

# Evaluating and Enhancing Out-of-Domain Generalization of Task-Oriented Dialog Systems for Task Completion without Turn-level Dialog Annotations

Anonymous ACL submission

## Abstract

Traditional task-oriented dialog (ToD) systems rely heavily on labor-intensive turn-level annotations, such as dialogue states and policy labels, for training. This work explores whether large language models (LLMs) can be fine-tuned solely on natural language dialogs to perform ToD tasks, without requiring such annotations. We evaluate their ability to generalize to unseen domains and compare their performance with models trained on fully annotated data. Through extensive experiments with three open-source LLMs of varying sizes and two diverse ToD datasets, we find that models fine-tuned without turn-level annotations generate coherent and contextually appropriate responses. However, their task completion performance – measured by accurate execution of API Calls – remains suboptimal, with the best models achieving only around 53% success in unseen domains. To improve task completion, we propose ZeroToD, a framework that incorporates a schema augmentation mechanism to enhance API Call accuracy and overall task completion rates, particularly in out-of-domain settings. We also compare ZeroToD with fine-tuning-free alternatives, such as prompting off-the-shelf LLMs, and find that our framework enables smaller, fine-tuned models that outperform large-scale proprietary LLMs in task completion. Additionally, a human study evaluating informativeness, fluency, and task completion confirms our empirical findings. These findings suggest the feasibility of developing cost-effective, scalable, and zero-shot generalizable ToD systems for real-world applications.

## 1 Introduction

Task-oriented dialog (ToD) systems (Zhang et al., 2020a) enable users to accomplish diverse tasks through natural language interactions. These systems power virtual assistants, customer service chatbots, and various other applications such as making reservations or scheduling appointments

(Williams et al., 2016; Zhang et al., 2019b). To be effective, ToD systems must not only engage in user interactions to collect and provide task-specific information but also interface with external systems to accurately complete user tasks.

Traditionally, the development of ToD systems has relied heavily on turn-level manually annotated conversational data, where natural language turns are labeled with dialog states and policy actions (Zhang et al., 2020b). However, this *reliance on turn-level annotated data limits the scalability of ToD systems*, as it prevents them from fully leveraging the vast amounts of readily available unannotated task-oriented conversational data. Furthermore, the annotation process is labor-intensive, expensive, and prone to inconsistencies and errors (Eric et al., 2020; Zang et al., 2020; Han et al., 2021; Budzianowski and Vulic, 2019).

Recent advancements in natural language processing, particularly the emergence of pre-trained large language models (LLMs) (Vaswani et al., 2017; Devlin et al., 2019; Radford et al., 2019), offer new opportunities to address these scalability challenges. LLMs have demonstrated remarkable capabilities in diverse language tasks, from understanding context to generating coherent responses. While pre-trained models (e.g., GPT-2) have been employed to develop ToD systems (Hosseini-Asl et al., 2020; Yang et al., 2020; Mosharrof et al., 2023b; Budzianowski et al., 2018), their potential to train ToD systems without turn-level annotations remains largely unexplored, as does their *ability to generalize effectively to unseen domains*.

Beyond natural language interactions – such as requesting task-specific information or providing updates – ToD systems must also interact with external systems (e.g., databases) to ensure successful task completion. This often requires retrieving information or executing actions, such as making a reservation via *an API Call*. While ToD systems described in the literature could, in theory, be

trained to make such API Calls, this capability is rarely evaluated in practice. The lack of rigorous evaluation in this area leaves a significant gap in understanding the readiness of current ToD systems for real-world deployments.

Motivated by the need to evaluate and enhance the out-of-domain generalization of ToD systems, this work investigates three research questions:

**RQ1:** Can pre-trained LLMs be adapted into effective ToD systems without turn-level annotated data (e.g., annotated dialog states)?

**RQ2:** How can we improve the out-of-domain generalization of ToD systems for task completion?

**RQ3:** How does the out-of-domain generalization of fine-tuned ToD systems compare to that of large-scale, proprietary LLMs?

To address RQ1, we frame ToD as a multi-task instruction fine-tuning problem, where the model learns to generate both natural language responses and API Calls by conditioning on the dialog history and domain schema. To enhance task completion performance, we introduce a schema augmentation mechanism that enriches training data with diverse schema variations, significantly improving robustness in unseen domains (RQ2). Finally, to investigate RQ3, we compare fine-tuned ToD systems against fine-tuning-free approaches that rely on large-scale, proprietary LLMs, which are often costly and less controllable.

We conduct extensive experiments on two benchmark ToD datasets – SGD (Rastogi et al., 2019) and KETOD (Chen et al., 2022) – using three open-source models: GPT-2(Radford et al., 2019), Llama-3.2, and FLAN-T5 (Chung et al., 2022). To provide a comprehensive evaluation, we introduce multiple metrics to assess API Call generation, including method name accuracy, parameter correctness, and complete API Call accuracy. For response generation, we use BERTScore (Zhang et al., 2019a) to better capture the semantic similarity between system outputs and ground truth responses. Additionally, we conduct human studies and qualitative analyses on a subset of both datasets to complement automatic evaluations.

Our empirical results provide clear answers to the research questions posed in this study. For RQ1, we compare our approach against state-of-the-art (SOTA) methods that rely on annotated data and find that ToD systems can function effectively without manual annotations by leveraging multi-task instruction fine-tuning. On the complete API accuracy metric, our best model improves by an

average of 62.9% across both datasets compared to the strongest baseline SOTA model trained with turn-level annotated data. For RQ2, we evaluate the impact of schema augmentation by comparing models trained with and without this mechanism. Our results show that augmentation significantly enhances out-of-domain generalization, improving complete API accuracy on unseen domains by 17.05% for FLAN-T5 and 35.6% for Llama-3.2 compared to their non-augmented counterparts.

For RQ3, we compare ZeroToD against fine-tuning-free alternatives in unseen domains and confirm that fine-tuning is advantageous for learning when to make API Calls and maintaining strong out-of-domain performance in complex, multi-turn task completion scenarios. On complete API accuracy for unseen domains, FLAN-T5 achieves an average improvement of 30.45% over the best SOTA approach built with the large-scale GPT-4o model. Furthermore, human study results evaluating informativeness, fluency, and task completion closely align with automatic metrics, confirming our empirical findings.

## 2 Related Work

**Pipeline Approaches.** ToD systems have traditionally been designed as pipeline systems, where separate components for Natural Language Understanding (NLU), Dialog State Tracking (DST), Dialog Policy, and Natural Language Generation (NLG) are used to handle specific parts of the dialog processing (Ren et al., 2018; Lee, 2013; Peng et al., 2018; Le et al., 2021; Wen et al., 2015; Peng et al., 2020; Chen et al., 2019; Budzianowski et al., 2018; Mosharrof et al., 2023a). However, this approach has drawbacks like error propagation, where errors made in early stages adversely effect modules later on in the pipeline.

**End-to-End Approaches.** Recent works have shifted towards E2E learning methods, where the ToD task is formulated as a conditional generation, where the model generates responses based on the entire dialog history and other relevant annotations (e.g., DST) (Hosseini-Asl et al., 2020; Lin et al., 2021; Bang et al., 2023; Zhang et al., 2023; hyun Ham et al., 2020; Chung et al., 2023; Yang et al., 2020; Sun et al., 2022a; Imrattanaurai and Fukuda, 2023; Sun et al., 2022b; Zhao et al., 2022; Peng et al., 2021; Mosharrof et al., 2023b; Siddique et al., 2022). A major drawback of these approaches is the dependency on manually annotated data, thus limiting the usage of the wealth of available data.

Additionally, most of these approaches assume that API Call results are included in the annotated data, thus limiting their ability to evaluate task completion.

**Prompting Approaches.** Another recent research direction in ToD systems is in-context learning, where pre-trained LLMs are adapted to specific domains based on contextual examples without requiring fine-tuning (Labruna et al., 2023; Hudevcek and Dusek, 2023; Dingliwal et al., 2021; Madotto and Liu, 2020; Li et al., 2022; Madotto et al., 2021; Xu et al., 2024). Even though these approaches show promise on generic domains, they fail on complex domains, and have specialized structure or requirements.

### 3 Methodology

#### 3.1 Problem Formulation

We formulate ToD task completion as a conditional sequence generation problem, where the system generates natural language responses or API Calls using the dialog history and related domain schemas. We leverage domain schema to facilitate out-of-domain generalization in ToD systems.

We formalize the schema for a given domain  $d_x \in D$  by specifying a set of user intents  $\mathcal{I}_{d_x}$ . For example, in the Restaurants domain, one such intent might be ReserveRestaurant. Each intent  $i_d \in \mathcal{I}_{d_x}$  is then associated with a set of slots  $\mathcal{S}_{i_d}$ . For instance, party size and reservation time might be slots for the ReserveRestaurant intent. Each slot  $s_i \in \mathcal{S}_{i_d}$  is characterized by a tuple (name( $s$ ), is\_required( $s$ )), indicating the name of slot (e.g., reservation date) and whether it is mandatory to fulfill a desired intent. We denote the entire domain schema for domain  $d_x$  as:

$$\Gamma_{d_x} = (d_x, \mathcal{I}_{d_x}, \{\mathcal{S}_{i_d} \mid i_d \in \mathcal{I}_{d_x}\}).$$

A dialog session  $\mathcal{T}_i$  of up to  $T$  turns is defined as a sequence of user and system utterances:  $\mathcal{T}_i = ((u_1, r_1), (u_2, r_2), \dots, (u_T, r_T))$ , where  $u_t$  is the user utterance and  $r_t$  is the system response at turn  $t$ . We denote the dialog history at turn  $t$  by  $H_t = \{(u_1, r_1), (u_2, r_2), \dots, (u_{t-1}, r_{t-1}), u_t\}$ , which encapsulates all user-system exchanges up to and including the current user utterance  $u_t$ . Since a single dialog may reference multiple domains, if  $\mathcal{T}_i$  spans  $m$  domains, we write  $\mathcal{T}_i \sim \{d_1, d_2, \dots, d_m\} \subseteq D$ .

#### 3.2 Schema Augmentation

Beyond the original set of domain schemas, we create semantic variations of each domain’s intents

and slots. Specifically, for each domain  $d_x \in D$ , we define its  $k$ -th schema variant as:  $\Gamma_{\tilde{d}_x^k} = (\tilde{d}_x^k, \tilde{\mathcal{I}}_{\tilde{d}_x^k}, \{\tilde{\mathcal{S}}_{\tilde{i}_d^k} \mid \tilde{i}_d^k \in \tilde{\mathcal{I}}_{\tilde{d}_x^k}\})$ , where  $\tilde{\mathcal{I}}_{\tilde{d}_x^k}$  is the renamed set of intents, and  $\tilde{\mathcal{S}}_{\tilde{i}_d^k}$  represents the renamed slots for each intent  $\tilde{i}_d^k$ .

For example, in the Restaurants domain, the original intent ReserveRestaurant might be changed to ReserveTable, and the slot party size might become number of people. To integrate these augmented schemas into the dialogs, we systematically replace schema references in existing dialogs with their counterparts from  $\Gamma_{\tilde{d}_x^k}$ . Concretely, for each dialog  $\mathcal{T}_i$  associated with domain  $d_x$ , we construct an augmented dialog  $\tilde{\mathcal{T}}_i^k$  by substituting all intents and slots with those from  $\Gamma_{\tilde{d}_x^k}$ . This procedure preserves the underlying dialog flow but exposes ZeroToD to multiple schema variations, ultimately improving its ability to generalize to out-of-domain task scenarios.

#### 3.3 Multi-task Instruction Fine-tuning

A ToD system must handle diverse interactions, including general conversation, requesting task-specific information, providing details, and making API Calls for task completion. Broadly, the system generates two types of outputs: (i) natural language responses, and (ii) API Calls, which include a method name, parameters, and corresponding values. We employ multi-task instruction fine-tuning that trains the model to autonomously decide between generating an API Call or a user response, without introducing special tokens.

Formally, an autoregressive language model (e.g., GPT-2 (Radford et al., 2019)) generates text by predicting the next token given the preceding context. For a given sequence of tokens  $(x_1, x_2, \dots, x_{t-1})$ , the probability distribution for the next token  $x_t$  is computed as:  $p(x_t \mid x_{1:t-1}; \theta) = f_\theta(x_{1:t-1})$ , where  $f_\theta$  represents the model parameterized by  $\theta$  and outputs a probability distribution over the vocabulary  $\mathcal{V}$ . The next token  $x_t$  is then sampled from this distribution. This formulation extends naturally to response generation in ToD systems, where the system response  $r_t$  at turn  $t$  is generated recursively until an end-of-sequence token ( $\langle \text{eos} \rangle$ ) is produced:  $r_t \sim p(r_t \mid H_t; \theta)$ , where  $H_t$  denotes the dialog history up to turn  $t$ .

To improve out-of-domain generalization, ZeroToD introduces an additional conditioning variable, the domain schema  $\Gamma_{d_x}$  for each domain



Datasets	SGD	KETOD
# Dialogs	16142	5324
Average Turns / Dialog	20.44	9.78
# Unique API methods:all	46	46
# Unique API methods:unseen	8	8
# Unique API parameters: all	137	134
# Unique API parameters: unseen	88	88

Table 1: Dataset statistics.

$d_x$  and an instruction prompt  $P$ . The instructions encourage the model to comprehend schema representations to better generalize across unseen domains and dialog contexts. Extending the above formulation to multi-task instruction fine-tuning for multi-turn dialogs of length  $T$ , where each dialog may span multiple domains  $\{d_1, d_2, \dots, d_m\} \subseteq D$ , we optimize the following objective:  $-\sum_{t=1}^T \log p(r_t \mid P, \{\Gamma_{d_j}\}_{j=1}^m, H_t; \theta)$ . Since LLMs operate under a finite context length, the dialog history  $H_t$  consists of only the most recent  $k$  turns, where  $k \leq t$ .

### 3.4 Training Details

The dialog history and domain schema are passed through a structured template to form the inputs to the model. The template is detailed in Figure 2 in Appendix D. Training begins with 500 warm-up steps and early stopping on the evaluation loss with a patience value of 3. We used the AdamW (Loshchilov and Hutter, 2017) optimizer with weight decay and a learning rate of 0.001. Experiments were conducted with GPT2-Medium, FLAN-T5 Large and Llama 3.2 3B Instruct models. GPT-2 and FLAN-T5 were fine-tuned fully, while Llama-3.2 used Low-Rank Adaptation (LoRA) (Hu et al., 2021) and 8-bit quantization (Jacob et al., 2018) for memory efficiency.

## 4 Experimental Setup

### 4.1 Datasets

We use two ToD datasets: Schema-Guided Dialog (SGD) dataset, and Knowledge-Enhanced Task-Oriented Dialog (KETOD) dataset. Table 1 shows detailed statistics about the datasets. These datasets are publicly available, large, and represent a wide range of domains that span different tasks. We have selected these datasets as they describe the domain using schema and have the necessary information to simulate communication with external resources through API Calls.

### 4.2 Evaluation

We evaluate the system across four domain categories: *All Domains* (dialogs from all domains), *Seen Domains* (dialogs from training domains), *Un-*

*seen Domains* (dialogs from domains not included in the training data), and *Mixed Domains* (dialogs with both seen and unseen domains). We analyze the performance of overall responses as well as its sub-tasks—Request and Inform. For task completion, we introduce custom metrics to assess the performance of individual components.

**Response Generation.** To evaluate the quality of the response generation of models, we report BERTScore. We used microsoft/mpnet-base as the model type for calculating the BERTScore. We report BLEU-4 (Papineni et al., 2002) scores in Appendix B

**API Calls.** The format for an API Call is: `APICall(method=method_name, parameters = {(s_i, v_i)}_{i=1}^n)`. The parameters attribute is a list of slot name and slot value pairs, where  $s_i$  represents the slot name and  $v_i$  represents the value of that slot.

We use regular expressions to extract different parts of the API Call, and apply custom metrics to access different parts of an API Call.

*Invoke Accuracy* measures whether the system can understand when to make an API Call. *Method Accuracy* checks whether the appropriate method name was used in the API Call. *Param Name Accuracy* assesses whether all the parameter names used to construct the API Call are accurately. *Param Value Accuracy* evaluates whether each parameter value corresponding to a parameter name is correct. It is important to note that this metric will only be considered if the corresponding parameter name is correct. *Complete API Call Accuracy* metric checks whether the complete API Call (i.e., all components) was generated correctly.

### 4.3 Baselines

*SOLOIST* (Peng et al., 2021) introduced an E2E ToD system that employs a transformer-based autoregressive model that generates dialog responses grounded in user goals and real-world knowledge for task completion.

*SimpleTOD* (Chen et al., 2022) introduced a ToD model as an end-to-end sequence generation problem that utilizes the dialog history, dialog states and system actions to generate system responses.

*ZS-TOD* (Mosharrof et al., 2023b) introduced a zero-shot generalizable E2E ToD model that incorporates domain schema and dialog annotations to generate dialog responses.

*AutoTOD* (Xu et al., 2024) introduced a zero shot autonomous ToD agent, that works without manual

Dataset	Model	Annotations Required	Overall Response (BertScore-F1)				Complete API Accuracy			
			all	seen	mixed	unseen	all	seen	mixed	unseen
SGD	SimpleTOD	Yes	0.6100	0.600	0.6300	0.5800	20.30	44.94	25.57	09.39
	SOLOIST	Yes	0.6214	0.6538	0.6265	0.6097	19.25	47.82	24.32	07.66
	ZS-ToD	Yes	0.5704	0.6439	0.5648	0.5600	20.38	56.15	20.28	12.62
	GPT-2	No	0.7002	0.7291	0.7149	0.6800	35.53	74.77	42.87	19.24
	Llama-3.2	No	0.7629	<b>0.7850</b>	0.7708	0.7507	52.84	<b>90.19</b>	57.44	39.84
	FLAN-T5	No	<b>0.7633</b>	0.7792	<b>0.7723</b>	<b>0.7513</b>	<b>65.87</b>	89.88	<b>72.99</b>	<b>53.16</b>
KETOD	SimpleTOD	Yes	0.5248	0.5540	0.5382	0.4735	36.24	61.07	31.19	08.74
	SOLOIST	Yes	0.5035	0.5201	0.4895	0.4983	24.12	43.79	17.29	05.98
	ZS-ToD	Yes	0.4759	0.4822	0.4643	0.4809	26.70	43.29	21.75	10.34
	GPT-2	No	0.6766	0.7001	0.6821	0.6410	36.75	59.56	32.62	10.80
	Llama-3.2	No	0.7369	0.7624	0.7363	0.7057	63.32	91.61	55.08	35.17
	FLAN-T5	No	<b>0.7431</b>	<b>0.7665</b>	<b>0.7457</b>	<b>0.7112</b>	<b>77.26</b>	<b>95.97</b>	<b>75.58</b>	<b>53.79</b>

Table 2: Performance comparison between annotation-dependent baselines and annotation-free ZeroToD models on Overall Response and API Accuracy (RQ1).

Dataset	Model	Annotations Required	Inform (BertScore-F1)				Request (BertScore-F1)			
			all	seen	mixed	unseen	all	seen	mixed	unseen
SGD	SimpleTOD	Yes	0.6100	0.6400	0.6300	0.5800	0.4300	0.4000	0.4500	0.4200
	SOLOIST	Yes	0.6596	0.6982	0.6730	0.6356	0.4852	0.5069	0.4797	0.4858
	ZS-ToD	Yes	0.4714	0.5324	0.4590	0.4681	0.5012	0.5561	0.4944	0.4970
	GPT-2	No	0.7451	0.7597	0.7516	0.7344	0.5287	0.5200	0.5302	0.5291
	Llama-3.2	No	<b>0.7853</b>	<b>0.7962</b>	<b>0.7904</b>	0.7771	<b>0.6073</b>	<b>0.6244</b>	<b>0.6038</b>	<b>0.6071</b>
	FLAN-T5	No	0.7838	0.7906	0.7876	<b>0.7781</b>	0.6034	0.6169	0.5965	0.6066
KETOD	SimpleTOD	Yes	0.4275	0.4445	0.4637	0.3659	0.4267	0.4341	0.4376	0.4076
	SOLOIST	Yes	0.5076	0.4961	0.5317	0.4971	0.4638	0.4588	0.4604	0.4729
	ZS-ToD	Yes	0.3489	0.3334	0.3456	0.3734	0.5167	0.5473	0.4989	0.4991
	GPT-2	No	0.6831	0.7048	0.6878	0.6489	0.4814	0.4919	0.4733	0.4772
	Llama-3.2	No	<b>0.7488</b>	<b>0.7691</b>	0.7503	<b>0.7198</b>	0.5976	<b>0.6198</b>	0.5794	<b>0.5901</b>
	FLAN-T5	No	0.7447	0.7656	<b>0.7607</b>	0.6994	<b>0.5981</b>	0.6189	<b>0.5893</b>	0.5829

Table 3: Results for Response Generation sub-tasks: Inform and Request (RQ1).

annotations and also has the ability to communicate with external resources.

SOLOIST, SimpleTOD and ZS-ToD were implemented using GPT-2 Medium. During inference, we extract the system response and disregard the additional information like dialog state and system actions.

## 5 Results

Table 2 presents the findings for *RQ1: Can pre-trained LLMs be adapted into effective ToD systems without turn-level annotated data?* Our results show that ZeroToD models, which do not rely on turn-level annotations, outperform models trained with annotated data in response generation. A key reason for this improvement is that ZeroToD models focus solely on generating system responses, whereas annotation-based models must produce structured outputs that include dialog state, system actions, and responses—requiring the model to optimize for multiple complex tasks simultaneously. Furthermore, the substantial performance gap between the baseline approaches built with GPT-2 and the GPT-2 variant of ZeroToD suggests that learning to generate responses directly is a more effective approach for ToD systems.

For task completion, all models trained without turn-level annotations consistently outperform the

annotated models. This finding highlights the sufficiency of dialogue history as a standalone source of context for completing complex tasks. Table 2 reveals more insights about the different ZeroToD models. FLAN-T5 and Llama-3.2 being the larger models, significantly outperform the smaller GPT-2 model for task completion. However, even though Llama-3.2 is a larger model than FLAN-T5, it does not have better task completion performance.

This discrepancy may stem from differences in the training methodologies. Specifically, Llama-3.2 was trained using 8-bit quantization and LoRA adapters, whereas FLAN-T5 underwent full fine-tuning. The use of LoRA significantly reduces the number of trainable parameters and the 8-bit quantization introduces precision loss due to the reduced bit width. These factors likely contributed to Llama-3.2’s lower performance despite its larger model size.

**Detailed Response Generation Performance.** To get a better understanding of the response generation task, we break it down into two sub-tasks—Inform and Request—and present the results in Table 3. The inform sub-task focuses on providing responses to user requests, while the request sub-task involves prompting users for additional information. Similar to Table 2, the results here

Dataset	Model	Augmented	Overall Response(BertScore-F1)				Complete API Accuracy			
			all	seen	mixed	unseen	all	seen	mixed	unseen
SGD	GPT-2	✗	0.7002	0.7291	0.7149	0.6800	35.53	74.77	42.87	19.24
	GPT-2	✓	0.7266	0.7431	0.7437	0.7068	47.66	82.01	53.76	33.75
	Llama-3.2	✗	0.7629	0.7850	0.7708	0.7057	52.84	90.19	57.44	39.84
	Llama-3.2	✓	0.7623	<b>0.7852</b>	0.7693	<u>0.7506</u>	62.38	<b>94.31</b>	63.99	<u>53.68</u>
	FLAN-T5	✗	<b>0.7633</b>	0.7792	<b>0.7723</b>	<b>0.7513</b>	<u>65.87</u>	89.88	<u>72.99</u>	53.16
	FLAN-T5	✓	0.7320	0.7494	0.7411	0.7196	<b>73.49</b>	<u>90.65</u>	<b>81.76</b>	<b>61.07</b>
KETOD	GPT-2	✗	0.6766	0.7001	0.6821	0.6410	36.75	59.56	32.62	10.80
	GPT-2	✓	0.6677	0.6867	0.6738	0.6372	48.43	72.48	40.64	25.52
	Llama-3.2	✗	0.7369	0.7624	0.7363	0.7057	63.32	91.61	55.08	35.17
	Llama-3.2	✓	0.7405	<u>0.7679</u>	0.7389	0.7082	73.24	<b>97.48</b>	67.02	48.05
	FLAN-T5	✗	<u>0.7431</u>	0.7665	<u>0.7457</u>	<u>0.7112</u>	<u>77.26</u>	95.97	<u>75.58</u>	<u>53.79</u>
	FLAN-T5	✓	<b>0.7549</b>	<b>0.7786</b>	<b>0.7541</b>	<b>0.7261</b>	<b>82.66</b>	<u>96.48</u>	<b>82.35</b>	<b>64.14</b>

Table 4: Impact of Schema Augmentation Mechanism on Response Generation and API Accuracy (RQ2).

show a consistent trend, with ZeroToD models outperforming those trained on annotated data. Additionally, we observe that the Request sub-task is significantly more challenging than Inform. This is expected, as there are multiple plausible pieces of information a system could request, but if they do not align with the gold standard, the model receives a lower score. In contrast, the Inform sub-task is more straightforward since the user explicitly requests specific information, making it easier for the system to generate the correct response.

**Schema Augmentation Performance.** Table 4 presents the results for *RQ2: How can we improve the out-of-domain generalization of ToD systems for task completion??* Across all the models, we can see that the response generation performance is similar, but there are improvements in task completion performance, specially a big increment in the unseen domain. For seen domains, there is a small improvement, which is expected as the augmentation mainly teaches the models how to use the schema to generalize to out-of-domain data, however for unseen domains, this learning is very useful and the models have shown considerable improvements. Between Llama-3.2 and FLAN-T5, we can see that for seen domains Llama-3.2 has a slightly better performance however for unseen domains Llama-3.2 has much lower performance. One reason for this could be the size of the two models, Llama-3.2 being the larger model may have a higher capacity to memorize the training data, which could explain its stronger performance on seen domains. However, this can also make it more prone to over-fitting and may not generalize well to new, unseen domains.

**Detailed Task Completion Performance.** To complete a task, a model has to make a correct API Call. An API Call has many aspects in it, and we present detailed results in Table 5. We can

see that ZeroToD models considerably outperform baseline approaches across all metrics. Upon inspecting the API Call Invoke Accuracy, we see that baseline approaches have much lower scores, indicating that they struggle in identifying when to make API Calls. The API Call Method Accuracy evaluates whether a model generates the correct method name in the API Call. A common pattern that we see across all models is that there is a drop in parameter names accuracy when compared to the previous metrics. Generating the correct list of parameters for the API Call is inherently a harder problem than deciding when to make an API Call and what method to use, so the performance degradation is understandable.

A key observation from Table 5 is the significant impact of the schema augmentation on the API Call parameter names metric. Our results indicate that schema augmentation yields the largest improvement for this metric. API Call parameters are directly derived from the schema, and schema augmentation enables the models to better recognize and utilize these patterns, thus improving the model’s ability to generate the correct list of parameters, leading to a notable increase in parameter names accuracy. Furthermore, the API Call parameter values accuracy also improved as a result, since a model is only rewarded for generating the correct value if it is assigned to the appropriate parameter name.

For instance, consider the task of finding a bus using the FindBus method. We compare two schema variations, Buses\_1 and Buses\_11, which define different slot names for the same concepts. In Buses\_1, the slot names are from\_station and to\_station, and for Buses\_11, the slot names are origin and destination.

A model trained without schema augmentation tends to overfit to specific slot names seen

Dataset	Model	Augmented	API Invoke Accuracy			API Method Accuracy			Param Names Accuracy			Param Values Accuracy		
			all	seen	unseen	all	seen	unseen	all	seen	unseen	all	seen	unseen
SGD	SOLOIST	✗	79.92	80.37	79.44	64.51	72.82	60.74	47.06	72.05	35.54	45.16	71.30	33.13
	SimpleToD	✗	66.28	54.52	68.44	32.94	26.32	33.53	28.52	29.13	26.25	26.98	28.54	24.45
	ZS-ToD	✗	86.65	90.97	86.91	64.18	80.69	61.82	44.44	73.15	40.17	42.74	72.22	38.13
	GPT-2	✗	90.51	97.66	86.21	78.62	96.26	71.42	60.65	92.07	49.52	58.64	91.24	46.98
	GPT-2	✓	93.89	96.81	91.26	83.07	89.17	77.92	76.94	91.71	72.15	74.18	90.86	68.59
	Llama-3.2	✗	98.08	<b>99.69</b>	<u>97.52</u>	92.18	<b>99.69</b>	91.00	84.53	<u>98.36</u>	80.44	81.55	<u>97.92</u>	76.55
	Llama-3.2	✓	<b>98.78</b>	99.38	<b>98.80</b>	95.26	99.38	<u>94.97</u>	<u>89.23</u>	<b>99.10</b>	<u>88.11</u>	<u>86.25</u>	<b>98.73</b>	<u>84.02</u>
	FLAN-T5	✗	<u>98.65</u>	<u>99.61</u>	98.24	<b>96.84</b>	<u>99.61</u>	<b>95.23</b>	79.71	96.97	74.58	76.03	94.04	70.43
	FLAN-T5	✓	96.28	99.53	94.42	<u>95.97</u>	99.53	93.89	<b>94.37</b>	97.23	<b>92.17</b>	<b>92.08</b>	96.99	<b>88.86</b>
KETOD	SOLOIST	✗	52.20	56.04	44.14	42.96	47.65	34.25	37.10	47.43	24.94	36.31	46.98	23.95
	SimpleToD	✗	44.85	41.95	41.84	34.30	33.39	29.43	25.87	30.74	16.87	25.31	30.38	16.30
	ZS-ToD	✗	43.03	43.62	40.46	32.91	36.58	31.72	28.06	31.92	26.89	26.53	30.94	24.43
	GPT-2	✗	78.83	80.87	73.56	71.48	78.36	62.53	57.02	72.98	39.60	55.46	71.83	37.73
	GPT-2	✓	92.96	91.78	94.48	86.62	88.09	85.29	75.66	84.53	69.97	72.85	83.55	65.02
	Llama-3.2	✗	<u>96.55</u>	<u>97.48</u>	<u>95.17</u>	<u>92.90</u>	<u>96.98</u>	88.97	86.04	<u>97.13</u>	76.43	84.26	<u>96.65</u>	73.37
	Llama-3.2	✓	<b>98.49</b>	<b>99.16</b>	<b>97.70</b>	<b>96.80</b>	<b>99.16</b>	<b>94.94</b>	<b>91.86</b>	<b>98.67</b>	<u>86.05</u>	<b>90.10</b>	<b>98.62</b>	<u>82.83</u>
	FLAN-T5	✗	90.45	89.93	90.34	89.07	89.93	86.67	78.62	88.92	67.93	76.52	87.94	65.28
	FLAN-T5	✓	92.34	93.46	92.18	92.15	93.12	<u>92.18</u>	<u>90.77</u>	93.53	<b>88.84</b>	88.86	93.23	<b>85.26</b>

Table 5: Additional API Metrics for baseline approaches and ZeroToD models (RQ1, RQ2).

during training. If the model was trained on Buses\_1, it might always generate from\_station and to\_station, even when interacting with Buses\_11, leading to incorrect API Calls. For example, given the user utterance: “*I want to find a bus from LA to SFO*”, the model without augmentation might generate:

```
API Call(method=FindBus, parameters=
    from_station=LA, to_station=SFO }).
```

In the Buses\_11 schema, the slot names from\_station and to\_station do not exist, thus making the API Call invalid.

On the other hand, a model trained with schema augmentation learns to generalize across schema variations by recognizing slot name patterns from multiple schemas, and might generate:

```
API Call(method=FindBus, parameters=
    origin=LA, destination=SFO}).
```

The model can dynamically align its output with the schema it is conditioned on. By learning to use the slot names from the provided schema rather than relying on the memorized slot names, a model trained with schema augmentation demonstrates improved robustness and generalization.

**Fine-tuning Performance.** Table 7 presents the results on unseen domains for ZeroToD models, and Auto-ToD, which was built using GPT-4o. Using the results in Table 7, we can answer RQ3: *How does the out-of-domain generalization of fine-tuned ToD systems compare to that of large-scale, proprietary LLMs?* For the Complete API Accuracy metric, except for the GPT-2 model, all other ZeroToD models outperform Auto-ToD. For all the other metrics, Auto-ToD has much lower scores than the ZeroToD models. A key metric to note here is

the API Invoke Accuracy, which measures whether a model is making an API call on the right turn, and Auto-ToD has a very low score on this metric when compared to ZeroToD models. Due to this issue, Auto-ToD also has a much lower score for the Overall Response metric, as it makes API Calls on turns where a general interaction is expected. Based on these results, we can state that fine-tuning is an important step to identify the timing of making an API Call in ToD systems.

**Domain Specific Results.** To get a deeper understanding of the performance of Auto-ToD and ZeroToD models, we present some domain specific results for the API Invoke Accuracy and Complete API Accuracy metrics in Table 6. For the Api Invoke Accuracy, we see the same pattern as before, with Auto-ToD having much lower scores than ZeroToD models. From these results, we can make another interesting observation, Auto-ToD has higher Complete Api Accuracy for simple domains like Alarm and Movies, however it has poor performance for complex domains like Restaurants, Buses, and Music. Since ZeroToD models have been fine-tuned, the models have a better understanding of the structure of complex domains. The models do not have a big drop in performance across domains, showing the robustness achieved through fine-tuning. FLAN-T5 and Llama-3.2 being the larger models, show more stability in performance over the smaller GPT-2 model.

**Human Evaluation.** To supplement the automatic metrics and get a qualitative analysis, we conducted a human evaluation using Amazon Mechanical Turk to assess the performance of various models. Two baseline models (SOLOIST and Auto-ToD)



Dataset	Domains	API Invoke Accuracy				Complete API Accuracy			
		Auto-ToD	GPT-2	Llama-3.2	FLAN-T5	Auto-ToD	GPT-2	Llama-3.2	FLAN-T5
SGD	Alarm_1	77.78	71.11	98.89	<b>100.00</b>	76.67	15.56	<b>78.89</b>	61.11
	Buses_3	62.84	94.59	<b>100.00</b>	99.32	37.16	29.05	46.62	<b>57.43</b>
	Events_3	64.29	92.06	92.86	<b>96.83</b>	27.78	55.56	50.79	<b>60.32</b>
	Homes_2	43.06	97.92	<b>100.00</b>	99.31	39.58	69.44	<b>76.39</b>	74.31
	Hotels_4	43.88	94.24	<b>100.00</b>	98.56	40.29	49.64	<b>100.00</b>	77.70
	Movies_3	67.80	54.24	<b>98.31</b>	96.61	47.46	23.73	<b>77.97</b>	67.80
	Music_3	41.51	94.34	98.11	<b>100.00</b>	30.19	58.49	73.58	<b>84.91</b>
	RentalCars_3	48.67	99.12	99.12	<b>100.00</b>	41.59	35.40	54.87	<b>63.72</b>
KETOD	Restaurants_2	60.29	99.26	<b>100.00</b>	<b>100.00</b>	28.68	84.56	77.94	<b>85.29</b>
	Alarm_1	66.67	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	66.67	00.00	<b>100.00</b>	66.67
	Buses_3	50.00	86.36	95.45	<b>100.00</b>	09.09	18.18	<b>45.45</b>	36.36
	Events_3	81.82	<b>100.00</b>	<b>100.00</b>	90.91	45.45	<b>72.73</b>	<b>72.73</b>	54.55
	Homes_2	70.59	<b>100.00</b>	94.12	93.75	<b>70.59</b>	41.18	64.71	<b>70.59</b>
	Hotels_4	43.75	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	37.50	37.50	81.25	<b>100.00</b>
	Movies_3	71.43	85.71	85.71	<b>95.00</b>	<b>57.14</b>	28.57	<b>57.14</b>	<b>57.14</b>
	Music_3	22.22	88.89	<b>100.00</b>	78.26	22.22	44.44	<b>88.89</b>	66.67
	RentalCars_3	62.50	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	<b>56.25</b>	31.25	50.00	50.00
	Restaurants_2	50.00	<b>100.00</b>	87.50	87.50	25.00	00.00	<b>87.50</b>	62.50

Table 6: Domain-wise evaluation of API Invoke and Complete API on unseen domains for Auto-ToD and ZeroToD models (RQ3).

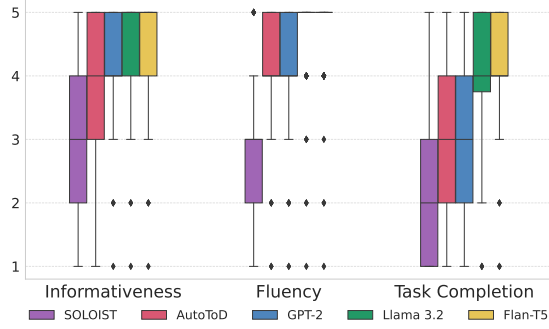


Figure 1: Human Evaluation Study on SGD and KETOD. Evaluators were asked to rate the dialog samples between a range of 1-5 on 3 categories.

and three ZeroToD models (GPT-2, Llama-3.2, and FLAN-T5) were taken into account. We sampled 100 dialogs from each dataset, with 50 coming from single-domain tasks and the remaining 50 from multi-domain tasks, all from the test dataset. Human evaluators were asked to rate the models on a scale from 1 to 5 on three questions: the accuracy of information presented in the responses (Informativeness), how fluent and natural the conversation is (Fluency), and whether the models can make accurate API Call (Task Completion).

The results, shown in Figure 1, align with the automatic metrics, where ZeroToD models outperform the existing SOTA approaches. This demonstrates a strong alignment between quantitative and qualitative assessments. Notably, for task completion and fluency, Llama-3.2 and FLAN-T5 demonstrate superior performance compared to all other models, which is consistent with our previous findings. Another important observation is that Llama-3.2 and FLAN-T5 have less variance in performance across all tasks when compared to all other

Dataset	Model	Overall Response	API Invoke Accuracy	Complete API Accuracy
SGD	Auto-ToD	0.5471	63.15	42.20
	GPT-2	0.7068	91.26	33.75
	Llama-3.2	<b>0.7506</b>	<b>98.80</b>	<b>53.68</b>
	FLAN-T5	0.7196	94.42	<b>61.07</b>
KETOD	Auto-ToD	0.5471	63.22	41.61
	GPT-2	0.6372	94.48	25.52
	Llama-3.2	<b>0.6454</b>	<b>97.70</b>	<b>48.05</b>
	FLAN-T5	0.7050	92.18	<b>59.54</b>

Table 7: Evaluation of fine-tuned approaches against large-scale proprietary LLMs on unseen domains (RQ3).

models, which further solidifies the robustness of our approach.

## 6 Conclusion

This work demonstrates that LLMs fine-tuned solely on natural language dialogs can effectively generalize to unseen domains by framing ToD as a multi-task instruction fine-tuning problem. To further enhance their out-of-domain task completion performance, we introduce schema augmentation, which improves model adaptability to unseen domains and strengthens task completion performance. To ensure robust evaluation of task completion, we explicitly incorporate API Calls as a core task and assess performance using both automatic metrics and human evaluations. Furthermore, we show that fine-tuned ToD systems generalize better to unseen domains than fine-tuning-free approaches that rely on large-scale proprietary LLMs. These results highlight the feasibility of developing cost-effective, scalable, and zero-shot generalizable ToD systems that achieve strong out-of-domain generalization without requiring turn-level annotations, paving the way for their practical adoption in real-world applications.



## 7 Limitations

ZeroToD has been developed by fine-tuning LLMs such as GPT-2, Llama-3.2, and FLAN-T5. These LLMs require significant computational resource requirements to train, particularly Llama-3.2. Training and inference with these models can be expensive, limiting their practicality for deployment in resource-constrained environments.

The LLMs used in the system function as black boxes, making it challenging to interpret the reasoning behind their responses. This lack of transparency hinders the ability to diagnose and correct erroneous outputs, which is crucial in ToD systems where accuracy is critical. Furthermore, the models may inherit biases present in the training data, leading to biased or unfair responses in certain scenarios. Although efforts were made to mitigate this issue by fine-tuning using the dialog datasets, completely eliminating biases remains a challenging task. The reliance on pre-trained models introduces limitations related to the coverage of the pre-training data. If the pre-training data lacks specific domain knowledge, the ToD system may under perform in those domains.

The deployment of LLMs in ToD systems raises ethical and privacy concerns, particularly regarding the handling of sensitive user data. Ensuring that the system complies with privacy regulations and ethical standards is an ongoing challenge that requires continuous monitoring and updates. Similar to other AI technologies, there is a scope for potential misuse of our system. If ZeroToD is used with malicious intent or the model is fed inappropriate data, there is a risk of abuse. We would strongly advise to take necessary precautions and appropriate usage policies.

Addressing the limitations outlined above is crucial for advancing the effectiveness and reliability of ToD systems. While the usage of pre-trained LLMs offers significant advantages, these models are not without their challenges. Increasing model interpretability, mitigating biases, and addressing ethical and societal concerns are essential steps toward creating more robust and responsible ToD systems.

## References

Namo Bang, Jeehyun Lee, and Myoung-Wan Koo. 2023. [Task-optimized adapters for an end-to-end task-oriented dialogue system](#). *ArXiv*, abs/2305.02468.

- Pawel Budzianowski, Iñigo Casanueva, Bo-Hsiang Tseng, and Milica Gasic. 2018. Towards end-to-end multi-domain dialogue modelling.
- Paweł Budzianowski and Ivan Vulic. 2019. Challenges in building intelligent open-domain dialog systems. In *Proceedings of the 2019 International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6429–6433. IEEE.
- Wenhu Chen, Jianshu Chen, Pengda Qin, Xifeng Yan, and William Yang Wang. 2019. Semantically conditioned dialog response generation via hierarchical disentangled self-attention. *arXiv preprint arXiv:1905.12866*.
- Zhiyu Chen, Bing Liu, Seungwhan Moon, Chinnadurai Sankar, Paul A. Crook, and William Yang Wang. 2022. [Ketod: Knowledge-enriched task-oriented dialogue](#). *ArXiv*, abs/2205.05589.
- Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. [Scaling instruction-finetuned language models](#). *ArXiv*, abs/2210.11416.
- 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*, abs/2310.08885.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Saket Dingliwal, Bill Gao, Sanchit Agarwal, Chien-Wei Lin, Tagyoung Chung, and Dilek Z. Hakkani-Tür. 2021. [Few shot dialogue state tracking using meta-learning](#). In *Conference of the European Chapter of the Association for Computational Linguistics*.
- Mihail Eric, Rahul Goel, Shachi H Paul, Abhinav Sethi, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428.
- Ting Han, Chia-Chun Wang, Guan-Lin Chao, Zihan Liu, Ryuichi Takanobu, Hua Xu, and Minlie Huang. 2021. Multiwoz 2.3: A multi-domain task-oriented

732	dialogue dataset enhanced with annotation correc-	Ilya Loshchilov and Frank Hutter. 2017. <a href="#">Decoupled</a>	786
733	tions and co-reference annotation. <i>arXiv preprint</i>	<a href="#">weight decay regularization</a> . In <i>International Confer-</i>	787
734	<i>arXiv:2010.05594</i> .	<i>ence on Learning Representations</i> .	788
735	Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu,	Andrea Madotto, Zhaojiang Lin, Genta Indra Winata,	789
736	Semih Yavuz, and Richard Socher. 2020. Simpletod:	and Pascale Fung. 2021. <a href="#">Few-shot bot: Prompt-</a>	790
737	A simple language model for task-oriented dialogue.	<a href="#">based learning for dialogue systems</a> . <i>ArXiv</i> ,	791
738	In <i>Advances in Neural Information Processing Sys-</i>	abs/2110.08118.	792
739	<i>tems (NeurIPS)</i> .		
740	J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan	Andrea Madotto and Zihan Liu. 2020. <a href="#">Language mod-</a>	793
741	Allen-Zhu, Yuezhi Li, Shean Wang, and Weizhu	<a href="#">els as few-shot learner for task-oriented dialogue sys-</a>	794
742	Chen. 2021. <a href="#">Lora: Low-rank adaptation of large</a>	<a href="#">tems</a> . <i>ArXiv</i> , abs/2008.06239.	795
743	<a href="#">language models</a> . <i>ArXiv</i> , abs/2106.09685.		
744	Vojtech Hudec and Ondrej Dusek. 2023. <a href="#">Are large</a>	Adib Mosharraf, Moghis Fereidouni, and A.B. Siddique.	796
745	<a href="#">language models all you need for task-oriented dia-</a>	2023a. <a href="#">Toward open-domain slot filling via self-</a>	797
746	<a href="#">logue?</a> In <i>SIGDIAL Conferences</i> .	<a href="#">supervised co-training</a> . <i>Proceedings of the ACM Web</i>	798
		<i>Conference 2023</i> .	799
747	Dong hyun Ham, Jeong-Gwan Lee, Youngsoo Jang, and	Adib Mosharraf, Muhammad Hasan Maqbool, and A.B.	800
748	Kyungmin Kim. 2020. <a href="#">End-to-end neural pipeline</a>	Siddique. 2023b. <a href="#">Zero-shot generalizable end-to-end</a>	801
749	<a href="#">for goal-oriented dialogue systems using gpt-2</a> . In	<a href="#">task-oriented dialog system using context summariza-</a>	802
750	<i>Annual Meeting of the Association for Computational</i>	<a href="#">tion and domain schema</a> . <i>ArXiv</i> , abs/2303.16252.	803
751	<i>Linguistics</i> .		
752	Wiradee Imrattananatrat and Ken Fukuda. 2023. <a href="#">End-to-</a>	Kishore Papineni, Salim Roukos, Todd Ward, and Wei-	804
753	<a href="#">end task-oriented dialogue systems based on schema</a> .	Jing Zhu. 2002. <a href="#">Bleu: a method for automatic evalu-</a>	805
754	In <i>Annual Meeting of the Association for Computa-</i>	<a href="#">ation of machine translation</a> . In <i>Annual Meeting of</i>	806
755	<i>tional Linguistics</i> .	<i>the Association for Computational Linguistics</i> .	807
756	Benoit Jacob, Skirmantas Kligys, Bo Chen, Meng-	Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayan-	808
757	long Zhu, Matthew Tang, Andrew Howard, Hartwig	deh, Lars Lidén, and Jianfeng Gao. 2021. <a href="#">Soloist:</a>	809
758	Adam, and Dmitry Kalenichenko. 2018. Quantiza-	<a href="#">Building task bots at scale with transfer learning and</a>	810
759	tization and training of neural networks for efficient	<a href="#">machine teaching</a> . <i>Transactions of the Association</i>	811
760	integer-arithmetic-only inference. In <i>Proceedings of</i>	<i>for Computational Linguistics</i> , 9:807–824.	812
761	<i>the IEEE Conference on Computer Vision and Pat-</i>		
762	<i>tern Recognition (CVPR)</i> , pages 2704–2713.	Baolin Peng, Xijun Li, Jianfeng Gao, Jingjing Liu,	813
		Kam-Fai Wong, and Shang-Yu Su. 2018. Deep dyna-	814
		q: Integrating planning for task-completion dialogue	815
		policy learning. <i>arXiv preprint arXiv:1801.06176</i> .	816
763	Tiziano Labruna, Sofia Brenna, Andrea Zaninello, and	Baolin Peng, Chenguang Zhu, Chunyuan Li, Xijun	817
764	Bernardo Magnini. 2023. <a href="#">Unraveling chatgpt: A crit-</a>	Li, Jinchao Li, Michael Zeng, and Jianfeng Gao.	818
765	<a href="#">ical analysis of ai-generated goal-oriented dialogues</a>	2020. Few-shot natural language generation for task-	819
766	<a href="#">and annotations</a> . In <i>International Conference of the</i>	<a href="#">oriented dialog</a> . <i>arXiv preprint arXiv:2002.12328</i> .	820
767	<i>Italian Association for Artificial Intelligence</i> .		
768	Nhat Le, AB Siddique, Fuad Jamour, Samet Oymak,	Alec Radford, Jeffrey Wu, Rewon Child, David Luan,	821
769	and Vagelis Hristidis. 2021. Predictable and adaptive	Dario Amodei, and Ilya Sutskever. 2019. Language	822
770	goal-oriented dialog policy generation. In <i>2021 IEEE</i>	models are unsupervised multitask learners. <i>OpenAI</i>	823
771	<i>15th International Conference on Semantic Comput-</i>	<i>blog</i> , 1(8):9.	824
772	<i>ing (ICSC)</i> , pages 40–47. IEEE.		
773	Sungjin Lee. 2013. Structured discriminative model for	Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara,	825
774	dialog state tracking. In <i>Proceedings of the SIGDIAL</i>	Raghav Gupta, and Pranav Khaitan. 2019. <a href="#">Towards</a>	826
775	<i>2013 Conference</i> , pages 442–451.	<a href="#">scalable multi-domain conversational agents: The</a>	827
		<a href="#">schema-guided dialogue dataset</a> . In <i>AAAI Confer-</i>	828
		<i>ence on Artificial Intelligence</i> .	829
776	Zekun Li, Wenhu Chen, SHIYANG LI, Hong Wang,	Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018.	830
777	Jingu Qian, and Xi Yan. 2022. <a href="#">Controllable dia-</a>	<a href="#">Towards universal dialogue state tracking</a> . <i>arXiv</i>	831
778	<a href="#">logue simulation with in-context learning</a> . <i>ArXiv</i> ,	<i>preprint arXiv:1810.09587</i> .	832
779	abs/2210.04185.		
780	Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul A.	A.B. Siddique, Muhammad Hasan Maqbool, Kshitija	833
781	Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou	Taywade, and Hassan Foroosh. 2022. <a href="#">Personalizing</a>	834
782	Yu, Andrea Madotto, Eunjoon Cho, and Rajen	<a href="#">task-oriented dialog systems via zero-shot generaliz-</a>	835
783	Subba. 2021. <a href="#">Leveraging slot descriptions for zero-</a>	<a href="#">able reward function</a> . <i>Proceedings of the 31st ACM</i>	836
784	<a href="#">shot cross-domain dialogue statetracking</a> . <i>ArXiv</i> ,	<i>International Conference on Information &amp; Knowl-</i>	837
785	abs/2105.04222.	<i>edge Management</i> .	838

- Haipeng Sun, Junwei Bao, Youzheng Wu, and Xiaodong He. 2022a. [Bort: Back and denoising reconstruction for end-to-end task-oriented dialog](#). In *NAACL-HLT*.
- Haipeng Sun, Junwei Bao, Youzheng Wu, and Xiaodong He. 2022b. [Mars: Semantic-aware contrastive learning for end-to-end task-oriented dialog](#). *ArXiv*, abs/2210.08917.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.
- Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*.
- J. Williams, Antoine Raux, and Matthew Henderson. 2016. [The dialog state tracking challenge series: A review](#). *Dialogue Discourse*, 7:4–33.
- Heng-Da Xu, Xian-Ling Mao, Puhai Yang, Fanshu Sun, and Heyan Huang. 2024. [Rethinking task-oriented dialogue systems: From complex modularity to zero-shot autonomous agent](#). In *Annual Meeting of the Association for Computational Linguistics*.
- Yunyi Yang, Yunhao Li, and Xiaojun Quan. 2020. [Ubar: Towards fully end-to-end task-oriented dialog systems with gpt-2](#). *ArXiv*, abs/2012.03539.
- Xiaoxue Zang, Qi Zhu, Jiachen Zhang, Zheng-Yu Li, Baolin Peng, Xiujuan Li, Liangjun Zhou, Jinchao Li, Jianfeng Gao, and Yun-Nung Chen. 2020. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117.
- Jianguo Zhang, Stephen Roller, Kun Qian, Zhiwei Liu, Rui Meng, Shelby Heinecke, Haiquan Wang, Silvio Savarese, and Caiming Xiong. 2023. [Enhancing performance on seen and unseen dialogue scenarios using retrieval-augmented end-to-end task-oriented system](#). In *SIGDIAL Conferences*.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019a. [Bertscore: Evaluating text generation with bert](#). *ArXiv*, abs/1904.09675.
- Yichi Zhang, Zhijian Ou, and Zhou Yu. 2019b. [Task-oriented dialog systems that consider multiple appropriate responses under the same context](#). In *AAAI Conference on Artificial Intelligence*.
- Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020a. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9604–9611.
- Zheng Zhang, Ryuichi Takanobu, Minlie Huang, and Xiaoyan Zhu. 2020b. [Recent advances and challenges in task-oriented dialog systems](#). *Science China Technological Sciences*, 63:2011 – 2027.
- Jeffrey Zhao, Yuan Cao, Raghav Gupta, Harrison Lee, Abhinav Rastogi, Mingqiu Wang, Hagen Soltau, Izhak Shafran, and Yonghui Wu. 2022. [Anytod: A programmable task-oriented dialog system](#). In *Conference on Empirical Methods in Natural Language Processing*.

Dataset	Model	Overall Response (BLEU-4)				Inform (BLEU-4)				Request (BLEU-4)			
		All	Seen	Mixed	Unseen	All	Seen	Mixed	Unseen	All	Seen	Mixed	Unseen
SGD	SimpleTOD	0.1696	0.1834	0.1877	0.1494	0.1685	0.1790	0.1896	0.1438	0.0228	0.0195	0.0216	0.0243
	SOLOIST	0.1902	0.2798	0.1990	0.1655	0.1813	0.2226	0.1945	0.1568	0.0281	0.0339	0.0265	0.0284
	ZS-ToD	0.0590	0.1413	0.0568	0.0512	0.0255	0.0402	0.0228	0.0246	0.0231	0.0367	0.0221	0.0215
	Auto-ToD	0.0487	0.0523	0.0501	0.0466	0.0854	0.0743	0.0884	0.0851	0.0173	0.0111	0.0159	0.0195
	GPT-2	0.2015	0.2109	0.2229	0.1802	0.2181	0.2421	0.2368	0.1923	0.0400	0.0275	0.0423	0.0403
	Llama-3.2	<b>0.2445</b>	<b>0.2905</b>	<b>0.2568</b>	<b>0.2242</b>	<b>0.2888</b>	<b>0.3180</b>	<b>0.3043</b>	<b>0.2650</b>	<b>0.0641</b>	<b>0.0803</b>	<b>0.0614</b>	<b>0.0634</b>
KETOD	FLAN-T5	0.2110	0.2332	0.2226	0.1961	0.2811	0.3098	0.2911	0.2631	0.0569	0.0625	0.0541	0.0582
	SimpleTOD	0.0821	0.1015	0.0910	0.0538	0.1147	0.1362	0.1268	0.0726	0.0178	0.0266	0.0149	0.0106
	SOLOIST	0.0970	0.1018	0.0945	0.0848	0.0957	0.1185	0.0933	0.0675	0.0167	0.0145	0.0174	0.0185
	ZS-ToD	0.0394	0.0439	0.0254	0.0385	0.0183	0.0231	0.0059	0.0250	0.0260	0.0328	0.0198	0.0243
	Auto-ToD	0.0480	0.0528	0.0492	0.0415	0.0797	0.0678	0.0932	0.0812	0.0134	0.0157	0.0151	0.0092
	GPT-2	0.1890	0.2106	0.1961	0.1524	0.2105	0.2437	0.2078	0.1687	0.0346	0.0500	0.0252	0.0263
KETOD	Llama-3.2	<b>0.2398</b>	<b>0.2864</b>	<b>0.2354</b>	<b>0.1862</b>	<b>0.2701</b>	<b>0.3165</b>	<b>0.2579</b>	<b>0.2208</b>	0.0581	0.0723	<b>0.0508</b>	<b>0.0490</b>
	FLAN-T5	0.2082	0.2351	0.2048	0.1792	0.2727	0.3025	<b>0.2811</b>	<b>0.2234</b>	0.0526	<b>0.0750</b>	0.0454	0.0339

Table 8: BLEU Scores for Overall Response Generation, Inform and Request.

## A Template for multi-task instruction fine-tuning

Figure 2 shows the template used to process the inputs for ZeroToD. The template first informs about the domains involved in the dialog. Next, it provides task-specific instructions and provides the domain schemas. The dialog history is provided and the model is instructed to generate a system response based on the dialog history, search results, and the task instructions.

You are an expert chat assistant for the domains: [domains].

**Instructions:** As an expert, you must generate the most appropriate response for the chat assistant. The response can be an api call or a response to the user.

Based on the **Last User Utterance**, you must find the relevant **Intent** from the **Schema** and your request should use the **required slots** and **optional slots** from that **Intent**.

You will be provided with the Schema for domains: [domains]

[schemas]

You will be provided an incomplete dialog between a user and a chat assistant, and an optional search results.

**Dialog History:** [dialog history]

Using the **Dialog History**, **Search Results**, and by following the **Instructions** please generate the response for the chat assistant.

Figure 2: Multi-task instruction finetuning template. Items in blue are dynamic elements and those in purple are important aspects of the prompt.

## B BLEU Scores for Response Generation

Table 8 presents additional metrics for response generation. BLEU-4 scores are reported for baselines (SimpleTOD, SOLOIST, ZS-ToD, Auto-ToD), and ZeroToD models (GPT-2, Llama-3.2, FLAN-T5) with schema augmentations. We see a similar trend here as well, with ZeroToD models outperforming baseline approaches. However, BLEU-4 scores are better for Llama-3.2 than FLAN-T5, particularly for the seen domains. Since the BLEU-4 metric is calculated by n-gram matches, Llama-3.2 having better supervised performance tends to generate responses closer to the ground truth, thus yielding higher BLEU-4 scores.

## C Dialog Examples

Table 9 shows an example dialog in the Restaurant domain. The table contains the turn id, user utterance, gold response, SOLOIST, Auto-ToD, GPT-2, Llama-3.2, and FLAN-T5 response. Text highlighted in



red outlines the portions where the system response by a model is incorrect and green highlights the correct parts. Texts highlighted in orange indicate that the model is partially correct and is missing some information.

In the example dialog, we can see that the responses of Auto-ToD is longer and more descriptive, whereas ZeroToD models produce succinct responses, similar to the gold standards. This observation indicates that ZeroToD was able to pick up the text generation style of the gold standard, which is an important aspect for ToD systems.

The baseline SOLOIST fails to generate a valid response on a few turns. This stems from the complex structure of the output, which consists of dialog states, actions and system response. The output structure has a certain format and different parts are identified with special tokens. On many occasions, the model does not produce the correct tokens, thus leading to invalid generations.

On turns where the user asks about multiple information, the GPT-2 and SOLOIST model struggles to respond to all the user requests. However, all the other models do not have this problem. This could be attributed to the fact that GPT-2 models are the smallest model when compared to all the other models.

Llama-3.2 and FLAN-T5 generates responses that are very similar to the gold standard. These models are able to pickup the generation style and can respond to multiple requests of the user. In the example dialog, the user provided information about the restaurant name, time and date on turn 1, but then changed the restaurant name on turn 2. Llama-3.2 and FLAN-T5 models were able to identify what changed and moved forward with the API Call by using all the correct information. However, Auto-ToD was not able to keep track of all the changes and made an incorrect API Call. SOLOIST was able to make a partially correct API Call since it was not able to generalize to the number\_of\_seats slots. It used the party\_size slot that it had seen during training.

This dialog example showed some of the drawbacks of baseline approaches and some example cases of where they struggle. For ZeroToD models, the limitations of GPT-2 was evident, as it was not able to process multiple requests of the user. However, Llama-3.2 and FLAN-T5 had performance similar to the gold standard across multiple turns, which falls in line to our automatic metrics and human evaluation.

Table 9: Dialog Example—SGD dataset. Dialog id 1\_00001

Model	Response—SGD dataset. Dialog id 1_00001
Turn	1
User	Can you book a table for me at the Ancient Szechuan for the 11th of this month at 11:30 am?
Gold	In which city are you trying to book the table?
SOLOIST	Please confirm the following details: Booking a table for 2 at Ancient Szechuan in San Francisco on March 11th at 11:30 am.
Auto-ToD	Could you please provide the city where the restaurant ‘Ancient Szechuan’ is located and the number of seats you would like to reserve?
GPT-2	Sure, I have a table for 2 at Ancient Szechuan in San Francisco at 11:30 am today.
Llama-3.2	What is the location of the restaurant?
FLAN-T5	Where should I look for the restaurant?
Turn	2
User	Can you book a table at the Butterfly restaurant in San Francisco?
Gold	Please confirm that you need a table for 2 at the Butterfly Restaurant in San Francisco at 11:30 am on March 11th.
SOLOIST	Can you please confirm that you want to book a table for 2 at the Butterfly restaurant in San Francisco at 11:30 am on March 11th?
Continued on next page...	

Dataset	Response—SGD dataset. Dialog id 1_00001
Auto-ToD	Could you please confirm the date and time for the reservation at the Butterfly restaurant in San Francisco? Additionally, let me know the number of seats you would like to reserve.
GPT-2	Can you please confirm that you want to book a table for 2 people at the Butterfly restaurant in San Francisco at 11:30 am on March 11th?
Llama-3.2	Can you confirm the following details: You want to book a table at the butterfly Hotel in San Francisco at 11:30 am. You want 2 seats. You want to book the table on March 11th.
FLAN-T5	You wish to book a table for 2 at the Butterfly Restaurant in San Francisco for March 11th at 11:30 am. Is this correct?
Turn	3
Gold	ApiCall(method='ReserveRestaurant', parameters='date': '2019-03-11', 'location': 'San Francisco', 'number_of_seats': '2', 'restaurant_name': 'Butterfly Restaurant', 'time': '11:30' )
SOLOIST	ApiCall(method='ReserveRestaurant', parameters='city': 'San Francisco', 'date': '2019-03-11', 'party_size': '2', 'restaurant_name': 'The Butterfly Restaurant', 'time': '11:30')
Auto-ToD	ApiCall(method='FindRestaurants', parameters='category': 'Butterfly', 'location': 'San Francisco')
GPT-2	ApiCall(method='ReserveRestaurant', parameters='date': '2019-03-11', 'location': 'San Francisco', 'number_of_seats': '2', 'restaurant_name': 'The Butterfly Restaurant', 'time': '11:30')
Llama-3.2	ApiCall(method='ReserveRestaurant', parameters='date': '2019-03-11', 'location': 'San Francisco', 'number_of_seats': '2', 'restaurant_name': 'Butterfly Restaurant', 'time': '11:30')
FLAN-T5	ApiCall(method='ReserveRestaurant', parameters='date': '2019-03-11', 'location': 'San Francisco', 'restaurant_name': 'Butterfly Restaurant', 'number_of_seats': '2', 'time': '11:30' )
Turn	4
User	That's perfect. What cuisine do they offer?
Gold	Your table has been booked successfully and they serve Asian cuisine.
SOLOIST	Your table has been booked.
Auto-ToD	It looks like the reservation at the Butterfly Restaurant in San Francisco has been successfully made for March 11th at 11:30 am for 2 people. The restaurant offers Asian cuisine. If you need any more assistance or have any other questions, feel free to ask. Have a great day! Goodbye!
GPT-2	Your table is booked.
Llama-3.2	Your reservation has been successful. They offer Asian cuisine.
FLAN-T5	Your reservation has been made. They serve Asian food.
Turn	5
User	Can you get me the user rating of the restaurant? Is it expensive?
Gold	Their prices are moderate and they have a user rating of 4.0.
SOLOIST	✗ (Failed to generate a valid response)
Auto-ToD	The Butterfly Restaurant in San Francisco has a user rating of 4.00. The price range is moderate, so it's not considered expensive. If you have any more questions or need further assistance, feel free to ask!
GPT-2	The rating is 4.0.
Continued on next page...	

Dataset	Response—SGD dataset. Dialog id 1_00001
Llama-3.2	The restaurant has a 4.0 rating and is considered moderate in price.
FLAN-T5	They have a moderate price range and a 4.0 rating.
Turn	6
User	Thanks a bunch!
Gold	Is there anything else I can do for you?
SOLOIST	✗ (Failed to generate a valid response)
Auto-ToD	You're welcome! If you need any more assistance, feel free to ask. Have a great day!
	Goodbye!
GPT-2	Is there anything else I can help you with?
Llama-3.2	Do you need anything else?
FLAN-T5	Do you need help with anything else?
Turn	7
User	No. Thank you. That's all I need.
Gold	Have a great day!
SOLOIST	Have a great time at the restaurant!
Auto-ToD	You're welcome! If you need any more assistance in the future, feel free to reach out.
	Have a great day! Goodbye!
GPT-2	Have a good one!
Llama-3.2	Have a great day.
FLAN-T5	Have a nice day.

## D User Study Instructions

### Disclaimers of any risks to participants or annotators

There are no significant risks associated with participating in this study. However, annotators may experience mild fatigue or cognitive strain due to prolonged reading and evaluation of multiple conversations. If you feel discomfort or fatigue, please take breaks as needed.

### Instructions for Human Study Participants

Your task is to evaluate model-generated responses in multi-turn, task-oriented conversations based on the following criteria:

1. Fluency
2. Informativeness
3. Task Completion

### Task Overview

- You will be presented with multiple conversations, where a user interacts with a model to complete a task (e.g., booking a flight).
- Your job is to rate each model response independently using a 1-5 scale based on the provided criteria.
- This study is blind — you will not know which model produced which response.

### How to Rate Responses

You will assign a score for each response based on the following:

## 1. Fluency

Measures whether the response is grammatically correct, well-structured, and natural to read.

- 1 = Unnatural, grammatically incorrect, or hard to read.
- 5 = Perfectly fluent, natural, and error-free.

### Example (Good Fluency)

User: I need to book a flight to New York next Monday.

Model: Sure! What time would you like to depart?

### Example (Poor Fluency)

User: I need to book a flight to New York next Monday.

Model: Sure! You next Monday flight want time what?

## 2. Informativeness

Measures whether the response provides useful and relevant information to advance the task.

- 1 = Vague, incorrect, or unhelpful.
- 5 = Precise, relevant, and useful.

### Example (Good Informativeness)

User: Can you recommend a vegetarian restaurant nearby?

Model: Yes! Green Leaf Café is a highly rated vegetarian restaurant.

### Example (Low Informativeness)

User: Can you recommend a vegetarian restaurant nearby?

Model: There are many restaurants in your area.

## 3. Task Completion

Measures whether the response successfully progresses or completes the task in the conversation.

- 1 = Fails to address the request.
- 5 = Fully completes the task.

### Example (Good Task Completion)

User: I'd like to book a table for two at an Italian restaurant at 7 PM today.

Model: I found a reservation at Bella Italia for 7 PM. Would you like me to reserve it for 2 people at 7 PM today?

### Example (Failed Task Completion)

User: I'd like to book a table for two at an Italian restaurant at 7 PM.

Model: There are Italian restaurants in your area.

## Additional Guidelines

- Be objective — rate based on quality, not personal preference.
- If a response is unclear or ambiguous, leave a comment.
- Do not try to guess which model produced the response.



### **Study Duration & Submission**

998

- The study will take approximately 10 minutes to complete.
- Once you have evaluated all responses, submit your ratings.

999

1000

**Thank you for your time and valuable feedback!**

1001