original}, x_{<t})$ and $p(x_t|\mathbf{x}_{repurposed}, x_{<t})$, and yield the output token through Eq. 2 and Eq. 1
- **Encoder-decoder backbone.** For the eencoder-decoder architecture where causal decoding can not be adopted, the encoder and decoder operate separately, the process can be formulated as:
  1. Encoding the prompt $\mathbf{x}_i$ separately in the encoder.
  2. Feeding the encoded prompts and the sequence $x_{<t}$ to the decoder.
  3. Comparing the output probabilities for each token from the decoder, conditioned on each encoded prompt and yield the output token through Eq. 2 and Eq. 1

B.2 Model Training

B.2.1 DST Model Training

For different backbones, we utilized the following hyperparameters:

- **T5-small (60M), T5-base (220M)**. Training was conducted with a learning rate of 3e-4, batch size of 8, maximum input length of 512, maximum target length of 128, and 5 epochs.
- **LLaMA-7B**: Utilizing LORA for efficiency, with a learning rate of 3e-4, batch size of 128, a cutoff length of 512, and 5 epochs. Lora settings were r = 8, alpha = 16, dropout = 0.05, targeting modules [[q_proj,k_proj,v_proj,o_proj]]. For testing, settings included temperature = 0.02, top_p = 0, top_k = 1, num_beams = 1, max new tokens = 128. Experiments are carried out using 8 NVIDIA 3090 GPUs with 24GB memory.

B.2.2 Student Model Training

We utilized **T5-base (220M)** as the student model backbone, with a learning rate of 3e-4, batch size of 8, maximum input length of 512, maximum target length of 128, and 15 epochs.

We train the student model using a dataset comprising five thousand samples generated by ChatGPT. These samples are evenly split between instances with a ground truth of "None" and instances with specified ground truth values other than "None".

C Further Ablation Study on Parameters of Instruction-following Contrastive Decoding

The proposed Instruction-following Contrastive Decoding method incorporates two key hyperparameters, α in Eq. 3 and β in Eq. 1. We choose an example with the output slot query: "Given the conversation in [DIALOGUE_CONTEXT], can you identify the specific day of the week the user intends to travel?", and vary the two hyperparameters to see their influence on the output, as is shown in Table 9.

β	α	Regenerated Slot with Instruction-following Contrastive Decoding
0.9	0.1	Considering the conversation in [DIALOGUE_CONTEXT], can you identify the specific day of the week the user wishes to take the **train**?
0.5	0.1	Considering the conversation in [DIALOGUE_CONTEXT], can you identify the specific day of the week the user wishes to take the **train**?
0.3	0.1	Considering the conversation in [DIALOGUE_CONTEXT], can you identify the specific day of the week the user wishes to take the **train**?
0.1	0.1	Considering the conversation in [DIALOGUE_CONTEXT], can you identify the specific day of the week the user wishes to travel?
0.9	0.9	Considering the conversation in [DIALOGUE_CONTEXT], can you identify the specific day of the week the user wishes to travel?

Table 9: Sensitivity analysis of the parameters in instruction-following contrastive decoding.