Call, Reward, Repeat: Advancing Dialog State Tracking with GRPO and Function Calling

Timur Ionov[1,2]*, Anna Marshalova[1]*, Valentin Malykh[1,2,3]

[1]MWS AI, [2]ITMO University, [3]IITU University

Correspondence: t.ionov@mts.ai

Abstract

Recent advancements in Large Language Models (LLMs) have notably enhanced task-oriented dialogue systems, particularly in Dialogue State Tracking (DST), owing to their generative capabilities and strong generalization. Although recent approaches such as LDST and FnCTOD significantly improved cross-domain DST performance via supervised fine-tuning (SFT), these methods typically require substantial amounts of domain-specific data. In this paper, we address this limitation by employing Group Relative Policy Optimization (GRPO) - a critic-free reinforcement learning method that efficiently guides LLMs toward improved DST accuracy even under low-resource conditions. Our results on established DST benchmarks, including MultiWOZ 2.1 and 2.4, demonstrate that the RL approach achieves superior performance to existing methods while using significantly reduced out-of-domain training data. In addition, we found out that models pretrained specifically for tool-use tasks can be a better starting point, especially on small scales.

1 Introduction

Task-oriented dialogue (TOD) systems serve as critical facilitators in domains ranging from travel planning to technical support. Central to these systems is Dialogue State Tracking (DST), which maintains a persistent representation of user constraints.

Recent advances in Large Language Models (LLMs) have transformed DST by enabling structured state generation directly from context, specifically through a function calling paradigm, where dialogue states are expressed as explicit function invocations with named arguments (Li et al., 2024). While this formulation enhances performance, existing approaches still largely depend on supervised fine-tuning or prompting. Crucially, these methods optimize for probabilistic likelihood rather than logical correctness. Conversely, Reinforcement Learning (RL) allows for the direct optimization of verifiable metrics. In DST-as-function-calling, predicted arguments can be deterministically matched against ground truth, providing a framework for verifiable rewards. This paper addresses how RL can be effectively applied to this paradigm to achieve robust performance at scale.

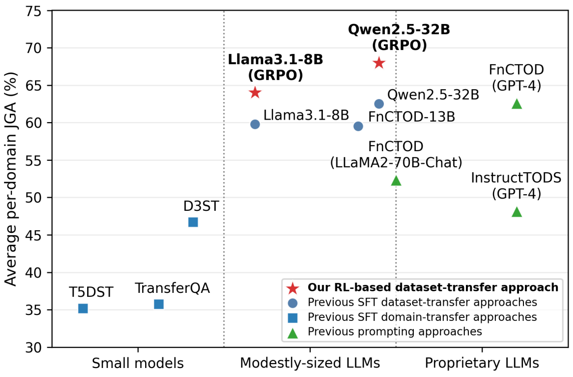


Figure 1: DST performance comparison among (1) previous SFT domain-transfer approaches; (2) previous SFT dataset-transfer approaches; (3) prompting approaches relying on advanced proprietary and large-scale LLMs; and (3) our RL-based approach with various LLMs on MultiWOZ 2.1.

Our contributions can be summarized as follows.

- We present a reinforcement learning recipe for DST-as-function-calling using GRPO with a fully verifiable reward over function calls, improving Joint Goal Accuracy (JGA) over matched SFT baselines without training a separate critic/value model.
- We evaluate cross-dataset transfer to MultiWOZ 2.1 and 2.4 across model families and scales (1.5B–32B), and show consistent gains from GRPO, especially under limited per-domain data budgets.
- We study the impact of tool-use/function-

---

*Equal contribution

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 4: Student Research Workshop, pages 292–303
March 26, 2026 ©2026 Association for Computational Linguistics

calling pretraining on DST transfer, showing it is particularly beneficial for smaller models and that GRPO reduces sensitivity to initialization.

Our implementation is publicly available at the following repository: `https://github.com/sir-timio/CallRewardRepeat`.

2 Related Work

Supervised DST before LLMs Initial DST approaches treated the task as a supervised classification or span extraction problem over fixed ontologies, relying heavily on domain-specific annotations. Methods such as TRADE (Wu et al., 2019), TripPy (Heck et al., 2020), and SUMBT (Lee et al., 2019) introduced copy mechanisms, BERT-based encoders, and ontology-aware decoding. Although subsequent extensions, including DS-DST (Zhang et al., 2020), MetaASSIST (Ye et al., 2022b), and paDST (Ma et al., 2019), improved robustness and scalability, these models remained limited in cross-domain generalization and required extensive schema-aligned training data.

Supervised Finetuning of LLMs for DST Many recent studies explore supervised fine-tuning (SFT) of generative models for DST. Existing approaches fine-tune T5, GPT, LLaMA, and similar architectures for DST using either slot-based or structured output representations (Lin et al., 2021a,b; Zhao et al., 2022; Feng et al., 2023; Hosseini-Asl et al., 2022; Wang et al., 2024; Carranza and Rojas, 2025).

To improve generalization and alignment with downstream tasks, several works explore various prompting strategies. For instance, SimpleTOD (Hosseini-Asl et al., 2022) and LDST (Feng et al., 2023) employ natural language prompts to guide the model toward producing structured state representations. FnCTOD (Li et al., 2024) advances this direction by casting DST specifically as function calling. While, in standard DST, models are trained to output specialized text sequences representing constraints (e.g., `restaurant-food: italian, restaurant-area: center`), function calling treats the domain ontology as a code interface: user intents are mapped to function names and constraints are generated as formal arguments e.g., `find_restaurant(food='italian', area='center')`. This formulation leverages the pre-trained code-generation capabilities of LLMs to enforce stricter schema adherence. This function-call-oriented prompting scheme substantially improves zero-shot performance and enables better compositional generalization across domains.

Yet, despite the structural benefits offered by function-style formulation, the fundamental reliance on supervision remains a limiting factor. Consequently, the generalization capacity of these methods is strictly bounded by the diversity and volume of annotated data available.

RL for DST and Tool-Use Reinforcement learning (RL) is a powerful alternative to supervised fine-tuning of LLMs. Recent studies show that RL fine-tuning leads to stronger generalization, particularly on out-of-distribution tasks, while SFT often results in memorization of training data (Chu et al., 2025).

This generalization capability makes RL especially effective for DST, where models must robustly track evolving user intents across turns. For instance, TOATOD (Su et al., 2022) introduces lightweight adapters trained with REINFORCE on JGA-based reward. Approaches like Deep Dyna-Q (Peng et al., 2018) leverage simulated user environments to improve policy robustness, while AURL (Zhang et al., 2023) combines asynchronous updates, curriculum learning, and user simulation to reduce error propagation. Fine-grained reward shaping (Du et al., 2024) further allows targeted optimization.

RL is also extensively applied to function calling and tool use, enabling LLMs to interact effectively with external APIs and resources. StepTool (Yu et al., 2025) rewards individual tool steps, iTool (Zeng et al., 2025) uses iterative fine-tuning with Monte Carlo Tree Search (MCTS) and combined outcome/self-evaluation rewards, and ReTool (Feng et al., 2025) employs binary rewards for code correctness.

An increasingly popular RL method used for tool calling is Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which estimates advantages by sampling the policy multiple times and using the average reward, thereby eliminating the need for a separate value function. This makes it computationally efficient and particularly suitable for tool use training, as rewards in this domain are often verifiable and easily computable based on factors like tool invocation success, format correctness, and outcome accuracy. It has been applied in frameworks like Nemotron-Research-Tool-N1

(Zhang et al., 2025), Reflect, Retry, Reward (Bensal et al., 2025), and ToolRL (Qian et al., 2025) to improve tool selection and execution.

```
You are a task-oriented assistant. You can use
the given functions to fetch further data to
help the users.
"name": "find_book_hotel",
"description": "hotel reservations and vacation
stays.",
"arguments": [
{
"name": "name",
"type": "string",
"description": "name of the hotel"
},
{
"name": "pricerange",
"type": "string",
"description": "price budget of the hotel",
"possible_values":    ["expensive",   "cheap",
"moderate"]
}
...
]
```

Figure 2: Example of a system prompt containing a JSON schema of available functions for the assistant

In summary, RL improves LLM generalization for DST and enables reliable performance in tool-use scenarios. By leveraging reward-driven training, RL enables models to adapt dynamically and robustly. Methods like GRPO simplify training and make RL computationally efficient for these applications.

3 Methodology

3.1 Dataset construction

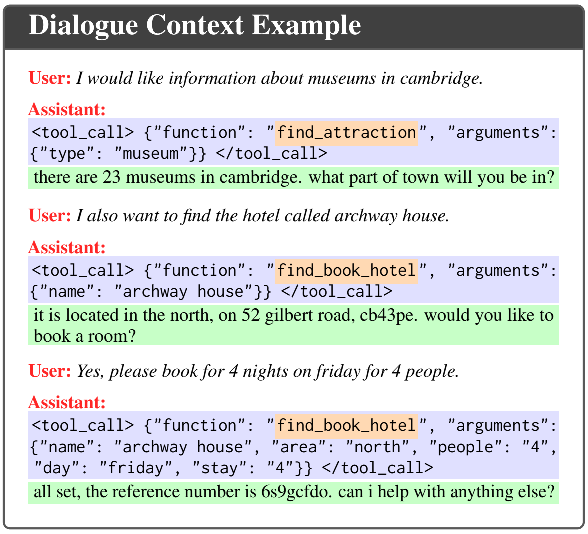
System Prompt. Prompt construction and setup task of DST as function calling are fully inherited from prior work (Li et al., 2024). System prompt of each dialogue session contains general task instructions and structured specifications in a JSON schema of relevant functions. An example is provided in Figure 2.

Dialogue. In the dialogues comprising the dataset, each user turn is followed by the assistant's response, which may contain a function call and a natural language answer shown in Figure 3.

Dialogue Sampling. To ensure balanced representation of function calls in the dataset, we employ a stratified deterministic (controlled by a seed) sampling approach, selecting $PDsize$ (per-domain) dialogues per each of 35 unique functions.

3.2 Training

SFT Phase. The model is trained using next-token prediction, but the loss is computed only over tokens inside tool call spans. Since the model observes user queries but not the results of database calls, supervising natural language responses referencing unseen outputs would be misaligned and potentially harmful.



Dialogue Context Example

User: *I would like information about museums in cambridge.*

Assistant:

`<tool_call> {"function": "find_attraction", "arguments": {"type": "museum"}} </tool_call>`

there are 23 museums in cambridge. what part of town will you be in?

User: *I also want to find the hotel called archway house.*

Assistant:

`<tool_call> {"function": "find_book_hotel", "arguments": {"name": "archway house"}} </tool_call>`

it is located in the north, on 52 gilbert road, cb43pe. would you like to book a room?

User: *Yes, please book for 4 nights on friday for 4 people.*

Assistant:

`<tool_call> {"function": "find_book_hotel", "arguments": {"name": "archway house", "area": "north", "people": "4", "day": "friday", "stay": "4"}} </tool_call>`

all set, the reference number is 6s9gcfdo. can i help with anything else?

Figure 3: Example of a dialogue context with multiple tool calls. Tool calls are highlighted with blue and natural language responses with green colors.

GRPO Phase. This phase uses Group Relative Policy Optimization (GRPO) (Shao et al., 2024) — a memory- and compute-efficient reinforcement learning method for fine-tuning LLMs. For each prompt s_j, the model generates G candidate responses $\{a_{jk}\}_{k=1}^{G}$ and obtains their rewards $\{r_{jk}\}_{k=1}^{G}$ with a verifiable reward-function. The group mean reward is defined as $\bar{r}_j = \frac{1}{G}\sum_k r_{jk}$, and each response's advantage is computed as the difference between its reward and the group's average reward, which can be expressed as $A_{jk} = r_{jk} - \bar{r}_j$. The policy is updated using a PPO-style clipped surrogate objective (Schulman et al., 2017) with KL regularization, which encourages increasing the probability of higher-advantage responses while preventing overly large policy updates.

Crucially, GRPO avoids the need for a separate critic network by leveraging within-group statistics for baseline estimation, leading to significant savings in memory and computational overhead compared to PPO while retaining PPO's stability via clipping and KL penalties.

For GRPO training, we reuse the exact same dialogues from the SFT corpus, slicing them turn-by-turn into multiple prompt-answer pairs. This ensures consistency in data distribution and allows for direct comparison between training strategies. To reduce computational cost during GRPO training, we sample only a single prompt-response pair (slice) per dialogue. The slice is selected from a pool of slices containing function calls using a reversed Poisson-skewed sampling strategy with $\lambda = 2$, which biases selection toward near last turns, based on the intuition that these contain richer and more contextually grounded supervision signals. Sampling function pseudocode is shown in Algorithm 1.

Reward details. We define three cases based on tool-call presence: (i) both prediction and gold contain a tool call; (ii) exactly one contains a tool call; (iii) neither contains a tool call. Case (ii) receives a reward of -1.0 to penalize both over-calling and under-calling. Case (iii) receives a reward of $+1.0$ (correct abstention). In case (i), we compute a reward using one of two matching strategies (full-match or partial-match), described below, after canonicalizing function names and argument keys/values.

Full-match: reward $= 1.0$ if the function name matches and the canonicalized argument dictionaries are exactly equal; otherwise 0.0.

Partial-match: if the function name matches, reward is the fraction of gold arguments whose key and value are correctly predicted: $r = \frac{|\{k \in \text{keys}(\mathbf{a}^{gold}) : \mathbf{a}^{pred}[k] = \mathbf{a}^{gold}[k]\}|}{|\text{keys}(\mathbf{a}^{gold})|}$ (extra predicted arguments that are not in the gold do not increase reward). If the function name does not match, $r = 0.0$.

4 Experiments and Results

4.1 Datasets

We adopt the cross-dataset training setup introduced in Li et al. (2024), using the same diverse mix of task-oriented dialogue datasets for fine-tuning: WOZ 2.0 (Mrkšić et al., 2017), CamRest676 (Wen et al., 2017), MSR-E2E (Li et al., 2018), TaskMaster (Byrne et al., 2019), and Schema-Guided Dialogue (SGD) (Rastogi et al., 2020). Together, these datasets cover 37 domain instances, as detailed in Table 1. Each domain is further represented as a distinct function.[1]

Algorithm 1 Poisson-Skewed Sampling of a Dialogue Slice

Input: `slices` — list of dialogue (prompt, response) pairs, $\lambda > 0$
Output: A single sampled slice from the dialogue

```
$n \leftarrow$ len(slices)
for $i \in [0, 1, \ldots, n-1]$ do
    $probs[i] \leftarrow \frac{e^{-\lambda} \cdot \lambda^i}{i!}$
end for
reverse($probs$)        // bias toward later turns
$probs \leftarrow$ normalize($probs$)
$selected \leftarrow$ random.choices(slices, weights=probs)
return $selected$
```

Algorithm 2 Reward modeling

Require: Generated completions $C = \{c_i\}$, ground truth answers $A = \{a_i\}$

```
for $i = 1$ to $|C|$ do
    $fn_c \leftarrow$ parse_fn($c_i$)
    $fn_a \leftarrow$ parse_fn($a_i$)
    if ($fn_c = \emptyset$ xor $fn_a = \emptyset$) then
        $r_i \leftarrow -1.0$
    else
        $r_i \leftarrow$ match_fn($fn_c, fn_a$)
    end if
end for
return $\{r_i\}$
```

For evaluation, we use the standard test splits of MultiWOZ 2.1 (Eric et al., 2020) and its latest version, MultiWOZ 2.4 (Ye et al., 2022a). The test set in both versions contains 1,000 dialogues in five domains. MultiWOZ 2.1 is widely used as an established benchmark for dialogue state tracking, while MultiWOZ 2.4 offers further improvements in annotation quality and reduced noise in slot value annotations, providing a more reliable benchmark for evaluating state-of-the-art DST methods.

This cross-dataset transfer setup enables us to assess zero-shot generalization to unseen domains from unseen datasets, while the MultiWOZ variants allow evaluation of performance on complex multi-domain dialogues with varying levels of annotation quality.

4.2 Metrics

We evaluate our models using a standard metric for dialogue state tracking: Joint Goal Accuracy (**JGA**), which measures the proportion of turns in which the predicted dialogue state exactly matches

---

[1] For the SGD dataset, we use services as the domains since, e.g., Music_1 and Music_2 contain partially different function sets.

Dataset	**Domains**	**#Domains**	**#Dialogues**
	Training		
SGD	RentalCars_1, RentalCars_2, Buses_1, Buses_2, Events_1, Events_2, Services_1, Services_2, Services_3, Media_1, RideSharing_1, RideSharing_2, Travel_1, Hotels_1, Hotels_2, Hotels_3, Flights_1, Flights_2, Restaurants_1, Calendar_1, Music_1, Music_2, Weather_1, Movies_1, Homes_1, Banks_1	26	16,000
TaskMaster	Pizza_Ordering, Movie, Auto_Repair, Taxi, Coffee_Ordering, Restaurant	6	13,215
MSR-E2E	Restaurant, Movie, Taxi	3	10,087
WOZ 2.0	Restaurant	1	1,200
CamRest676	Restaurant	1	680
	Evaluation		
MultiWOZ 2.1/2.4	Restaurant, Hotel, Attraction, Train, Taxi	5	1,000

Table 1: Overview of the DST corpora utilized for fine-tuning (37 domains) and evaluation (5 domains). This table details the datasets along with their specific domains, the number of domains included in each dataset, and their size in terms of dialogues.

the ground truth across all domains. We report both Overall JGA, calculated over all turns and domains, and Average JGA, defined as the macro-average of JGA across individual domains.

4.3 State Extraction

We evaluate using Joint Goal Accuracy (JGA) on MultiWOZ by converting each predicted tool call into a turn-level belief-state update and then accumulating updates over the dialogue. Following the function-calling formulation, each function corresponds to a domain, and each argument key corresponds to a slot name within that domain. The function schema used for evaluation is show in Table 2.

Parsing. We extract the first `<tool_call>` ... `</tool_call>` span (if present) and parse the enclosed JSON into a tuple $(f, \mathbf{a})$ consisting of a function name f and an argument dictionary $\mathbf{a}$. If parsing fails or no tool-call span is present, we treat the prediction as “no tool call” for that turn.

State update. Let S_{t-1} be the accumulated belief state before turn t and $(f_t, \mathbf{a}_t)$ be the parsed prediction at turn t. If a tool call is present, we update S_t by overwriting the slots specified in $\mathbf{a}_t$ (slots not mentioned remain unchanged). If no tool call is present, we set $S_t = S_{t-1}$. We canonicalize slot keys and values with simple normalization (e.g., lowercasing and whitespace normalization) and map dataset-specific special values (e.g., `dontcare`) to a single form.

Normalization. Before comparison, we canonicalize slot keys and values with simple normalization (e.g., lowercasing and whitespace normalization) and map dataset-specific special values (e.g., `dontcare`) to a single form. We apply the same canonicalization to gold states.

JGA. At each turn t, JGA counts a hit if S_t exactly matches the gold belief state G_t across all active domains/slots. We report both Overall JGA (micro over all turns) and Average JGA (macro over domains).

4.4 Baselines

We compare our approach against three distinct groups of baselines: (1) Cross-domain transfer TransferQA (Lin et al., 2021a), T5DST (Lin et al., 2021b), D3ST (Zhao et al., 2022) with leave-one-domain-out training on MultiWOZ; (2) Different prompting techniques of strong proprietary models; (3) Cross-dataset transfer finetuned LLMs using LDST (Feng et al., 2023) and FnCTOD (Li et al., 2024) prompting techniques.

MultiWOZ domain	Function	Arguments (slots)
Restaurant	`find_restaurant`	area, day, food, name, people, pricerange, time
Hotel	`find_book_hotel`	area, day, internet, name, parking, people, pricerange, stars, stay, type
Attraction	`find_attraction`	area, name, type
Train	`find_train`	arrive, day, departure, destination, leave, people
Taxi	`find_taxi`	arrive, departure, destination, leave

Table 2: Function schema used for MultiWOZ evaluation. Each argument corresponds to a belief-state slot.

Model	**Size**	**Method**	**MultiWOZ 2.1**		**MultiWOZ 2.4**	
			Average	**Overall**	**Average**	**Overall**
Cross-domain Transfer approaches						
T5DST	60M	SFT	35.20	–	–	–
TransferQA	770M		35.77	–	–	–
D3ST	11B		46.70	–	–	–
Prompting/In-context learning approaches						
InstructTODS$_{GPT-4}$	–	Zero-shot	48.16	–	–	–
FnCTOD$_{LLaMA-2}$	70B	Few-shot	52.36	28.38	–	–
FnCTOD$_{GPT-4}$	–	Zero-shot	62.59	38.71	–	–
Cross-dataset Transfer approaches						
LDST$_{LLaMA}$	7B	SFT	–	–	–	31.6
FnCTOD$_{LLaMA-2}$	13B		59.54	37.67	–	–
LLaMA-3.1	8B		59.87±0.9	37.67±1.0	62.74±1.1	41.27±1.2
Qwen-2.5	32B		62.39±1.5	39.9±2.0	65.93±1.6	44.75±2.0
Our RL-based Cross-dataset Transfer approach						
LLaMA-3.1	8B	GRPO	*63.96±1.0*	*41.89±1.4*	*67.26±1.6*	*45.75±2.2*
Qwen-2.5	32B		**67.98±1.2**	**46.53±1.6**	**71.69±1.5**	**52.0±2.6**

Table 3: Joint Goal Accuracy (JGA) of various models and training strategies on MultiWOZ 2.1 and 2.4 benchmarks. We report both domain-average (macro) and overall JGA for each dataset (both in %). Few-shot prompting was performed with 5 examples. Baseline metrics are taken directly from the corresponding publications. Results, averaged over 5 runs and reported as mean ± std, highlight consistent improvements of our method across all evaluated settings, with all GRPO results showing statistically significant gains over their corresponding SFT baselines ($p < 0.05$, Welch's t-test, $n = 5$).

4.5 Evaluated Models

As the starting points for our main experiments, we used two model families. First, **LLaMA-3.1-8B-Instruct** (Grattafiori et al., 2024), an 8-billion-parameter instruction-tuned model optimized for chat and tool use. Second, the **Qwen-2.5-Instruct** family across multiple parameter scales: 1.5, 3, 7, 14, and 32 billions, to study scaling and data efficiency. To investigate the effect of function-calling pretraining on DST transfer, we additionally evaluate specialized variants at matched scales; these are introduced in §4.7.3.

4.6 Training and Evaluation Setup

For both supervised and reinforcement learning experiments, we adopted a highly efficient training and inference pipeline. Model finetuning was performed using the Unsloth framework (Han et al., 2023) and LoRA (Hu et al., 2021) adapters with rank 16 applied to all attention layers, enabling efficient adaptation while minimizing memory and storage requirements. For accelerated generation in the GRPO phase, we employed the vLLM (Kwon et al., 2023) engine, allowing for scalable and memory-efficient batched decoding and the TRL (von Werra et al., 2020) library. For

GRPO-based training, we generated 8 completions with temperature 1.0 per prompt during optimization. During inference, we set the decoding top_k at 0.2 and temperature to 0.01 to ensure stability and reproducibility of results. The fine-tuning was conducted on a single H100 80GB GPU. Detailed hyperparameter settings and further implementation specifics are provided in Tables 5 and 6 in Appendix A.

4.6.1 Performance Comparison on DST Benchmarks

Table 3 and Figure 1 present a comprehensive comparison of our approach against state-of-the-art cross-dataset and cross-domain methods across benchmarks. More detailed results with metrics for each domain are presented in Table 4.

Our RL-based training recipe, leveraging GRPO, achieves the best overall performance on both MultiWOZ 2.1 and 2.4, substantially outperforming all previous models. Notably, LLaMA-3.1 with only 8B parameters trained with GRPO surpass not only previous cross-domain transfer baselines, but also the much larger 32B SFT baselines and GPT-4, achieving improvements of over 3 percentage points in Overall JGA on MultiWOZ 2.1.

In summary, our results establish a new state-of-the-art for dataset-transfer DST, demonstrating that reinforcement learning with GRPO not only closes the gap with much larger models, but can also reliably outperform established prompting and supervised transfer paradigms.

4.7 Analysis of Training Regimes

4.7.1 Impact of Domain-specific Training Data Volume

Figure 4 illustrates the relationship between the volume of domain-specific training data and model performance across various training methods. PD size of n stands for per-domain size and means that we collect n dialogues for each domain in mixed corpora 1. GRPO consistently surpasses SFT across most evaluated data regimes, often by a significant margin. Notably, GRPO achieve better performance than standalone SFT using as little as 20 per-domain examples for Qwen-2.5-3B model. It is also worth noting that xLAM-2-3B-fc-r, a function-calling variant of Qwen-2.5-3B (described in §4.7.3), exhibits a stronger performance trend with training size expansion across all methods.

4.7.2 Effect of Model Scale

Figure 6 shows that increasing model size consistently improves DST performance on MultiWOZ 2.4 across all settings (zero-shot, SFT, and GRPO). Importantly, our GRPO-based RL approach maintains a clear advantage over SFT at every scale, and this margin increases as models grow, becoming most pronounced at 14B and 32B. This indicates that GRPO leverages additional model capacity more effectively for optimizing structured DST behavior than supervised fine-tuning alone. The SFT+GRPO hybrid closely tracks GRPO, suggesting that the RL phase is the primary driver of the scaling gains.

4.7.3 Influence of Function-Calling Pretraining and Model Initialization

To investigate the impact of function-calling (FC) pretraining on DST, we pair each base instruct model with a FC-specialized counterpart at matched scale. For LLaMA 3.1 at the 8B scale, we use **ToolACE-2-8B** (Liu et al., 2025), a variant further fine-tuned with the ToolACE framework, which synthesizes tool-calling dialogues over 26,500+ APIs with dual rule/model verification, yielding strong performance on benchmarks such as BFCL (Patil et al., 2023). For the Qwen 2.5 family, we use **xLAM-2-fc-r** variants (Zhang et al., 2024), trained on synthesized multi-domain tool-calling trajectories and performing strongly on τ-bench (Yao et al., 2024) and BFCL (Patil et al., 2023).

Figure 7 reports the marginal effect of FC initialization on DST performance by comparing the best-achieved JGA scores between these specialized models and their base instruct counterparts across matched parameter scales.

4.7.4 Comparative Analysis of Training Approaches

When analyzing hybrid training methods (SFT&GRPO) (see Fig. 4), we observe distinct behaviors based on model size. For smaller (3B) models, SFT&GRPO initially offers performance benefits under limited-data conditions; however, its advantage diminishes as the training set grows, ultimately yielding to superior pure GRPO performance. Conversely, for larger (8B) models, SFT&GRPO consistently lags behind GRPO alone, except in the 400 dialogues per domain training setup, although it remains superior to SFT alone. Overall, GRPO demonstrates remarkable

Model	Size	Training Method	Attraction		Hotel		Taxi		Train		Restaurant		JGA	
			JGA	F1	JGA	F1	JGA	F1	JGA	F1	JGA	F1	Avg.	O/a
Cross-domain Transfer approaches														
T5DST	60M	SFT	33.09	–	21.21	–	21.65	–	64.62	–	35.43	–	35.20	–
TransferQA	110M		31.25	–	22.72	–	26.28	–	61.87	–	36.72	–	35.77	–
D3ST	11B		56.40	–	21.80	–	38.20	–	**78.40**	–	38.70	–	46.70	–
Prompting/In-context learning approaches														
InstructTODS$_{GPT-4}$	-	Zero-shot	39.53	78.99	31.23	84.07	55.86	88.23	63.24	82.71	59.83	89.72	48.16	–
FnCTOD$_{LLaMA-2}$	70B	Few-shot	62.24	84.99	*46.83*	85.39	60.27	88.69	67.48	80.39	60.90	89.88	59.54	37.67
FnCTOD$_{GPT-4}$	-	Zero-shot	58.77	81.84	45.15	85.07	63.18	**91.06**	*76.39*	87.73	**69.48**	*90.16*	62.59	38.71
Cross-dataset Transfer approaches														
FnCTOD$_{LLaMA-2}$	13	SFT	49.76	76.80	29.50	67.60	48.87	81.33	64.66	68.97	53.59	85.09	49.28	25.68
Qwen-2.5	3B		56.51	83.48	38.83	82.98	68.77	84.02	51.65	88.52	51.0	87.15	53.35	31.02
LLaMA-3.1	8B		62.89	86.53	45.77	*85.85*	72.2	86.89	60.35	91.98	58.15	*90.16*	59.87	37.67
Qwen-2.5	32B		*66.07*	*87.02*	45.54	84.55	76.08	86.46	62.62	92.05	61.62	89.96	62.39	39.9
Our RL-based Cross-dataset Transfer approach														
Qwen-2.5	3B	GRPO	54.39	82.04	39.13	82.4	71.44	85.37	53.67	88.15	53.81	88.02	54.49	31.56
LLaMA-3.1	8B		63.87	85.31	46.19	85.61	*77.14*	88.41	71.97	*93.54*	60.63	89.96	*63.96*	*41.89*
Qwen-2.5	32B		**66.59**	**87.79**	**50.1**	**87.83**	**80.69**	*90.01*	74.79	**94.72**	*67.75*	**92.91**	**67.98**	**46.53**

Table 4: Per-domain Joint Goal Accuracy (JGA) and F1 scores of various models and training strategies on MultiWOZ 2.1 benchmark. Few-shot prompting was performed with 5 examples. Baseline metrics are taken directly from the corresponding publications. Metrics of our runs are averaged over 5 runs.

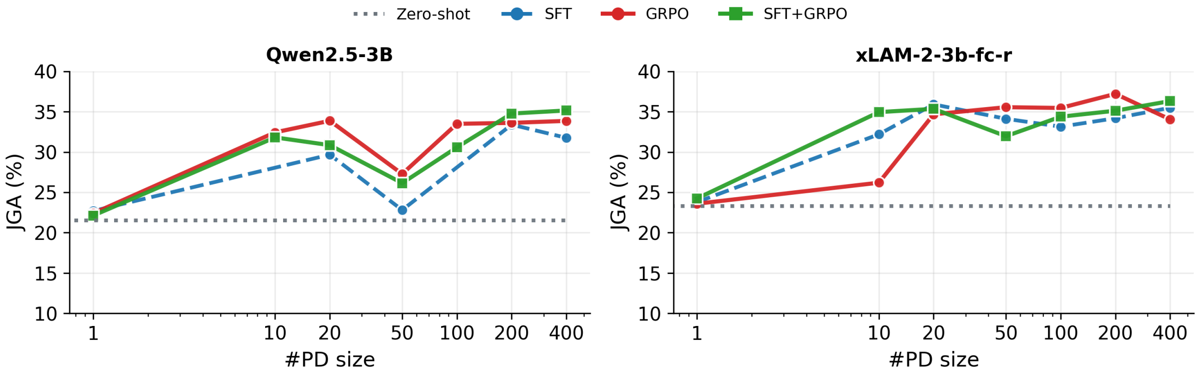


Figure 4: Performance (JGA, %) on MultiWOZ 2.4 of various training methods as a function of the number of per-domain (#PD size) dialogues seen during finetuning. The subplots contrast the base instruct model (Qwen-2.5-3B) and its function-calling pretrained counterpart (xLAM-2-3b-fc-r). Zero-shot results are shown as horizontal dotted lines.

data efficiency, providing substantial performance gains even with minimal training data and clearly surpassing all SFT baselines for 8B models across all examined data sizes.

4.7.5 Comparison of Reward Strategies

Figure 5 presents the comparison between two reward strategies within GRPO training: *full-match* — 1.0 if all slots are predicted correctly, otherwise 0; and *partial-match* — the fraction of right predicted slots. The full-match reward consistently achieves higher performance across all models and per-domain data sizes compared to the partial-match strategy, which rewards near-correct function calls. The superiority of the full-match reward likely stems from its stronger alignment with the exact-match evaluation metric (Joint Goal Accuracy), encouraging models to produce precise and structurally correct function calls rather than approximately correct ones. In contrast, the partial-match strategy might introduce ambiguity by allowing the model to partially fulfill the task, ultimately diluting the precision of the reward signal. These findings underscore the importance of carefully selecting the reward function based on the exact nature of the downstream evaluation criteria, highlighting that a more stringent reward scheme can effectively guide the model towards superior overall performance.

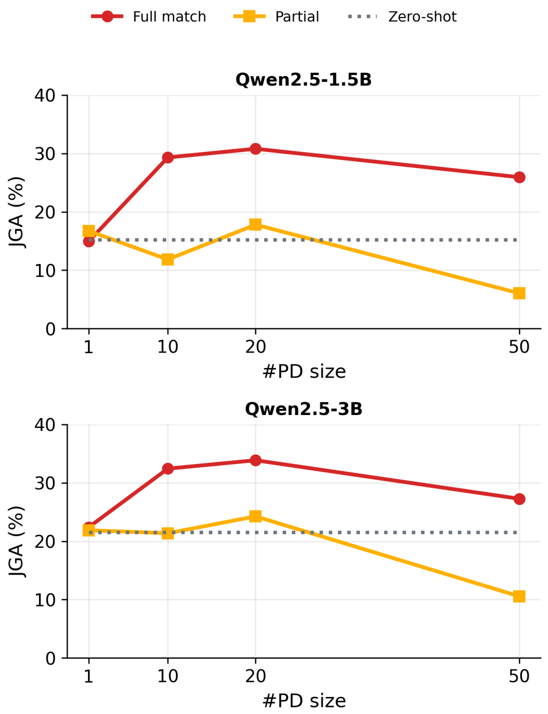


Figure 5: Performance (JGA, %) on MultiWOZ 2.4 of various reward strategies. Zero-shot results are shown as horizontal dotted lines.

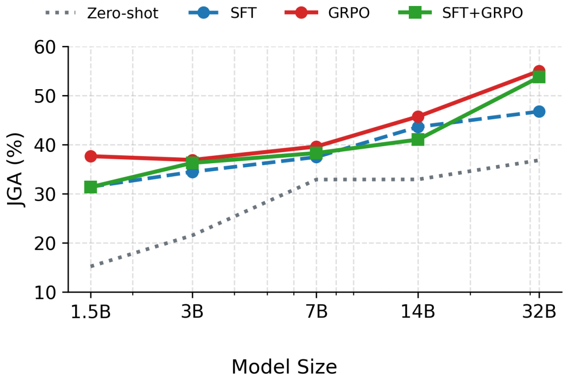


Figure 6: Performance (JGA, %) on MultiWOZ 2.4 of various Qwen-2.5 model scales from 1.5B up to 32B with different traning regimes.

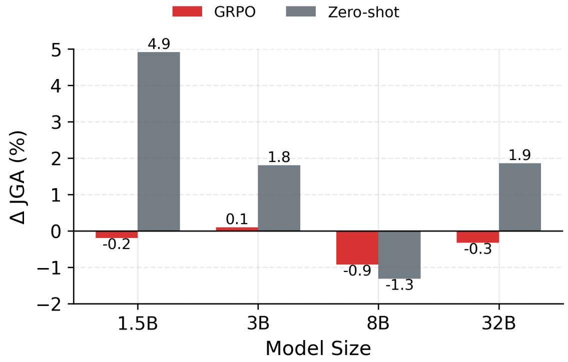


Figure 7: Delta performance (JGA, %) among tool-calling specialized models and their initial instruct models on MultiWOZ 2.4 of various model sizes. xLAM and Qwen cover the 1.5B, 3B, and 32B scales, whereas ToolACE and LLaMA represent the 8B scale.

5 Conclusion

In this work, we demonstrated that reinforcement learning (RL), specifically Group Relative Policy Optimization (GRPO), significantly enhances the generalization and data efficiency of large language models (LLMs) for Dialogue State Tracking (DST). Our results show that RL-based fine-tuning surpasses supervised fine-tuning (SFT) across a range of model scales (from 1.5B up to 32B), families (LLaMA and Qwen) and training data regimes, achieving state-of-the-art performance on the MultiWOZ 2.1 and 2.4 benchmarks in cross-dataset setting. Notably, our approach with an 8B model outperformed even much larger models, including zero-shot prompted GPT-4 and 13B SFT baselines.

Further, our study highlighted the remarkable data efficiency of GRPO, where few dialogues was sufficient for substantial performance improvements, significantly lowering the barrier for adapting models for cross-dataset inference. Additionally, we illustrated the benefits of leveraging small models pretrained on structured tool-use tasks, achieving superior results compared to vanilla instruction-tuned counterparts.

Overall, our findings underscore the potential of reinforcement learning methods such as GRPO in advancing robust, efficient, and scalable DST solutions, setting a promising direction for future research in task-oriented dialogue systems.

Limitations

Despite the consistent gains reported in this work, our study has several limitations.

First, the pipeline depends on non-trivial *schema harmonization*, across heterogeneous training corpora and MultiWOZ. While we describe canonicalization and a function-call-to-belief-state mapping procedure (§4.3), small implementation choices (e.g., normalization, special values, naming conventions) can materially affect JGA. Moreover, our evaluation parses only the first `<tool_call>` span per turn, potentially missing complex settings involving multiple calls or compositional schemas.

Second, explicit penalties for tool-presence mismatches may bias the policy toward over-calling.

We do not report calibration diagnostics like false-positive rates, nor do we evaluate online interaction quality where the decision to call is context-dependent.

Finally, our reliance on parameter-efficient adaptation (LoRA) and single-slice sampling improves efficiency but may differ from full fine-tuning performance. It also remains an open question whether multi-slice or curriculum-style sampling would further improve robustness.

References

Shelly Bensal, Umar Jamil, Christopher Bryant, Melisa Russak, Kiran Kamble, Dmytro Mozolevskyi, Muayad Ali, and Waseem AlShikh. 2025. Reflect, retry, reward: Self-improving llms via reinforcement learning. *arXiv preprint arXiv: 2505.24726*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4516–4525, Hong Kong, China. Association for Computational Linguistics.

Rafael Carranza and Mateo Alejandro Rojas. 2025. Interpretable and robust dialogue state tracking via natural language summarization with llms. *arXiv preprint arXiv:2503.08857*.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *Preprint*, arXiv:2501.17161.

Huifang Du, Shuqin Li, Minghao Wu, Xuejing Feng, Yuan-Fang Li, and Haofen Wang. 2024. Rewarding what matters: Step-by-step reinforcement learning for task-oriented dialogue. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8030–8046, Miami, Florida, USA. Association for Computational Linguistics.

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

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. 2025. Retool: Reinforcement learning for strategic tool use in llms. *Preprint*, arXiv:2504.11536.

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, EMNLP 2023, Singapore, December 6-10, 2023*, pages 739–755. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, and Arthur Hinsvark and. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Daniel Han, Michael Han, and Unsloth team. 2023. Unsloth.

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. *Preprint*, arXiv:2005.02877.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2022. A simple language model for task-oriented dialogue. *Preprint*, arXiv:2005.00796.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

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

Xiujun Li, Yu Wang, Siqi Sun, Sarah Panda, Jingjing Liu, and Jianfeng Gao. 2018. Microsoft dialogue challenge: Building end-to-end task-completion dialogue systems. *arXiv preprint arXiv:1807.11125*.

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. *Preprint*, arXiv:2402.10466.

Zhaojiang Lin, Bing Liu, Andrea Madotto, Seungwhan Moon, Zhenpeng Zhou, Paul Crook, Zhiguang Wang, Zhou Yu, Eunjoon Cho, Rajen Subba, and Pascale Fung. 2021a. Zero-shot dialogue state tracking via cross-task transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7890–7900, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Weiwen Liu, Xu Huang, Xingshan Zeng, xinlong hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong WANG, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Wang Xinzhi, Yong Liu, Yasheng Wang, and 8 others. 2025. ToolACE: Winning the points of LLM function calling. In *The Thirteenth International Conference on Learning Representations*.

Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiying Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. 2019. An end-to-end dialogue state tracking system with machine reading comprehension and wide & deep classification. *arXiv preprint arXiv:1912.09297*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2023. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Kam-Fai Wong. 2018. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. *Annual Meeting of the Association for Computational Linguistics*.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. Toolrl: Reward is all tool learning needs. *arXiv preprint arXiv: 2504.13958*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv: 1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv: 2402.03300*.

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.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Xingguang Wang, Xuxin Cheng, Juntong Song, Tong Zhang, and Cheng Niu. 2024. Enhancing dialogue state tracking models through LLM-backed user-agents simulation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8724–8741, Bangkok, Thailand. Association for Computational Linguistics.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *Preprint*, arXiv:2406.12045.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022a. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 351–360, Edinburgh, UK. Association for Computational Linguistics.

Fanghua Ye, Xi Wang, Jie Huang, Shenghui Li, Samuel Stern, and Emine Yilmaz. 2022b. MetaASSIST: Robust dialogue state tracking with meta learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1157–1169, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuanqing Yu, Zhefan Wang, Weizhi Ma, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. 2025. Steptool: Enhancing multi-step tool usage in llms through step-grained reinforcement learning. *Preprint*, arXiv:2410.07745.

Yirong Zeng, Xiao Ding, Yuxian Wang, Weiwen Liu, Wu Ning, Yutai Hou, Xu Huang, Bing Qin, and Ting Liu. 2025. itool: Reinforced fine-tuning with dynamic deficiency calibration for advanced tool use. *arXiv preprint arXiv: 2501.09766*.

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

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Akshara Prabhakar, Haolin Chen, and 1 others. 2024. xlam: A family of large action models to empower ai agent systems. *arXiv preprint arXiv:2409.03215*.

Sai Zhang, Yuwei Hu, Xiaojie Wang, and Caixia Yuan. 2023. An asynchronous updating reinforcement learning framework for task-oriented dialog system. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. 2025. Nemotron-research-tool-n1: Exploring tool-using language models with reinforced reasoning. *arXiv preprint arXiv: 2505.00024*.

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*.

A Appendix

Implementation Details Details about the fine-tuning hyperparameters for SFT and GRPO phase can be found in Table 5 and in Table 6 accordingly. The fine-tuning was conducted on a single H100 80GB GPU.

Parameter	**Value**
Precision mode	bfloat16
LoRA target modules	$Q_{proj}, K_{proj}, V_{proj}$
LoRA rank	16
LoRA alpha	16
LoRA dropout	0.05
Epochs	1
Batch size	8
Gradient accumulation steps	4
Learning rate	0.0002
Optimizer	AdamW
Weight decay	0
Learning rate scheduler	cosine
Warmup steps	0
Cutoff length	4096

Table 5: SFT hyperparameters and training configuration

Parameter	**Value**
Precision mode	bfloat16
LoRA target modules	$Q_{proj}, K_{proj}, V_{proj}$
LoRA rank	16
LoRA alpha	16
LoRA dropout	0.05
Epochs	1
Batch size	8
Gradient accumulation steps	1
Learning rate	0.0003
Optimizer	AdamW
Weight decay	0
Learning rate scheduler	cosine
Warmup steps	0
Cutoff length	4096
Temperature	1.0
Top p	1.0
Top k	−1
Number of generations	8

Table 6: GRPO hyperparameters and training configuration