Building Task-Oriented Dialogue Systems via Instruction Guidance without Annotated Data

Henry Gao and Jinho D. Choi
College of Arts and Sciences, Emory University
henry.gao2@emory.edu,jinho.choi@emory.edu

Abstract

Task-oriented dialogue (TOD) systems conventionally rely on supervised fine-tuning over large datasets, an approach that is both resource-intensive and difficult to generalize across domains. We investigate whether large language models (LLMs) can serve as effective TOD agents without any fine-tuning, relying solely on in-context prompting and unstructured conversational logs. To this end, we propose a two-stage framework in which an LLM first induces structured procedural instructions from raw multi-turn dialogues, then leverages these instructions to generate goal-oriented interactions. An iterative refinement loop further improves instruction quality by evaluating intermediate dialogue outputs and propagating feedback to update the instructions.

To address limitations inherent in existing evaluation protocols, we introduce an interactive evaluation framework centered on a constrained user simulator with access to ground-truth task goals. This design enables flexible assessment of task success beyond fixed dialogue trajectories, more faithfully reflecting the conditions of real-world deployment. Experiments demonstrate that the proposed approach produces coherent and task-effective dialogues without any annotated data. In our evaluation framework and metrics, we use **Gemma-3-27b-it** as the backbone model, achieving a a dialogue state F1 of 86.3%, which outperforms GALAXY (He et al., 2021) (84.3%) and MARS (Sun et al., 2023) (84.6%).

1 Introduction

Pretrained language models have achieved strong performance in both Natural Language Understanding (NLU) and Natural Language Generation (NLG) (Tian et al., 2024), enabling their widespread use in open-domain and task-oriented dialogue (TOD) systems. Unlike open-domain dialogue, TOD requires strict adherence to task constraints, structured decision-making, and accurate execution of domain-specific actions.

A common paradigm for TOD is to fine-tune pretrained models on task-specific data. While effective, this approach is costly in computation, and scales poorly across domains requiring multiple specialized models. This limitation is particularly evident in settings such as academic advising, where conversational patterns vary across institutions, making fine-tuning inefficient and impractical.

In this work, we challenge the necessity of fine-tuning for TOD systems. We investigate whether LLMs can be guided solely through prompting and existing human conversations to achieve competitive performance. We hypothesize that LLMs can leverage automatically extracted textual instructions from dialogue data to mimic human behavior and generate coherent task-oriented interactions.

To this end, we ask:

> *Can LLMs construct task-oriented dialogue systems by dynamically aggregating procedural instructions from multi-turn human conversations?*

We propose a two-stage framework (Figure 1). First, an LLM processes raw dialogue histories to induce structured natural-language instructions capturing user behavior and task procedures. Second, these instructions guide dialogue generation to replicate observed interaction patterns. This approach operates directly on unannotated data and produces interpretable instructions that can be inspected and modified by humans.

To improve instruction quality, we incorporate an iterative refinement process in which the model evaluates generated dialogues and updates instructions accordingly, mitigating errors such as missing or misordered actions.

[0]The code and data can be found on GitHub: https://github.com/emorynlp/TODEval

Proceedings of the 27th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pages 773–785
August 2–5, 2026.

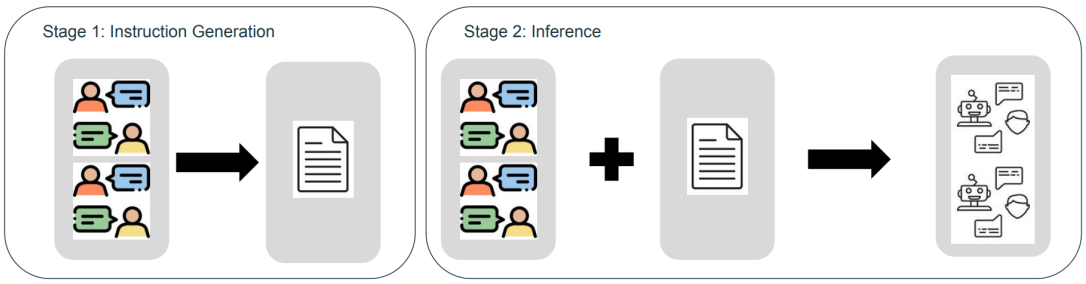

Figure 1: Two-stage framework in which texutal instructions are generated, and then used to produce a similiar dialogue.

We further address limitations in existing TOD evaluation metrics, such as inform and success rates (Budzianowski et al., 2018), which assume fixed action sequences. We propose an interactive simulation framework (Figure 2) using a user simulator with known goals, enabling evaluation based on task completion rather than strict trajectory matching. Building on prior metrics, we introduce additional measures (e.g., Inform Precision/Recall and Success Precision/Recall) to better assess action selection and response correctness.

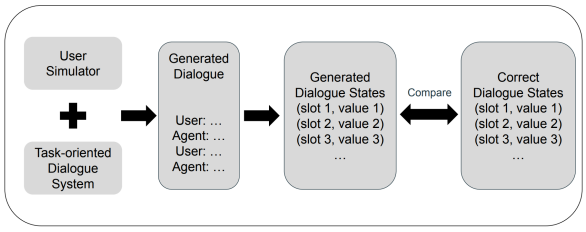

Figure 2: Evaluation Framework. The user simulator and the agent generates a dialogue, whose dialogue state is extracted and compared to the correct dialogue state for evaluation purpose.

Overall, we demonstrate that LLMs can achieve competitive TOD performance without fine-tuning by leveraging induced textual instructions, offering a scalable, interpretable, and resource-efficient alternative for dialogue system construction.

2 Related Works

2.1 Task-Oriented Dialogue Systems

2.1.1 LLM-based TOD

Recent work leverages large language models (LLMs) to unify dialogue state tracking, policy learning, and response generation via free-form text (Hu et al., 2024; Yang et al., 2024; Zhang et al., 2025). Methods such as AutoTOD (Xu et al., 2024) achieve strong performance by implicitly modeling dialogue states, while continual learning frameworks address domain adaptation (Zeng et al., 2025). However, relying on implicit reasoning often leads to inconsistencies and weak adherence to task constraints, limiting reliability in structured settings.

2.1.2 Instruction-Guided TOD

To improve controllability, prior work introduces structured guidance through predefined schemas or decision processes (Wen et al., 2025), or uses instructions to guide generation (Xu et al., 2024). While effective, many approaches require manually predefined schema. In contrast, we induce task instructions directly from raw dialogues, providing a lightweight and interpretable alternative that constrains generation without additional supervision.

2.2 Evaluation Metrics

2.2.1 Modular Systems

Modular TOD systems are evaluated using intermediate signals such as dialogue states or intents, with metrics like Joint Goal Accuracy and Slot-F1 enabling fine-grained assessment (Dey et al., 2025; Rim et al., 2025). These metrics are not applicable to end-to-end LLM systems that lack explicit intermediate outputs.

2.2.2 End-to-End Systems

Most LLM-based dialogue systems operate under an end-to-end paradigm, directly generating system responses without explicit intermediate representations. Evaluation in this setting has traditionally relied on the Inform and Success metrics introduced alongside the MULTIWOZ benchmark (Budzianowski et al., 2018). The Inform metric measures whether the system provides entities satisfying user constraints, while the Success metric evaluates whether all user requests are fulfilled.

However, this evaluation paradigm is subject to several substantive limitations. First, it assumes that the system follows a fixed trajectory aligned with the ground-truth dialogue, when in practice multiple valid action sequences may lead to successful task completion. For instance, when booking a hotel, an agent may elicit the user's name before or after confirming reservation details without affecting the final outcome. Conventional metrics do not account for such flexibility, and may penalize valid but non-canonical dialogue strategies.

Second, existing metrics evaluate each turn against ground-truth dialogue context, which precludes meaningful assessment of error recovery. When a system makes an incorrect decision early in the dialogue, subsequent turns are nonetheless generated conditioned on the correct past utterances rather than the system's own (potentially erroneous) history. This leads to a systematic overestimation of system performance that does not reflect the cascading consequences of errors encountered in real-world deployment.

These limitations were less pronounced in earlier dialogue systems, where the primary challenge resided in accurate information extraction and database retrieval. As pretrained models become increasingly capable of handling such subtasks, however, the evaluation focus necessarily shifts toward decision-making and action selection. In this context, metrics anchored to fixed trajectories and ground-truth states are insufficient for characterizing the behavior of dynamic, generation-based dialogue systems.

AutoTOD (Xu et al., 2024) proposes an evaluation framework grounded in user simulation; however, its simulator takes an active role in volunteering information and steering the interaction, such that the evaluation primarily measures the system's ability to collect and present information rather than its capacity for autonomous decision-making. Consequently, dialogue flow is largely preserved by the simulator, and the agent is not sufficiently challenged to select appropriate dialogue acts or manage conversational flow under conditions of uncertainty—a limitation that motivates the more constrained simulation paradigm adopted in the present work.

To address these limitations, we adopt an interactive evaluation framework in which the agent interacts with a constrained user simulator that withholds information unless explicitly solicited. This design compels the agent to actively select appropriate dialogue acts and manage conversational flow, rather than passively responding to volunteered cues. Evaluation is consequently grounded in multi-turn task completion, yielding a more realistic and discriminative assessment of decision-making capability in LLM-based dialogue systems.

3 Approach

3.1 Task Definition

A dialogue $D = \{u_1, \ldots, u_N\}$ alternates between user (odd) and system (even) utterances, with an associated dialogue state $S = \{(s_i, v_i)\}$. The objective is to generate a dialogue whose final state exactly matches a target S.

3.2 Overview & Architecture

We construct an evolving instruction graph $G = (V, E)$, where nodes are natural-language instructions and edges encode plausible transitions. This representation captures multiple valid task flows while preserving procedural structure. Instructions are induced incrementally from raw training dialogues without any manual annotation, and are subsequently used to guide dialogue generation at inference time. We investigate two variants of the instruction generation procedure, which differ in whether intermediate model outputs are incorporated during the refinement process.

3.3 Instruction Generation

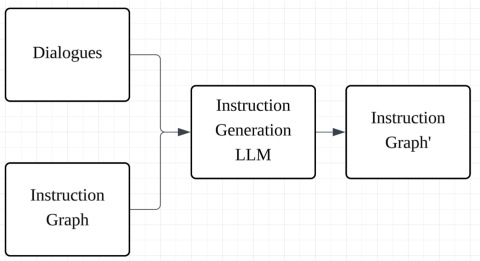

Figure 3: Baseline instruction generation: sequential construction and update of the instruction graph from training dialogues.

As shown in Figure 3, the baseline approach initializes with an empty instruction graph at the start of training. For each training dialogue, the current instruction graph and the dialogue context are passed to GEMMA3, which produces a revised instruction graph incorporating patterns observed in the new dialogue. This procedure is formalized in Algorithm 1.

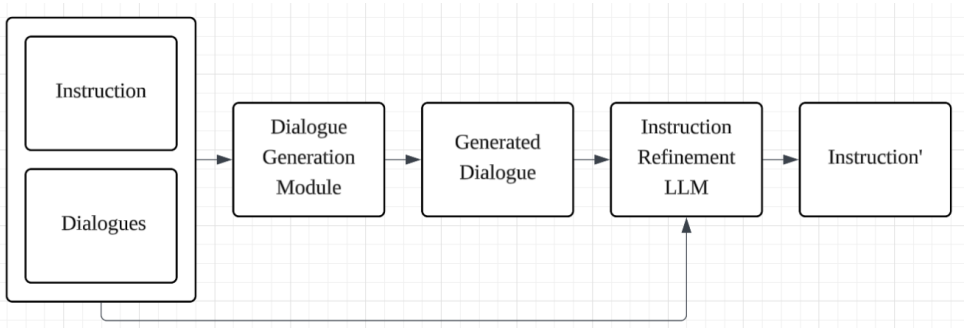

Figure 4: Proposed approach: self-refinement of the instruction graph using intermediate dialogue outputs generated by the model itself.

Algorithm 1 Instruction Generation (Baseline)

Require: Training dialogues $D = \{d_1, d_2, \dots\}$
Ensure: Final instruction graph I
$I \leftarrow \emptyset$
for each dialogue $d \in D$ **do**
  Feed (I, d) into GEMMA3
  $I \leftarrow$ updated instruction graph
end for
return I

While this approach yields a coherent instruction graph, it relies solely on observed training dialogues and provides no mechanism for detecting or correcting procedural inconsistencies in the induced instructions.

3.4 Instruction Generation with Intermediate Output

To mitigate this limitation, we propose a self-refinement loop (Figure 4), wherein the instruction graph is iteratively evaluated against model-generated dialogue and subsequently updated. Specifically, for each training dialogue d, the agent interacts with a constrained user simulator conditioned on the ground-truth dialogue state S extracted from d, producing an intermediate dialogue d'. Both the original dialogue d and the intermediate output d' are then provided to GEMMA3 alongside the current instruction graph, enabling error-driven refinement that corrects procedural inconsistencies surfaced during simulation. This procedure is formalized in Algorithm 2.

Compared to the approach without intermediate output, this variant incorporates feedback from the model's own behavior, enabling the instruction graph to be refined not only toward patterns present in training data but also away from failure modes observed during simulation. The key distinction

Algorithm 2 Instruction Generation with Intermediate Output

Require: Training dialogues $D = \{d_1, d_2, \dots\}$
Ensure: Final instruction graph I
$I \leftarrow \emptyset$
for each dialogue $d \in D$ **do**
  Extract dialogue state S from d
  Initialise user simulator with S
  $d' \leftarrow$ dialogue between agent and user simulator
  Feed (I, d, d') into GEMMA3
  $I \leftarrow$ updated instruction graph
end for
return I

is that the intermediate dialogue d' serves as a diagnostic signal: discrepancies between d' and d reveal gaps in the current instructions, which are then corrected in the subsequent update step.

3.5 Inference

At test time, the agent generates responses conditioned on the dialogue history and the final induced instruction graph I, without access to any ground-truth annotations.

3.6 Justification of Instruction Graph Architecture

We adopt a directed graph structure, which effectively captures dependency patterns (*e.g.*, if instruction A is executed, instruction B is likely to follow). This representation enables efficient instruction retrieval with time complexity $O(|I|)$, while allowing near constant-time access to subsequent candidate instructions.

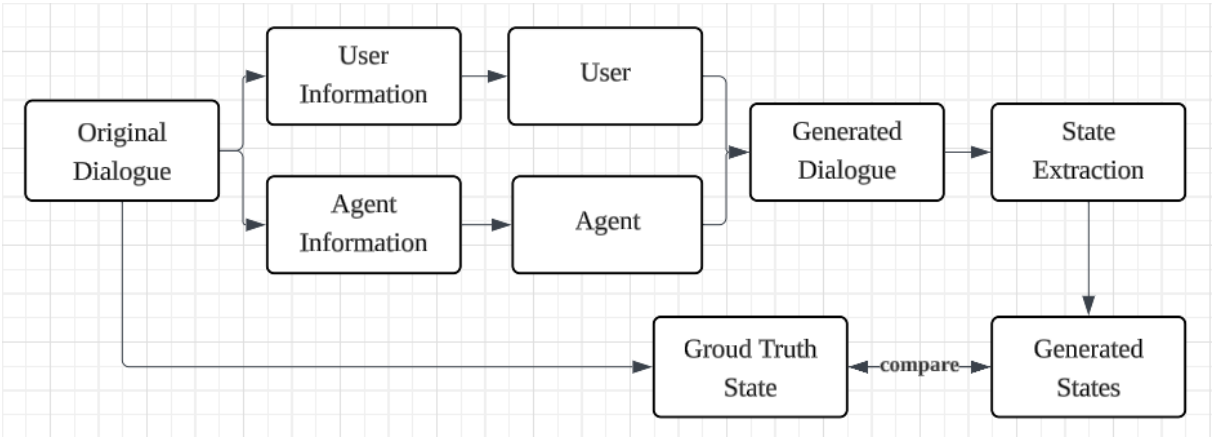

Figure 5: The evaluation framework leverages agent and user simulator to generate a new dialogue based on the original dialogue, and then compares the states of the original dialogue and the generated dialogue.

3.7 Generalizability

Although the proposed framework is evaluated on MULTIWOZ, it is inherently domain-agnostic and can be readily extended to a broad range of conversational settings beyond task-oriented dialogue. The nature of the induced instructions adapts to the target domain: whereas instructions in task-oriented settings typically govern information-seeking and information-providing acts, instructions in other domains—such as mental health support or counselling dialogues—may instead capture stylistic and affective dimensions of communication, such as empathetic framing or conversational register.

A further strength of the framework is its accessibility. As the approach requires no fine-tuning of underlying language models, it can be deployed by practitioners without deep technical expertise, lowering the barrier to constructing customized dialogue agents across diverse application domains. This property directly addresses a core motivation of the present work: enabling the rapid development of domain-specific agents in settings where fine-tuning is prohibitively costly, data is scarce, or annotation resources are unavailable.

3.8 Innovative Elements

Key contributions of the approaches include:

- Interpretable and controllable instruction-graph structure modeling dependencies between instructions.
- Integration of the instruction graph into generation for guided responses.
- Self-refinement loop where the LLM updates the graph based on output errors, analogous to gradient descent.

4 Evaluation Platform

4.1 Motivation

Traditional metrics for task-oriented dialogue (TOD) emphasize response quality and information retrieval, which are insufficient for modern LLM-based systems. We instead focus on decision-making: whether a system selects appropriate actions and actively elicits missing information to complete tasks.

4.2 Platform Design

We propose an interactive evaluation framework where the system engages with a user simulator. The simulator has access to the ground-truth user state but reveals information only when explicitly queried, enabling flexible yet controlled interactions and preventing reliance on fixed dialogue trajectories.

4.2.1 Dialogue Generation

Dialogue is generated through multi-turn interaction between the system and simulator (Figure 5). The simulator initializes the conversation using the original dialogue and responds based on the underlying user state, while the system generates actions conditioned on its policy. The interaction terminates upon task completion or after a maximum number of turns.

4.2.2 Dialogue Evaluation

Evaluation is conducted at the dialogue state level rather than over surface text. Predicted dialogue

states, represented as slot–value pairs, are extracted from generated dialogues and compared against ground-truth annotations. This formulation assesses the system's capacity to gather task-relevant information and execute appropriate dialogue acts, abstracting away from surface-level lexical variation. Although dialogue generation can be stochastic, the scoring procedure is fully deterministic given a fixed dialogue, ensuring reproducibility of evaluation outcomes.

4.3 Modularity

The framework adopts a modular design in which the agent, user simulator, and dialogue state extractor are implemented as independent components with well-defined interfaces. Inputs are exchanged via structured JSON objects—including fields such as `utterances`, `user_information`, and `agent_information`—providing a consistent contract between components. This design facilitates rapid experimentation and lowers the barrier to extending the framework with alternative models or evaluation protocols.

4.3.1 Metrics

We evaluate system performance using seven metrics across three categories: **Inform** (Precision, Recall, and F1), **Inform Correctness** (Precision and Recall), and **Success** (Precision and Recall). Together, these metrics assess the overlap between predicted and ground-truth dialogue states, capturing both the quality of action selection—specifically, the system's ability to elicit relevant information—and overall task completion. Crucially, evaluation emphasizes whether the system produces correct dialogue acts at each turn, rather than the factual accuracy of the generated utterances. Whereas Inform Precision, Recall, and F1 measure slot-level accuracy, Inform Correctness Precision and Recall evaluate value-level accuracy.

5 Experiments

5.1 Dataset and Preprocessing

We use the **MultiWOZ** dataset (Budzianowski et al., 2018), a standard benchmark for task-oriented dialogue. Dialogues are stored as JSON objects indexed by unique IDs. We adopt the official train/test split: 8,437 training dialogues (113,552 utterances) for instruction induction and 1,000 test dialogues (14,744 utterances) for evaluation.

5.2 Experimental Setup

Both approaches are trained on the training split for one epoch and evaluated on the official test split under identical dialogue ordering.

Choice of LLM We use **Gemma-3-27b-it** for instruction generation and dialogue simulation, and additionally evaluate **Qwen3-30b-a3b** for dialogue simulation to assess the framework's generalizability across backbone models. Both models were chosen for their favorable trade-off between capability, inference latency, and cost. **Gemma-3-27b-it** was accessed via the Google API, whereas **Qwen3-30b-a3b** was deployed locally on GPU. Following the release of **Gemma-4-26b-a4b-it** in early April, we further evaluated our framework by replacing both the dialogue generator and user simulator with this model through the API.

Choice of user simulator In our framework, **Gemma-3-27b-it** is configured to act as a user simulator, similiar to Xu et al. (2024). The simulated user is designed to systematically follow pre-defined goals, gradually expressing intents in a natural and coherent manner to ensure that the dialogue progresses toward task completion. The simulated user only provides information explicitly requested by the assistant and will respond with "I don't know" if asked for details outside the provided user information. The dialogue ends either when all task goals are achieved or when no further progress toward the goals is possible. By structuring the simulation this way, the LLM can generate realistic user behaviors, producing multi-turn interactions that accurately reflect how a human might interact with an AI assistant to complete a task efficiently. The prompt for user simulator can be found in Appendix 8.1.3.

Choice of state extraction method We adopt the state extraction method introduced by James Finch, as it demonstrates high accuracy in extracting slot–value pairs (Finch et al., 2025). This makes it well-suited for integration into our evaluation framework, ensuring reliable performance measurement.

5.3 Baselines

As our evaluation framework differs from standard benchmarks, results are not directly comparable to prior work. We therefore include strong baselines by collecting outputs from two of the best-performing task-oriented dialogue (TOD) systems

Model	IR	ICR	IP	ICP	Inform F1	SR	SP
GALAXY	0.806	0.698	0.884	0.754	0.843	0.367	0.356
MARS	0.802	0.697	**0.895**	**0.779**	0.846	0.365	0.355
Proposed (Qwen3 w/o intermediate)	0.839	0.761	0.711	0.644	0.770	**0.722**	**0.677**
Proposed (Qwen3 w/ intermediate)	0.722	0.659	0.803	0.732	0.760	0.482	0.464
Proposed (Gemma3 w/o intermediate)	**0.921**	**0.851**	0.811	0.746	**0.863**	0.699	0.651
Proposed (Gemma3 w/ intermediate)	0.900	0.832	0.811	0.749	0.853	0.633	0.589
Proposed (Gemma4 w/o intermediate)	0.842	0.767	0.773	0.701	0.807	0.764	0.719
Proposed (Gemma4 w/ intermediate)	0.850	0.769	0.770	0.695	0.808	0.757	0.707

Table 1: Evaluation results for different dialogue generation approaches. IR = Inform Recall, ICR = Inform Correctness Recall, IP = Inform Precision, ICP = Inform Correctness Precision , SR = Success Recall, SP = Success Precision. The best score in each metric is bolded.

on the MULTIWOZ benchmark: GALAXY TOD (He et al., 2021) and MARS TOD (Sun et al., 2023). Their responses are paired with the user simulator to reconstruct full dialogues for evaluation under our framework.

Because these systems generate utterances that closely resemble the original dialogue turns, the user simulator can be configured to reproduce the original user utterances, resulting in complete dialogue interactions.

5.4 Quantitative Results

As shown in Table 1, the proposed approaches outperform the baselines on most metrics, except for Inform Precision and Inform Correctness Precision. Notably, the variant without intermediate output achieves the highest F1 score. The lower precision suggests that, while the proposed methods effectively cover required actions, they may over-request information.

Interestingly, the **Gemma-3-27b-it** variant without intermediate refinement slightly outperforms the one with intermediate refinement. A possible explanation is that iterative refinement produces longer and more complex instructions, which may reduce the model's ability to effectively utilize the provided context. Consequently, the additional guidance does not translate into improved performance.

In contrast, **Gemma-4-26b-a4b-it** benefits from intermediate refinement, achieving better performance when intermediate outputs are used. This suggests that **Gemma-4-26b-a4b-it** may be better able to interpret and follow more structured instructions, allowing it to more effectively leverage the additional guidance.

Another notable observation is that **Gemma-4-26b-a4b-it** consistently underperforms **Gemma-3-27b-it**, even though both use instructions generated by **Gemma-3-27b-it**. One possible explanation is that Gemma4 may rely more heavily on its internal reasoning than on the provided instructions, resulting in lower overall performance.

5.5 Comparison with Existing Metrics

Because our framework emphasizes decision-making rather than surface-level response quality, traditional metrics are not directly applicable. For reference, we report Inform and Success scores for baseline models in Table 3.

MARS consistently outperforms GALAXY under both traditional and proposed metrics. This is expected, as both systems rely on ground-truth user utterances and are not required to actively decide which information to request. Consequently, performance differences mainly reflect response generation quality, where MARS shows a modest advantage.

6 Analysis

6.1 LLM Capability Analysis

To assess the inherent decision-making capability of pretrained language models, we evaluate model performance in an instruction-free setting. Due to the unavailability of **Gemma-3-27b-it** via API at the time of writing, we are only able to conduct instruction-based experiments with **Qwen3-30b-a3b** and **Gemma-4-26b-a4b-it**.

As shown in Table 2, the models achieve high recall but relatively low precision, suggesting broad coverage of plausible dialogue acts but limited selectivity. This indicates that while pretrained LLMs can surface common dialogue patterns learned during pretraining, they lack the task-specific control required to reliably execute domain policies without explicit instructions. The relatively strong Success scores further suggest that the models retain the ability to respond appropriately to direct user requests even in the absence of structured guidance;

Model	IR	ICR	IP	ICP	Inform F1	SR	SP
Proposed (Qwen3 w/o intermediate)	0.839	0.761	0.711	0.644	0.770	0.722	0.677
Proposed (Qwen3 w/ intermediate)	0.722	0.659	0.803	0.732	0.760	0.482	0.464
Empty instruction (Qwen3)	0.888	0.735	0.689	0.571	0.776	0.838	0.783
Proposed (Gemma4 w/o intermediate)	0.842	0.767	0.773	0.701	0.807	0.764	0.719
Proposed (Gemma4 w/ intermediate)	0.850	0.769	0.770	0.695	0.808	0.757	0.707
Empty instruction (Gemma4)	0.820	0.716	0.780	0.679	0.800	0.739	0.690

Table 2: Evaluation results for different dialogue generation approaches. IR = Inform Recall, ICR = Inform Correctness Recall, IP = Inform Precision, ICP = Inform Correctness Precision , SR = Success Recall, SP = Success Precision. The best score in each metric is bolded.

Model	Inform	Success
GALAXY	85.4	75.7
MARS	88.9	78.0

Table 3: Baseline performance on standard Inform and Success metrics.

however, this is not sufficient for consistent task completion.

An interesting observation is that **Qwen3-30b-a3b** achieves higher F1 in the instruction-free setting compared to its instruction-conditioned counterparts, whereas **Gemma-4-26b-a4b-it** exhibits the opposite trend, with lower F1 in the instruction-free setting. This discrepancy suggests that increased reliance on internal reasoning may lead to deviations from expected decision boundaries, even in domains where the model has been extensively pretrained.

6.2 Instruction Length Analysis

Counterintuitively, the variant without intermediate refinement achieves slightly better performance than its refined counterpart on both **Gemma-3-27b-it** and **Qwen3-30b-a3b**. As illustrated in Figure 6, iterative refinement substantially increases instruction length, often approaching or exceeding the model's effective context capacity. Furthermore, Figure 7 shows that refined instructions consist of fewer but considerably longer sentences, suggesting greater syntactic and semantic complexity. These observations indicate that overly long and information-dense instructions may hinder model comprehension, thereby diminishing the benefits of more detailed guidance. One potential direction for addressing this limitation is to replace monolithic instructions with a graph-based traversal mechanism, where embedding-based retrieval dynamically identifies the most relevant nodes and candidate subsequent actions according to the current dialogue state. Such an approach could significantly reduce instruction length while preserving task-relevant information, mitigating the context limitations of current LLMs.

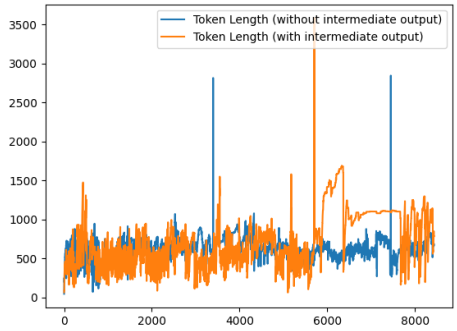

Figure 6: The token length of instructions. The x-axis is the number of dialogues processed and the y-axis is the token length of instructions.

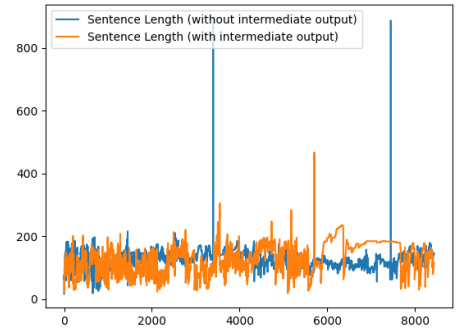

Figure 7: The sentence length of instructions. The x-axis is the number of dialogues processed and the y-axis is the sentence length of instructions.

6.3 Instruction Graph Following Capability

To assess whether LLMs can reliably follow complex instruction graphs, we analyze node traversal behavior during dialogue generation. Since LLM outputs may vary across runs, we conduct this experiment over ten independently generated dialogues. At each turn, the agent is explicitly

prompted to select the next node in the instruction graph, and the selected nodes are recorded across all dialogues. The most frequently visited nodes under our proposed approach correspond to core task-oriented dialogue acts: requesting information, confirming user intent, and retrieving results. The prevalence of these nodes aligns with expected behavior in TOD settings, suggesting that the LLM is capable of navigating the instruction graph in a task-coherent manner without explicit supervision over node selection. Full node visitation frequencies are reported in Appendix 8.4. We further attempted to analyze node traversal behavior at scale; however, the unpredictable outputs of smaller models frequently caused program failures, precluding systematic analysis. We leave a comprehensive evaluation of node traversal across model scales to future work, where more capable models may yield stable and interpretable traversal patterns.

6.4 State Extraction Analysis

We validate the reliability of the state extraction module using ground-truth annotations from MULTIWOZ. As shown in Table 4, the method achieves high precision (0.9040) and reasonable recall (0.7547), indicating accurate extraction of slot–value pairs. While more advanced models could potentially be used in this module, these results support the validity of our evaluation setup.

Metric	Score
Precision	0.9040
Recall	0.7547

Table 4: Performance of the state extraction method on the MULTIWOZ dataset.

6.5 Dataset Analysis

Manual inspection reveals annotation inconsistencies in MULTIWOZ. For example, values such as "free internet" may be labeled as "yes," despite being semantically equivalent. Such discrepancies negatively impact strict matching metrics (e.g., Inform Correctness), leading to potential underestimation of performance. Example error cases can be found in Appendix 8.3.

6.6 Discussion

The results highlight the importance of controlling instruction length to maintain LLM effectiveness. Future work may explore more efficient prompt design or structured representations of instruction graphs, enabling selective retrieval of relevant instructions and improving both efficiency and performance. In particular, an algorithm could be designed to dynamically extract a subset of relevant nodes from the instruction graph conditioned on the current dialogue context, and provide only these candidate instructions to the LLM. This would reduce the cognitive load on the model by avoiding full-graph processing, while preserving sufficient contextual information for accurate decision-making.

Furthermore, since the proposed framework is applicable to a broad range of text-based interaction tasks, future work should evaluate its effectiveness across diverse domains. In particular, experiments on tasks that remain challenging for current LLMs would provide a more comprehensive assessment of the framework's advantages and its potential to improve performance beyond conventional approaches.

7 Conclusion

This work investigates task-oriented dialogue (TOD) without annotated data, proposing a framework that induces natural-language instructions from raw conversations and uses them to guide generation. By separating instruction induction from generation, the approach enables LLMs to produce coherent, goal-oriented dialogues while maintaining interpretability and controllability through explicit procedures.

Results show that instruction-guided prompting, combined with iterative refinement, effectively captures task-specific behavior and mitigates common errors. The framework reduces reliance on costly finetuning, making it suitable for evolving domains. In addition, we introduce an interactive evaluation platform that assesses decision-making through multi-turn task completion, providing a more realistic alternative to static metrics.

However, performance depends on the quality of induced instructions and the underlying LLM, and robustness across domains remains an open challenge.

Overall, this work demonstrates that instruction-driven methods can serve as a scalable and interpretable alternative to fine-tuning for TOD. We hope it motivates future research on improved instruction induction, efficient prompting, and more robust evaluation of LLM-based dialogue systems.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Suvodip Dey, Yi-Jyun Sun, Gokhan Tur, and Dilek Hakkani-Tür. 2025. Know your mistakes: Towards preventing overreliance on task-oriented conversational AI through accountability modeling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28830–28843, Vienna, Austria. Association for Computational Linguistics.

James D. Finch, Yasasvi Josyula, and Jinho D. Choi. 2025. Generative induction of dialogue task schemas with streaming refinement and simulated interactions. *Preprint*, arXiv:2504.18474.

Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, Jian Sun, and Yongbin Li. 2021. GALAXY: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. *CoRR*, abs/2111.14592.

Chi Hu, Yimin Hu, Hang Cao, Tong Xiao, and JingBo Zhu. 2024. Teaching language models to self-improve by learning from language feedback. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6090–6101, Bangkok, Thailand. Association for Computational Linguistics.

Daniel Rim, Minsoo Cho, Changwoo Chun, and Jaegul Choo. 2025. To chat or task: a multi-turn dialogue generation framework for task-oriented dialogue systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 576–592, Vienna, Austria. Association for Computational Linguistics.

Haipeng Sun, Junwei Bao, Youzheng Wu, and Xiaodong He. 2023. Mars: Modeling context and state representations with contrastive learning for end-to-end task-oriented dialog. *Preprint*, arXiv:2210.08917.

Yufei Tian, Tenghao Huang, Miri Liu, Derek Jiang, Alexander Spangher, Muhao Chen, Jonathan May, and Nanyun Peng. 2024. Are large language models capable of generating human-level narratives? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17659–17681, Miami, Florida, USA. Association for Computational Linguistics.

Xiangyu Wen, Jianyuan Zhong, Zhijian Xu, and Qiang Xu. 2025. Guideline compliance in task-oriented dialogue: The chained prior approach. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6750–6776, Albuquerque, New Mexico. Association for Computational Linguistics.

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 *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2748–2763, Bangkok, Thailand. Association for Computational Linguistics.

Yuan Yang, Siheng Xiong, Ali Payani, Ehsan Shareghi, and Faramarz Fekri. 2024. Can LLMs reason in the wild with programs? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9806–9829, Miami, Florida, USA. Association for Computational Linguistics.

Min Zeng, Haiqin Yang, Xi Chen, and Yike Guo. 2025. Task-wrapped continual learning in task-oriented dialogue systems. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3173–3183, Albuquerque, New Mexico. Association for Computational Linguistics.

Xiaoying Zhang, Baolin Peng, Ye Tian, Jingyan Zhou, Yipeng Zhang, Haitao Mi, and Helen M. Meng. 2025. Self-tuning: Instructing LLMs to effectively acquire new knowledge through self-teaching. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5688–5724, Vienna, Austria. Association for Computational Linguistics.

8 Appendix

8.1 Prompts

8.1.1 Instruction Generation Prompt without Intermediate Output

```
Task: refine a graph of instructions on a task-
    oriented dialog system.

    Guidelines:
    Extract only essential agent actions or
     decisions; merge related ones.
    Create a separate graph for each distinct
     user goal (e. g., restaurant, hotel). Label
     each as "Goal 1: ...".
    Add conditional checks (e. g., "Check if
     user gave location preference") for details
     already provided.
    Each node covers one information type only.
    No specific names, values, or phrases. Use
     general terms.
    Include only dialog-level actions (ask/
     provide/confirm), not backend processes.
    Keep concise but complete.
    Refine the current information. Do not
     remove any information from the current
     instruction.

    Output Format:

    Goal [N]: [Goal description]

    Nodes:
    - N0: Start
    - N1: [Instruction]
```

```
- N2: [Instruction]
...
- NX: End

Edges:
- N0 -> N1
- N1 -> N2
...
- NX-1 -> NX


Input Dialogue Format:

user utterance 1
agent response 1
user utterance 2
agent response 2

Current Instruction:
{instruction}

Dialogue:
{dialog}

Now refine the instructions based on the dialog above. Only output the refined instruction. The refined instruction should include goals mentioned in current instruction.
```

8.1.2 Instruction Generation Prompt with Intermediate Output

```
Refine a graph of instructions based on its output on a task-oriented dialog system.

Guidelines:
Extract only essential agent actions or decisions; merge related ones.
Create a separate graph for each distinct user goal (e.g., restaurant, hotel). Label each as "Goal 1: ...".
Add conditional checks (e.g., "Check if user gave location preference") for details already provided.
Each node covers one information type only.
No specific names, values, or phrases. Use general terms.
Include only dialog-level actions (ask/provide/confirm), not backend processes.
Keep concise but complete.
Refine the current information. Do not remove any information from the current instruction.

Output Example:

Goal 1: Restaurant

Nodes:
- N0: Start
- N1: Ask price range
- N2: Ask location preference
...
- NX: End

Edges:
- N0 -> N1
- N1 -> N2
...
- NX-1 -> NX


Input Dialogue Format:

user utterance 1
agent response 1
generated response 1
user utterance 2
agent response 2
generated response 2

Current Instruction:
{instruction}

Reference dialogue:
{dialog}

Generated dialogue:
{generated_dialog}

Now refine the instructions based on the dialog above. Only output the refined instruction. The refined instruction should include goals mentioned in current instruction.
```

8.1.3 User Simulator Prompt

```
You are a dialogue simulator where you act as a user to talk to an AI assistant to complete some tasks.

You should carefully read and understand the User Goals below, then talk with the AI Assistant and gradually express the intents in the goals. Your purpose is to let the user achieve the goals as much as possible. You should not act as the agent.

Note that the AI Assistant is not perfect. It may make various mistakes, including ignoring the user's requests, executing the wrong instructions, forgetting early conversation content, etc. The user you play should talk to the AI Assistant as patiently as possible, remind him to correct when you find that the AI assistant made a mistake, and complete the task as much as possible.

When asking some information of a venue (restaurant, hotel, attraction) or a train, the user should specify the name or train id he chooses.

When the dialogue goals are completed or are not been completed, the user will output "Dialogue Ends" to indicate the end of the dialogue. The user doesn't need to try conditions other than the dialogue goals.

The user has a clear goal in mind, so he does not need to ask the AI assistant that "Is there anything else I need to know?".

The user does not need to talk too much with the AI assistant. If the task goals are
```

```
    completed, please end the conversation as
    soon as possible.

The user will not provide any information that
    the agent does not ask for. If the agent
    asks for some information not in the
    information below, answer "I don't know".

Here are the information related to the user:
{user_information}

Current conversation:
{context}

Please generate one to three sentences as the
    next user utterance.
```

8.2 Data File Example

```
"SNG0073.json": {
        "utterances": [
            "I would like a taxi from Saint John
    's college to Pizza Hut Fen Ditton.",
            "What time do you want to leave and
    what time do you want to arrive by?",
            "I want to leave after 17:15.",
            "Booking completed! your taxi will
    be blue honda Contact number is
    07218068540",
            "Thank you for all the help! I
    appreciate it.",
            "You are welcome. Is there anything
    else I can help you with today?",
            "No, I am all set. Have a nice day.
    Bye.",
            "you too! thank you"
        ],
        "user_information": {
            "slots": {
                "taxi-departure": [
                    "saint johns college",
                    "saint john's college"
                ],
                "taxi-destination": [
                    "pizza hut fen ditton"
                ],
                "taxi-leaveat": [
                    "17:15"
                ]
            },
            "requests": []
        },
        "agent_information": {}
    },
```

8.3 Errors in Dataset

Dialogue: User: Is there one in the moderate price range with free internet?

State: * hotel-internet: yes

Description of the state: whether the hotel has internet. Possible values include: free, no, yes

8.4 Frequency of nodes visited

Node Description	Frequency
Determine Request Type	19
Search for Information	31
Provide Initial Search Status & Offer Proactive Updates	16
Provide Information (If Details Available)	25
Ask for Confirmation Before Finalizing a Booking	8
Ask for Accommodation Check-in/Check-out Dates & Number of Guests	8

Table 5: Frequency of selected instruction graph nodes across training dialogues.