LEARNING FROM WHAT USERS DO NOT SAY: SELF-EVOLVING ASSISTANTS BY INTERNALIZING LATENT THOUGHTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly used as interactive assistants, yet their ability to understand users remains limited by partial observability: user utterances often reveal only incomplete information about the underlying intents, preferences, and concerns. In this paper, we study whether such latent user information can serve as privileged supervision during training and be internalized into the assistant for real-world interactions. We propose OASIS, a three-stage framework that learns from latent user thoughts without relying on a stronger teacher model. OASIS first uses user simulators to expose latent user states for supervised cold-start training, and then performs on-policy asymmetric self-distillation, where the same assistant learns from the information gap between a privileged view with latent thoughts and an observable view with dialogue only. Experiments on open-ended general assistance and goal-oriented commercial customer service show that OASIS consistently improves assistant performance across model scales and evaluation settings, with on-policy self-distillation providing up to 52.4% and 22% further improvements over the SFT initialization on WildBench and DianJin-CSC, respectively. These results demonstrate that latent user information can be effectively internalized into the assistant through asymmetric self-distillation.

1 INTRODUCTION

Large language models (LLMs) are increasingly deployed as interactive assistants in customer service, recommendation, planning, and other real-world applications. As interactions become longer and more complex, success requires more than generating a locally appropriate response to the latest user utterance: an effective assistant must understand what the user is trying to achieve, track how their needs evolve, and select responses actively move the interaction towards the user's goal. Recent work has explored proactive assistance, personalized alignment, and agentic behavior in multi-turn interactions (Sharma et al., 2024; Butala et al., 2024; Lu et al., 2025; Wu et al., 2025). Nevertheless, current models often remain reactive responders rather than proactive problem solvers, and can deviate from the user's actual needs when important information is not explicitly stated.

We argue that a fundamental source of this limitation is the *partial observability of user states*. Recent studies on personalized and proactive interaction have shown that user preferences and intentions are often incomplete, implicit, and dynamically revealed over the course of a conversation, motivating assistants to reason beyond the information explicitly stated in the current utterance (Wu et al., 2025; Luo et al., 2026). In real interactions, observable utterances are driven by latent factors such as underlying intents, evolving preferences, concerns, constraints, and decision states. Recent evidence from ThoughtTrace further shows that users' self-reported thoughts contain information that is distinct from observable messages and difficult for even strong LLMs to fully recover from dialogue context alone (Jin et al., 2026). As demonstrated in Figure 1, a customer may only say that she wants a skincare gift for her mother and that her mother does not like overly fancy products, while the factors that actually determine a suitable recommendation: a limited budget, sensitive skin, and a preference for practicality, remain unstated. An assistant that relies only on surface cues may therefore recommend a premium product, whereas accounting for these hidden constraints leads to a fundamentally different response strategy. We refer to such unexpressed user-side information as *latent user thoughts*. Without effectively accounting for these latent states, assistants may fail to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: A commercial shopping case showing that latent user thoughts can substantially change the assistant's decision. While the traditional assistant generates a general recommendation based on the user's utterances, the thought-aware assistant (OASIS) captures hidden constraints such as budget, sensitive skin, and practical preference, leading to a more appropriate shopping recommendation.

anticipate user needs or correct an interaction when it deviates from the user's true objective. This raises a central challenge: **how can we train assistants to benefit from user information that matters for interaction but is not directly observable at deployment?**

Building on the above insights, we propose **OASIS** (**O**n-policy **A**symmetric **S**elf-distillation of **I**mplicit **S**tates), a three-stage framework that uses latent user thoughts as training-time privileged information and internalizes their benefits into the assistant itself. As depicted in Figure 2, our approach consists of three key components. First, we introduce reasoning-capable user simulators to make user thoughts accessible during training. By using their chain-of-thought generation, the simulators produce a latent thought before each user utterance, providing a natural source of user privileged information. For general assistance, we train a user simulator on ThoughtTrace (Jin et al., 2026), which contains real users' self-reported thoughts alongside conversations on diverse everyday tasks. For commercial customer service, we adopt a user simulator trained on real user–agent interaction logs from industrial scenarios (Chen et al., 2026a), which similarly generates an implicit user thought before producing utterance. These simulators allow us to construct multi-turn trajectories in which the assistant can access both observable utterances and their underlying latent user states during training. Second, we use these trajectories for supervised fine-tuning, providing a cold start that teaches the assistant to associate conversational evidence with latent user needs and more appropriate response strategies. Third, because offline SFT trajectories cannot fully capture the state distribution induced by the evolving assistant itself, we further introduce on-policy asymmetric self-distillation. The deployment view interacts using only observable dialogue, while a privileged view of the *same assistant* additionally observes latent user thoughts at the resulting on-policy states and provides supervision to the deployment view. The resulting asymmetry comes from information rather than model capacity: OASIS requires no stronger external teacher, and continuously transfers user understanding available under privileged information into a policy that must operate without it.

We evaluate OASIS in two complementary settings: open-ended general assistance on WildBench (Lin et al., 2025) and goal-oriented commercial customer service on DianJin-CSC (Zhu et al., 2026), using both Qwen3-4B and Qwen3-8B. OASIS consistently improves assistant performance across model scales and evaluation protocols. Latent-thought SFT provides substantial gains over the corresponding base models, while on-policy asymmetric self-distillation further improves the SFT initialization by up to 52.4% on WildBench and 22% in response-strategy accuracy on DianJin-CSC. These improvements are consistent across two independent WildBench judges and under both reference-

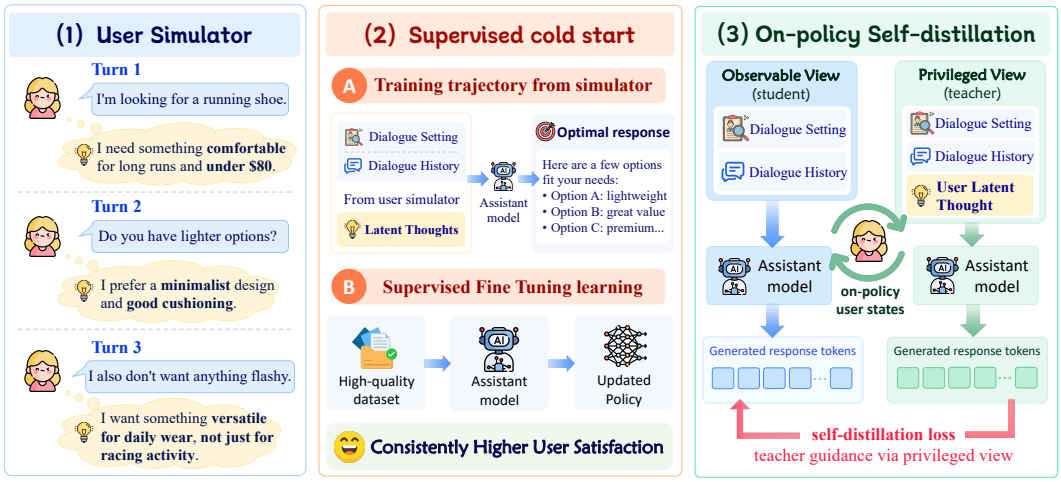


Figure 2: Overview of OASIS, a framework for internalizing latent user information into conversational assistants. The pipeline consists of three stages: **(1) User Simulator**: multi-turn user interactions are generated together with the latent thoughts underlying each utterance. **(2) Supervised Cold Start**: simulator trajectories provide privileged supervision for learning high-quality response strategies through supervised fine-tuning. **(3) On-policy Asymmetric Self-distillation**: the same assistant interacts on-policy under an observable view with dialogue only and a privileged view additionally containing latent user thoughts from user simulator. The privileged policy then guides the observable policy through token-level self-distillation, enabling the assistant to further internalize user-side information without an external teacher model.

context and the stricter generated-context evaluation on DianJin-CSC. Importantly, all privileged information is removed at evaluation time: the resulting assistant requires neither latent thoughts, user simulators, nor additional teacher models, demonstrating that user privileged information can be effectively internalized into the assistant's own capabilities.

The major contributions of this paper are summarized as follows:

- We introduce a new perspective on assistant self-evolution by treating latent user thoughts as training privileged information. By leveraging reasoning-capable user simulators, we expose user-side information that is unavailable in ordinary human–assistant interaction and use it to improve the assistant rather than merely simulate the user.
- We propose OASIS, a three-stage framework that combines latent-thought SFT with on-policy asymmetric self-distillation. The same assistant operates under privileged and deployment views, allowing user-side information to be progressively transferred through information asymmetry without relying on a stronger external teacher.
- We systematically validate OASIS across open-ended assistance and commercial customer service. Experiments across model scales and evaluation settings show substantial gains from latent-thought supervision and consistent further improvements from on-policy self-distillation, all while requiring only standard dialogue context at deployment.

2 Self-Evolving Assistants from Privileged Latent User States

In this section, we introduce OASIS, a three-stage framework that applies latent user states as privileged supervision during training and internalizes their benefits into an assistant that operates solely from observable dialogue at deployment. OASIS is built upon a reasoning-capable user simulator that exposes the implicit thoughts underlying each user utterance. Given such a simulator, we first construct privileged interaction trajectories for a supervised cold start, and then perform on-policy asymmetric self-distillation on multi-turn trajectories induced by the evolving assistant itself. The following subsections detail each component of OASIS.

2.1 PROBLEM FORMULATION

We formalize a multi-turn interaction between a user and an assistant. At turn t, let u_t denote the user utterance and a_t denote the assistant response. The observable dialogue history available to a deployed assistant is $h_t = (u_1, a_1, \ldots, u_{t-1}, a_{t-1}, u_t)$. Before producing u_t, the user is associated with an implicit state z_t that captures interaction-relevant information not necessarily expressed in the utterance, such as the underlying intent, preference, concern, constraint, or current decision state. We refer to z_t as the *latent user thought*.

For real users, z_t is intrinsically unobserved. However, during training we construct a user simulator $\mathcal{U}_\phi$ that explicitly generates a latent thought before each utterance. Let g denote the user-specific condition available to the simulator, such as a user profile, task goal, or ground-truth interaction state. The simulator generation process can be factorized as

$$p_\phi(z_t, u_t \mid g, h_{t-1}) = p_\phi(z_t \mid g, h_{t-1})\, p_\phi(u_t \mid g, h_{t-1}, z_t).$$

This formulation makes the otherwise hidden user reasoning accessible during training while preserving u_t as the only user-side observation available in real deployment.

We define two information views of the same assistant. Let $z_{\le t} = (z_1, \ldots, z_t)$ denote the latent-thought history associated with the conversation. The deployment view observes only the dialogue, $\pi_\theta^{\mathrm{D}}(y_t \mid h_t) = p_\theta(y_t \mid h_t)$, while the privileged view additionally receives the latent user thoughts, $\pi_\theta^{\mathrm{P}}(y_t \mid h_t, z_{\le t}) = p_\theta(y_t \mid h_t, z_{\le t})$. Here, y_t denotes the assistant completion, including its internal reasoning and final response. In practice, we encourage the assistant to first identify the user's current blocker or latent need, then determine an appropriate response strategy, and produce the answer.

The central objective of OASIS is to transfer the behavioral advantage induced by $z_{\le t}$ from the privileged view into the deployment view, such that the final assistant no longer requires latent thoughts when interacting with real users.

2.2 LATENT-THOUGHT USER SIMULATOR

A prerequisite of OASIS is a user simulator whose generation process exposes not only what the user says but also *why the user says it*. Standard user simulators typically optimize for observable utterance generation, whereas we explicitly reshape the simulator's chain-of-thought process so that it generates an implicit user thought before producing each utterance.

To construct such training data, latent thoughts can be obtained in two complementary ways. When human thought annotations are available, we directly use users' self-reported internal thoughts associated with each dialogue turn. When the user profile, task objective, or ground-truth behavioral state is known, we can instead synthesize a plausible reasoning trace conditioned on this privileged information and the observed interaction. The resulting simulator training set takes the form $\mathcal{D}_\mathcal{U} = \{(g, h_{t-1}, z_t, u_t)\}$, and the simulator is trained with standard supervised fine-tuning:

$$\mathcal{L}_\mathcal{U}(\phi) = -\mathbb{E}_{(g,h_{t-1},z_t,u_t)\sim\mathcal{D}_\mathcal{U}} \left[\log p_\phi(z_t, u_t \mid g, h_{t-1})\right].$$

2.3 STAGE I: PRIVILEGED SUPERVISED COLD START

Although the latent-thought simulator makes privileged user information available, directly performing on-policy optimization from the base assistant can be unstable: the initial policy may visit low-quality interaction states and may not yet know how to exploit the additional user-side information. Follow the prior experience (Guo et al., 2025), we establish a initialization through privileged trajectory generation. Specifically: Let θ_0 denote the parameters of the assistant backbone before OASIS training. At each interaction state, we provide the same assistant with both the observable history h_t and latent user thoughts $z_{\le t}$. The privileged assistant identifies the user's current blocker or need, selects an appropriate response strategy, and generates the response: $y_t^+ \sim \pi_{\theta_0}^{\mathrm{P}}(\cdot \mid h_t, z_{\le t})$. The generated response is then returned to the user simulator, which produces the next latent thought and utterance. In this way, the assistant and simulator interact for multiple turns to form a complete trajectory τ. Importantly, no stronger teacher model is introduced in this process. The target-generating policy uses exactly the same assistant backbone that will subsequently be trained and its advantage comes solely from privileged access to the user's hidden state. We retain high-quality trajectories for which the simulated interaction reaches a satisfactory final state, yielding a filtered trajectory set $\mathcal{T}^+$.

From these trajectories, we construct the training dataset $\mathcal{D}_{\mathrm{SFT}} = \{(h_t, y_t^+) \mid \tau \in \mathcal{T}^+\}$. The latent thought is used to produce the privileged target but is removed from the deployment input, forcing the assistant model to reproduce the resulting user-aware reasoning.

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_{(h,y^+)\sim\mathcal{D}_{\mathrm{SFT}}}\left[\frac{1}{|y^+|}\sum_{n=1}^{|y^+|}\log \pi_\theta^{\mathrm{D}}(y_n^+ \mid h, y_{<n}^+)\right].$$

Although implemented with SFT, this stage constitutes an initial form of self-improvement: the same model applies privileged user information to construct better interaction strategies and is trained to reproduce these strategies without observing that information. Empirically, this stage already yields substantial improvements, confirming that latent user thoughts provide a useful supervision signal. $\mathcal{D}_{\mathrm{SFT}}$ consists of offline trajectories collected before the assistant is updated. Multi-turn interaction is inherently policy-dependent: changing an assistant response changes subsequent user behavior and therefore changes the states the assistant will encounter. The fixed SFT distribution cannot fully cover the state distribution induced by the evolving deployment policy. We address this mismatch in the second stage.

2.4 Stage II: On-policy Asymmetric Self-distillation

Although privileged SFT provides a strong initialization, it relies on a fixed set of offline trajectories and cannot fully cover the interaction states induced by the evolving assistant. We therefore further perform on-policy asymmetric self-distillation. The student policy π_θ is initialized from the SFT checkpoint, while a frozen reference policy π_{ref} is initialized from the same checkpoint. The student interacts with the user simulator using only observable dialogue, thereby generating on-policy multi-turn trajectories. For an interaction state $(h_t, z_{\le t})$ and a student-sampled response $\hat{y}_t \sim \pi_\theta(\cdot \mid h_t)$, we evaluate each sampled token with the frozen reference model under two information views:

$$\ell_{t,n}^{\mathrm{D}} = \log \pi_{\mathrm{ref}}(\hat{y}_{t,n} \mid h_t, \hat{y}_{t,<n}), \qquad \ell_{t,n}^{\mathrm{P}} = \log \pi_{\mathrm{ref}}(\hat{y}_{t,n} \mid h_t, z_{\le t}, \hat{y}_{t,<n}).$$

Since the two views share identical model parameters and token prefixes, their difference isolates the contribution of privileged user information. We define the token-level privileged-information gain as $\Delta_{t,n} = \ell_{t,n}^{\mathrm{P}} - \ell_{t,n}^{\mathrm{D}}$. A positive $\Delta_{t,n}$ indicates that access to latent user thoughts makes the sampled behavior more compatible with the reference policy, while a negative value indicates the opposite. We convert this information gain into a detached policy-gradient advantage. Let $m_{t,n}$ indicate whether a valid latent thought is available for the target token. The training objective is

$$\mathcal{L}_{\mathrm{PI}}(\theta) = -\mathbb{E}_{\tau\sim\pi_\theta,\mathcal{U}_\phi}\left[\frac{1}{N_{\mathrm{valid}}}\sum_{t,n}\mathrm{sg}(m_{t,n}\Delta_{t,n})\log \pi_\theta(\hat{y}_{t,n} \mid h_t, \hat{y}_{t,<n})\right].$$

This reinforces on-policy behaviors that become more likely under privileged user information and suppresses those that become less likely, while the trainable student itself never observes the latent thoughts. To control policy drift, we additionally regularize the student towards the observable-view frozen reference using the low-variance KL estimator. Let $\delta_{t,n} = \ell_{t,n}^{\mathrm{D}} - \log \pi_\theta(\hat{y}_{t,n} \mid h_t, \hat{y}_{t,<n})$ and $\widehat{D}_{\mathrm{KL}}^{\mathrm{LV}}(t,n) = \exp(\delta_{t,n}) - \delta_{t,n} - 1$. The final objective is

$$\mathcal{L}_{\mathrm{OASIS}}(\theta) = \mathcal{L}_{\mathrm{PI}}(\theta) + \beta\,\mathbb{E}_{\tau\sim\pi_\theta,\mathcal{U}_\phi}\left[\frac{1}{N}\sum_{t,n}\widehat{D}_{\mathrm{KL}}^{\mathrm{LV}}(t,n)\right].$$

3 Experiments

We evaluate OASIS in two complementary settings: open-ended general assistance and goal-oriented commercial customer service. Our experiments examine whether latent user thoughts provide effective supervision, whether on-policy asymmetric self-distillation brings further improvements.

3.1 Experimental Settings

For general assistance, we evaluate on WildBench (Lin et al., 2025), a challenging benchmark constructed from real-world user queries. The task is open-ended response generation: given a

Model	Creative	Planning	Math / Data	Info / Advice	Coding / Debug	WB Score
Qwen3-8B	-15.45 / 12.30	-19.01 / 8.64	-25.48 / 3.97	-14.75 / 12.57	-29.53 / -4.62	-19.51 / 8.18
+ OPD	-13.70 / 28.99	-17.13 / 23.98	-22.70 / 16.43	-13.32 / 25.40	-31.51 / 6.13	-17.91 / 22.01
+ OASIS-SFT	29.66 / 50.80	26.43 / 46.64	21.90 / 41.03	26.78 / 48.47	17.92 / 32.83	24.71 / 45.53
+ **OASIS**	**38.14 / 58.40**	**41.05 / 56.35**	**33.81 / 52.38**	**40.45 / 56.14**	**37.45 / 46.51**	**37.66 / 54.67**
Qwen3-4B	-22.17 / 5.84	-23.35 / 3.32	-28.57 / 0.16	-16.29 / 9.55	-33.49 / -9.91	-23.48 / 3.20
+ OPD	-16.64 / 10.39	-20.12 / 6.01	-24.92 / 2.30	-13.96 / 11.58	-33.40 / -7.55	-20.62 / 5.84
+ OASIS-SFT	22.17 / 32.87	17.22 / 27.83	11.59 / 24.31	18.71 / 28.02	3.02 / 15.86	15.55 / 27.40
+ **OASIS**	**25.87 / 45.48**	**19.04 / 39.10**	**14.44 / 32.94**	**21.14 / 40.30**	**7.92 / 23.02**	**18.28 / 38.01**

Table 1: Results on general-task benchmark WildBench. Scores are evaluated by GPT-OSS-120B (left) and Qwen3-235B (right), respectively. Across both Qwen3-8B and Qwen3-4B, latent-thought-aware SFT yields substantial improvements over the base model and applying OPD alone. Building on this initialization, our on-policy asymmetric self-distillation further improves performance across all task categories and achieves the best overall WB Score under both judges.

user query, together with conversation history when available, the assistant must generate a helpful response that satisfies the user's intent and task-specific requirements. The benchmark covers diverse scenarios, including creative writing, planning, math and data analysis, information seeking, and coding and debugging. Following its evaluation protocol, we report category-level scores and the final WB Score. To reduce evaluator bias, we independently evaluate each model with GPT-OSS-120B (Agarwal et al., 2025) and Qwen3-235B (Yang et al., 2025) as judges.

For commercial customer service, we evaluate on DianJin-CSC (Zhu et al., 2026), a multi-turn customer-service benchmark that jointly evaluates response-strategy selection and response generation. At each turn, the assistant must first identify an appropriate service strategy according to the customer's current state and dialogue history, and then generate a response consistent with that strategy. We report response-strategy accuracy (ACC) together with BLEU-2/4, ROUGE-L, BERTScore, and BLEURT for response quality. Following the original benchmark, we consider two evaluation settings: reference context, where each turn conditions on the gold dialogue history, and the more challenging generated context, where the model conditions on its own previous responses and therefore must maintain effective interaction over multiple turns.

3.2 User Simulators and Training Details

For open-ended assistance, we train a reasoning-capable user simulator on about 10K thought-annotated examples from ThoughtTrace (Jin et al., 2026), whose thought annotations are collected from real users during human–AI interactions. At each turn, the simulator first generates an implicit user thought and then produces the observable utterance. These thoughts capture user privileged information that may remain unspoken in the dialogue, such as the current blocker, latent preference, concern, or unmet need. We train the simulator base on Qwen3-8B with supervised fine-tuning using LLaMA-Factory (Zheng et al., 2024). More details are provided in Appendix A. We further evaluate the faithfulness of the generated latent thoughts against held-out human-reported thoughts. As shown in Table 2, the trained simulator improves over the original backbone, indicating that its generated thoughts provide reasonable proxies for users' underlying states. Finally, we construct training trajectories using samples from WildChat (Zhao et al., 2024).

For commercial customer service, we adopt the Qwen2.5-7B-Instruct user simulator from Chen et al. (2026a), which is trained on real industrial user–agent interactions through supervised fine-tuning followed by multi-turn RL. The simulator generates an internal rationale and evolving intent state before each utterance and we use the training split of DianJin-CSC (Zhu et al., 2026) to construct the training trajectories.

Given the two simulators, Stage I collects multi-turn trajectories in which the assistant has privileged access to latent user thoughts, and then performs cold start training after removing these thoughts from the assistant input. Stage II initializes from the resulting SFT checkpoint and performs on-policy asymmetric self-distillation as described in Section 2. Training and evaluation are conducted on

Model	*Evaluation with reference context*						*Evaluation with generated context*					
	B-2	B-4	R-L	BS	BR	ACC	B-2	B-4	R-L	BS	BR	ACC
Qwen3-8B	7.94	2.10	19.66	68.17	53.23	15.59	6.90	1.57	18.89	67.46	52.58	13.11
+ OPD	9.02	2.95	21.59	68.65	53.39	17.70	7.79	2.39	20.51	67.72	52.53	15.41
+ OASIS-SFT	10.96	4.02	22.58	69.36	53.52	33.71	9.00	3.16	21.66	68.99	54.49	33.72
+ **OASIS**	**11.97**	**5.12**	**25.65**	**72.34**	**54.58**	**37.92**	**10.07**	**4.21**	**23.24**	**70.02**	**58.32**	**39.45**
Qwen3-4B	6.97	1.76	17.60	66.48	50.16	15.20	5.86	1.37	16.55	65.83	49.85	14.99
+ OPD	7.64	2.20	19.03	67.12	50.74	19.13	6.77	1.86	18.55	66.73	50.84	20.44
+ OASIS-SFT	10.29	3.64	21.03	68.82	52.03	25.99	8.69	3.01	20.84	69.10	54.62	29.33
+ **OASIS**	**11.31**	**4.28**	**22.65**	**69.29**	**52.72**	**31.24**	**9.82**	**3.69**	**22.91**	**69.67**	**55.01**	**35.78**

Table 3: Results on the commercial customer-service benchmark DianJin-CSC under reference-context and generated-context evaluation. Latent-thought supervision consistently improves performance across both model scales, while full method of OASIS further achieves the strongest results under both settings. The gains remain evident even in the stricter generated-context evaluation, indicating better policy quality and more robust multi-turn interaction in commercial scenarios.

NVIDIA A800 GPUs, with inference accelerated using vLLM (Kwon et al., 2023). Additional implementation details are provided in Appendix A.3.

We further validate whether the latent thoughts exposed by our user simulators provide reasonable traces for users' underlying states. In the general domain, we evaluate the simulator on 500 turns from held-out ThoughtTrace users, using the self-reported thoughts as human references. Three annotators with master's degrees in engineering independently evaluate each generated thought. To reduce reference-induced bias, annotators first assess plausibility (**Plaus.**) and context consistency (**Cons.**) based only on the preceding dialogue, and are then shown the human reported thought to evaluate human-thought alignment (**Align.**). As shown in Table 2, simulator training consistently improves all three dimensions over the original backbone, indicating that the generated thoughts better capture plausible and contextually grounded latent user states. For the commercial domain, we directly adopt the user simulator from TRACER, whose intermediate rationales are designed as interpretable proxies for latent user decision states. The original study validates these rationales through human plausibility evaluation and demonstrates strong behavioral alignment between simulated and real customer trajectories. Together, these results support the use of simulated latent thoughts as meaningful privileged supervision in both domains.

Model	Plaus.	Cons.	Align.	$\geq$4 (%)
Qwen3-8B	3.72	3.86	3.41	69.2
User Simulator	**4.31**	**4.39**	**3.96**	**86.8**

Table 2: Faithfulness evaluation of latent thoughts generated by the general-domain user simulator.

3.3 Overall Results

Results in Tables 1 and 3 show OASIS's superior performance across different scenarios.

OASIS consistently improves assistant performance across domains, model scales, and evaluation protocols. The full OASIS model achieves the strongest overall performance for both Qwen3 series model across the two benchmarks. On WildBench, the improvements are consistently observed under two independent judges and span diverse task categories. On DianJin-CSC, OASIS improves not only response-generation quality but also response-strategy accuracy under both two evaluation settings. These gains suggest that learning from latent user states benefits both open-ended assistance and goal-oriented interaction, rather than being specific to a particular task format or evaluator.

Latent user thoughts provide the primary supervision signal for improving user-aware behavior. Across both domains, the cold start stage accounts for the major improvement over the original backbone, whereas directly applying OPD to the base model produces smaller gains. This contrast indicates that the improvements do not simply arise from additional optimization. The SFT cold start allows the model to first associate observable conversational cues with underlying user needs and appropriate response strategies, establishing a strong user-aware policy before on-policy refinement.

On-policy asymmetric self-distillation provides consistent further gains beyond SFT policy. OPD further improves the latent-thought SFT initialization across both model scales and benchmarks,

with clear gains under generated-context evaluation on DianJin-CSC. Since this setting exposes the model to histories induced by its own responses, the stronger improvement suggests that on-policy training better complements offline SFT under policy-induced distribution shifts.

4 Analysis

4.1 Ablation Studies

Source of performance gains. Do the gains come from latent user thoughts or simply additional training? We construct two controlled variants to isolate the source of improvement in each stage of OASIS. First, for the SFT stage, we replace our privileged trajectories with approximately the same amount of standard multi-turn dialogue data sampled from WildChat, while keeping the backbone and training procedure unchanged. This control preserves the effect of additional supervised optimization while removing the latent-state supervision that distinguishes OASIS-SFT. As shown in Table 5, standard dialogue SFT provides little improvement and substantially underperforms OASIS-SFT for both model scales. The consistent gap indicates that simply exposing the model to more multi-turn dialogue is insufficient to reproduce the gains of our method. Instead, the improvement is closely associated with supervision derived from user-side latent thoughts, which provides training signals beyond those available in ordinary dialogue trajectories.

Model	WB Score
Qwen3-8B	-19.51
+ Standard-SFT	-5.62 (↑13.9)
+ OASIS-SFT	24.71 (↑44.2)
+ OPD w/o Privileged	24.98 (↑44.5)
+ OPD (Ours)	**37.66 (↑57.2)**
Qwen3-4B	-23.48
+ Standard-SFT	-13.10 (↑10.4)
+ OASIS-SFT	15.55 (↑39.0)
+ OPD w/o Privileged	15.71 (↑39.2)
+ OPD (Ours)	**18.28 (↑41.8)**

Table 4: Ablation results on WildBench. Percentages indicate gains over the corresponding base model.

Second, we remove the information asymmetry from the OPD stage by providing the frozen reference model with exactly the same observable context in both views, while otherwise retaining the same on-policy training procedure. Without latent thoughts, this additional training brings only marginal changes over the SFT initialization, whereas full OPD yields clear further improvements. Together, these controls indicate that the effectiveness of both stages is primarily associated with the additional user-side information introduced by OASIS, rather than simply performing SFT or other optimization strategies. Details are provided in Appendix A.1.

4.2 Effect of Latent User Thoughts

Before training OASIS, we first examine whether user-side latent thoughts provide useful information for assistant decision making at all. We conduct a controlled experiment on the general-assistance setting using the original Qwen3-8B without any additional training. For each interaction, we compare three conditions: **No User CoT**, where the assistant observes only the dialogue; **Matched User CoT**, where it additionally receives the latent thought generated for the current user state; and **Shuffled User CoT**, where the thought is randomly replaced by one from another example.

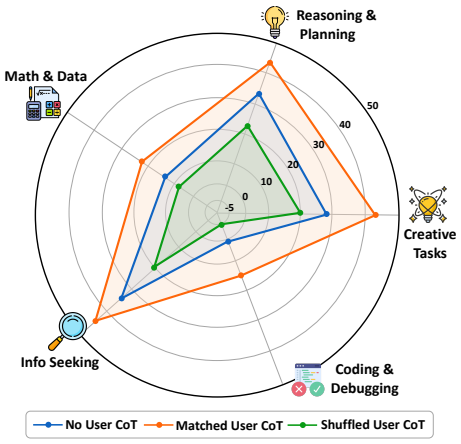


Figure 3: Effect of user latent thoughts on assistant performance. Matched thoughts consistently outperform dialogue-only and shuffled-thought settings.

As shown in Figure 3, providing matched user thoughts consistently improves performance across task categories, with particularly clear gains in reasoning, information seeking, and creative tasks. In contrast, shuffled thoughts substantially reduce the benefit and can even hurt performance. This comparison suggests that the improvement does not simply come from providing additional textual context; rather, it depends on user-state information that is relevant to the current interaction. These results motivate OASIS to use such latent thoughts as privileged supervision during training, while removing their dependency at deployment.

5 Related Work

User Modeling and Personalized Dialogue Understanding users beyond their immediate utterances has long been a central problem in conversational systems. Traditional task-oriented dialogue systems explicitly maintain dialogue states to track user goals and constraints, while personalized dialogue systems further incorporate personas, preferences, interaction histories, and long-term memories to adapt responses to individual users (Chen et al., 2024; Wang et al., 2024; Chen et al., 2025). With the emergence of LLM-based assistants, recent work has focused on preferences that are implicit and dynamically revealed through multi-turn interaction. For example, Wu et al. (2025) train LLMs to infer unspoken individual preferences from interaction and adapt subsequent responses, while other approaches perform personalization through preference elicitation, profile extraction, or persistent conversational memory (Baskar et al., 2025; Liu et al., 2025; Luo et al., 2026). OASIS takes a complementary perspective: instead of requiring such latent information to be explicitly available or maintained at deployment, we use it as privileged supervision during training and internalize its benefits into an assistant that ultimately operates from ordinary dialogue alone.

User Simulation and Reasoning Traces User simulation provides a scalable means of training and evaluating conversational agents without repeatedly interacting with real users. Recent LLM-based simulators have moved beyond traditional task-oriented settings towards richer multi-turn interactions, using human conversations, user profiles, goals, and behavioral histories to better approximate real user behavior (Kong et al., 2024; Sekulić et al., 2024; Wang et al., 2025; Chen et al., 2026b; Naous et al., 2026). Beyond reproducing observable utterances, emerging work has begun to model the latent reasoning that drives user behavior. Jin et al. (2026), for example, collect real human–AI conversations together with users' self-reported thoughts and show that these thoughts contain information not fully recoverable from observable dialogue. Other studies reconstruct or synthesize user-side reasoning traces to improve the fidelity of user simulation (Wu et al., 2026; Lu et al., 2026). While these works primarily use latent reasoning to better model, predict, or simulate users, we instead exploit the information exposed by user simulators to train a better assistant. Our optimization is also related to recent work on learning from privileged information through on-policy self-distillation (Zhao et al., 2026; Penaloza et al., 2026; Yuan et al., 2026). OASIS differs in the source and role of this privileged information: the additional information is the dynamically evolving latent state of the user, exposed by a user simulator as the interaction unfolds. We use this CoT information to supervise the assistant on its own interaction distribution and ultimately distill the resulting behavior into a deployment policy that no longer has access to either the latent thoughts or the simulator.

6 Discussion

Conclusions. We present OASIS, a three-stage framework for improving interactive assistants by learning from user information that is unavailable in ordinary dialogue. Our key insight is to use reasoning-capable user simulators to expose latent user thoughts as training-time privileged information, and progressively transfer their benefits into an assistant that ultimately operates from observable dialogue alone. OASIS first uses privileged interaction trajectories for an SFT cold start and then performs on-policy asymmetric self-distillation, where the same model learns from the information gap between privileged and deployment views without relying on a stronger teacher. Experiments across open-ended assistance and commercial customer service consistently show that latent user thoughts provide valuable supervision, while on-policy learning further improves upon offline SFT. Our results suggest that user simulators can serve not only as environments for evaluation or data generation, but also as a source of privileged supervision for developing assistants that better account for users' implicit needs. We hope OASIS will inspire future work on leveraging user-side privileged information to build assistants better understand and adapt to users' evolving latent needs.

Limitations. Although our faithfulness evaluation shows agreement between simulated and human-reported thoughts, such thoughts remain proxies for users' underlying states and may not fully capture the complexity or ambiguity of real human cognition. In addition, our experiments focus on two interaction domains and Qwen3 series backbones, leaving broader validation across model families, domains, and longer-horizon interactions in the future. Future work could explore more scalable ways of obtaining such supervision and investigate how robustly the learned capabilities transfer when simulator behavior differs from real users.

AI USE STATEMENT

Large Language Models were used only for limited auxiliary purposes during the preparation of this manuscript. In particular, the authors used LLMs to assist with language editing, including grammar correction, spelling checks, as well as to provide coding assistance during the experimental implementation process. All generated text and code suggestions were carefully reviewed, revised, tested, and approved by the authors. The LLMs did not contribute to the formulation of the research questions, conceptual ideas, experimental design, interpretation of results, or scientific conclusions. All research-level contributions, including the ideas, methodology, analysis, and final decisions, were made by the whole authors.

ETHICS STATEMENT

This work is conducted using publicly available and appropriately licensed datasets and models in compliance with their intended usage. No private, personally identifiable, or sensitive user information is collected or used in our experiments. All evaluations are performed on existing research benchmarks, and the study does not involve human subjects or interventions that raise additional privacy or safety concerns.

REPRODUCIBILITY STATEMENT

All experiments in this work are reproducible. Our training is conducted on publicly available datasets, and evaluation is performed on popular benchmarks. Key implementation details, including random seeds, model checkpoints, training configurations, and evaluation settings, are specified in the main paper and appendix. We also provide the code in the supplementary materials to facilitate reproduction of our results.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Sarvesh Baskar, Manas Gaur, Srinivasan Parthasarathy, and Tanmay Tulsidas Verlekar. (cper) from guessing to asking: An approach to resolving persona knowledge gap in llms during multi-turn conversations. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 4: Student Research Workshop)*, pp. 435–447, 2025.

Yash Butala, Siddhant Garg, Pratyay Banerjee, and Amita Misra. Promise: A proactive multi-turn dialogue dataset for information-seeking intent resolution. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 1774–1789, 2024.

Geng Chen, Ruotong Pan, Zhirui Yang, Qiqi He, Jiawei Chen, Zhang Yunfei, Chongyuan Chen, Minxuan Lv, Zheng Yang, Win-Bin Huang, Xiangyu Wu, and Wenwu Ou. Beyond surface style: Aligning multi-turn user simulators with behavioral consistency, 2026a.

Jiawei Chen, Xinyan Guan, Qianhao Yuan, Mo Guozhao, Weixiang Zhou, Yaojie Lu, Hongyu Lin, Ben He, Le Sun, and Xianpei Han. Consistentchat: Building skeleton-guided consistent multi-turn dialogues for large language models from scratch. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 8426–8452, 2025.

Jiawei Chen, Ruoxi Xu, Boxi Cao, Ruotong Pan, Yunfei Zhang, Yifei Hu, Yong Du, Tingting Gao, Yaojie Lu, Yingfei Sun, et al. Towards real-world human behavior simulation: Benchmarking large language models on long-horizon, cross-scenario, heterogeneous behavior traces. *arXiv preprint arXiv:2604.08362*, 2026b.

Yi-Pei Chen, Noriki Nishida, Hideki Nakayama, and Yuji Matsumoto. Recent trends in personalized dialogue generation: A review of datasets, methodologies, and evaluations. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 13650–13665, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Chuanyang Jin, Binze Li, Haopeng Xie, Cathy Mengying Fang, Tianjian Li, Shayne Longpre, Hongxiang Gu, Maximillian Chen, and Tianmin Shu. Thoughttrace: Understanding user thoughts in real-world llm interactions. *arXiv preprint arXiv:2605.20087*, 2026.

Chuyi Kong, Yaxin Fan, Xiang Wan, Feng Jiang, and Benyou Wang. Platolm: Teaching llms in multi-round dialogue via a user simulator. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7841–7863, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. In *International Conference on Learning Representations*, volume 2025, pp. 47852–47870, 2025.

Dongshuo Liu, Zhijing Wu, Dandan Song, and He-Yan Huang. A persona-aware llm-enhanced framework for multi-session personalized dialogue generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 103–123, 2025.

Yaxi Lu, Shenzhi Yang, Cheng Qian, Guirong Chen, Qinyu Luo, Yesai Wu, Huadong Wang, Xin Cong, Zhong Zhang, Yankai Lin, et al. Proactive agent: Shifting llm agents from reactive responses to active assistance. In *International Conference on Learning Representations*, volume 2025, pp. 47431–47457, 2025.

Yuxuan Lu, Jing Huang, Yan Han, Bingsheng Yao, Sisong Bei, Yaochen Xie, Yisi Sang, Qi He, and Dakuo Wang. Can llm agents simulate multi-turn human behavior? evidence from real online customer behavior data. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43961–43977, 2026.

Jiani Luo, Xiaoyan Zhao, Yang Zhang, Shuyi Miao, Bingbing Xu, Stefan Konigorski, and Tat-Seng Chua. Know you before you speak: User-state modeling for llm personalization in multi-turn conversation. *arXiv preprint arXiv:2605.24647*, 2026.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the dialogue: Training and evaluating user language models. In *International Conference on Learning Representations*, volume 2026, pp. 82079–82103, 2026.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026.

Ivan Sekulić, Silvia Terragni, Victor Guimarães, Nghia Khau, Bruna Guedes, Modestas Filipavicius, Andre Ferreira Manso, and Roland Mathis. Reliable llm-based user simulator for task-oriented dialogue systems. In *Proceedings of the 1st Workshop on Simulating Conversational Intelligence in Chat (SCI-CHAT 2024)*, pp. 19–35, 2024.

Ashish Sharma, Sudha Rao, Chris Brockett, Akanksha Malhotra, Nebojsa Jojic, and William B Dolan. Investigating agency of llms in human-ai collaboration tasks. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1968–1987, 2024.

Jian Wang, Chak Tou Leong, Jiashuo Wang, Dongding Lin, Wenjie Li, and Xiao-Yong Wei. Instruct once, chat consistently in multiple rounds: An efficient tuning framework for dialogue. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3993–4010, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.219. URL https://aclanthology.org/2024.acl-long.219/.

Kuang Wang, Xianfei Li, Shenghao Yang, Li Zhou, Feng Jiang, and Haizhou Li. Know you first and be you better: Modeling human-like user simulators via implicit profiles. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21082–21107, 2025.

Shujin Wu, Yi R Fung, Cheng Qian, Jeonghwan Kim, Dilek Hakkani-Tur, and Heng Ji. Aligning llms with individual preferences via interaction. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 7648–7662, 2025.

Xiaoting Wu, Yi Huang, Chunyang Gao, Mengfei Guo, Jingyu Yao, and Junlan Feng. Thinking alignment of scenario-oriented user simulation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42932–42945, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qianhao Yuan, Jie Lou, Xing Yu, Hongyu Lin, Le Sun, Xianpei Han, and Yaojie Lu. Vision-opd: Learning to see fine details for multimodal llms via on-policy self-distillation. *arXiv preprint arXiv:2605.18740*, 2026.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. In *International Conference on Learning Representations*, volume 2024, pp. 34590–34605, 2024.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Jie Zhu, Huaixia Dou, Junhui Li, Lifan Guo, Feng Chen, Chi Zhang, and Fang Kong. Evaluating, synthesizing, and enhancing for customer support conversation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35185–35194, 2026.

A Experimental Parameters and Details

A.1 Results of Ablation Study

Table 1 reports the full WildBench results underlying the ablation study in Section 4.1. The results show a clear separation between additional training and privileged-information supervision. Standard multi-turn SFT fails to improve over the base model, whereas OASIS-SFT yields substantial gains across nearly all task categories for both Qwen3-8B and Qwen3-4B. Similarly, OPD without privileged information brings only marginal changes over OASIS-SFT, while the full OASIS objective provides consistent further improvements. These detailed results support the conclusion that the gains of OASIS are primarily associated with the use of latent user states rather than additional optimization alone.

Model	Creative	Planning	Math / Data	Info / Advice	Coding / Debug	WB Score
Qwen3-8B	-15.45	-19.01	-25.48	-14.75	-29.53	-19.51
+ Standard-SFT	-12.82	-21.35	-30.63	-19.11	-30.28	-5.62
+ OASIS-SFT	29.66	26.43	21.90	26.78	17.92	24.71
+ OPD w/o Privileged	30.12	26.81	21.57	27.20	17.68	24.98
+ **OASIS**	**38.14**	**41.05**	**33.81**	**40.45**	**37.45**	**37.66**
Qwen3-4B	-22.17	-23.35	-28.57	-16.29	-33.49	-23.48
+ Standard-SFT	-23.62	-31.12	-34.92	-29.65	-36.98	-13.10
+ OASIS-SFT	22.17	17.22	11.59	18.71	3.02	15.55
+ OPD w/o Privileged	21.94	17.51	11.83	18.96	3.34	15.71
+ **OASIS**	**25.87**	**19.04**	**14.44**	**21.14**	**7.92**	**18.28**

Table 5: Results on the general-task benchmark WildBench, evaluated by GPT-OSS-120B. Latent-thought-aware SFT yields substantial improvements over the base model and standard SFT, while OASIS further improves performance by leveraging privileged user information through on-policy asymmetric self-distillation.

A.2 Results of User Thoughts

Table 6 provides the full results for the latent-thought access study in Section 4.2. Directly providing matched user CoT substantially improves performance across all WildBench categories, increasing the overall WB Score from 24.9 to 37.4. In contrast, replacing the thought with a randomly shuffled one reduces the score to 16.97 and consistently hurts performance across categories. This contrast shows that the benefit does not come from simply providing additional textual context, but from latent information that is specifically aligned with the current user's state and interaction.

Settings	Creative	Planning	Math / Data	Info / Advice	Coding / Debug	WB Score
No User CoT	28.83	33.90	13.05	34.13	2.26	24.90
Shuffled User CoT	22.21	25.62	9.32	19.10	-1.79	16.97
Matched User CoT	**44.95**	**45.56**	**23.17**	**42.70**	**16.32**	**37.40**

Table 6: Effect of providing user latent thoughts at inference time on WildBench. Matched user CoT substantially improves performance across all task categories, while shuffled user CoT degrades performance, demonstrating that the gains come from informative and user-specific latent thoughts rather than additional context alone.

A.3 Model Training and Inference Details

All models are trained and evaluated on NVIDIA A800 GPUs. We use LLaMAFactory for supervised fine-tuning and BF16 precision throughout training. For the general-domain user simulator, we use a per-device batch size of 1 with 2 gradient-accumulation steps, a learning rate of 1×10^{-5}, 3 training epochs, cosine learning-rate scheduling, and a warmup ratio of 0.1. The assistant models are

initialized from the corresponding Qwen3-4B or Qwen3-8B checkpoints, and the SFT checkpoint is subsequently used to initialize both the student policy and frozen reference policy in the OPD stage. During evaluation, all model variants use the same decoding configuration within each benchmark to ensure a fair comparison. We use vLLM (version 0.23.0) to accelerate trajectory generation and benchmark inference. Random seeds, checkpoint configurations, and other experiment-specific settings are fixed with 42 across comparable runs. Additional implementation details and complete training configurations are provided in the supplementary materials.

B CASE STUDY

	Case 1: General Assistance — Machine Learning Interview Preparation
User:	I have an ML interview next week and I feel like there is way too much to prepare.
Thought:	***I only have a few evenings left. My ML background is fine, but I am rusty at coding and live coding makes me nervous. I do not want a huge study checklist that I cannot finish.***
Assistant:	Instead of trying to review everything, I would narrow the preparation to three things: coding under time pressure, core ML concepts you may need to explain clearly, and one or two projects you can discuss in depth. A compact five-day plan will probably help more than a comprehensive review. I would also put the largest share of practice time into coding rather than rereading theory.
User:	Yeah, coding is probably the part I am most worried about.
Thought:	***Exactly. I can usually understand solutions when I see them, but I freeze when I have to derive one from scratch. I need concrete practice, not generic advice like "do more LeetCode."***
Assistant:	Then make the coding practice highly structured. For each session, do only two problems: one medium problem with a strict 30-minute limit, followed by one problem from the same pattern without looking at the previous solution. Focus on a small set of patterns—hash maps, two pointers, binary search, trees/DFS, and simple dynamic programming. After each problem, spend five minutes explaining the solution aloud as if the interviewer had asked why it works and what its complexity is.
User:	There is also a project discussion round. Most of my recent work is on a recommender system.
Thought:	***I know the project well, but when someone asks an open-ended question, I tend to start from implementation details and ramble. I am worried that the interviewer will not understand what was actually difficult or important.***
Assistant:	For the project round, do not start from the architecture. Prepare a 90-second story with four steps: problem $\rightarrow$ constraint $\rightarrow$ decision $\rightarrow$ impact. Explain what user or business problem the recommender addressed, what made the setting difficult, which modeling or system decision you personally made, and how you evaluated whether it worked. Only go into architecture details after the interviewer asks.
User:	I think I usually spend too much time explaining the model itself.
Thought:	***That is probably why my answers feel weak. I want them to see that I can make engineering and research decisions, not just train a model.***
Assistant:	A strong project answer is usually less about listing components and more about showing judgment. Prepare two concrete trade-offs you made—for example, accuracy versus latency, model complexity versus iteration speed, or offline metrics versus online behavior. For each one, explain what alternatives you considered, why you chose one, and what evidence confirmed or changed your decision.
User:	What should I do if they ask something I genuinely do not know?
Thought:	***This is the part that makes me panic. I sometimes start guessing because silence feels bad, and then the answer gets worse.***
Assistant:	Do not rush into a guess. Make your reasoning visible instead. You can say that you have not worked with the exact method but explain how you would reason about it. Then state what you know, identify the missing assumption, and derive a tentative answer from first principles. Interviewers can still evaluate a structured reasoning process even when the final answer is incomplete.
User:	Okay. I have tonight free. What would you actually do first?
Thought:	***I need a concrete starting point now. If the plan still feels too large, I will probably procrastinate and do nothing tonight.***
Assistant:	For tonight, keep it to 90 minutes: spend 40 minutes solving one medium coding problem under interview conditions, 20 minutes writing the four-sentence version of your recommender-system story, 20 minutes preparing two project trade-offs and the evidence behind your choices, and 10 minutes listing three questions you hope they ask and three you hope they do not ask. The goal is not to finish interview preparation tonight, but to create a small set of material that you can repeatedly practice.
User:	This is much more manageable. I was basically trying to review all of ML before.
Thought:	***This finally feels doable. I know exactly what to work on tonight, and I feel less anxious because I am practicing the parts that are actually weak.***
Assistant:	That is the right shift. At this point, depth and fluency are more valuable than broad coverage. If you can solve a few common coding patterns calmly, explain one project with clear decisions and trade-offs, and reason transparently when you do not know something, you will be much better prepared than if you spend the week passively reviewing dozens of topics.

Table 7: A multi-turn case in a general assistance scenario. The privileged assistant observes the user's latent thoughts and adapts its strategy to the user's hidden concerns, including limited preparation time, coding anxiety, and difficulty structuring project explanations, leading to a more actionable plan and higher user satisfaction.

Table 8 presents a multi-turn product-consultation scenario, where latent user thoughts reveal hidden preferences and purchase concerns, enabling the assistant to progressively resolve decision barriers and successfully complete the conversion.

	Case 2: Commercial Customer Service — Wireless Earbud Recommendation
User:	I'm looking for some wireless earbuds for my commute. I don't need anything too fancy.
Thought:	***I would like to stay below about $100. My commute is noisy, but I hate when salespeople immediately push the most expensive model. I mostly want something reliable that I will actually use every day.***
Assistant:	For a daily commute, I would compare the AirBuds Lite ($69) and AirBuds Flex ($89) rather than jumping to the Pro model. The Lite is the cheaper option for basic listening, while the Flex adds active noise cancellation, which is likely to be the more meaningful upgrade for commuting.
User:	Does noise cancellation actually make that much difference?
Thought:	***My train is pretty loud, so this might actually matter. But I do not want to pay for a feature that only looks good on the specification sheet.***
Assistant:	For a quiet office, I would not treat it as essential. On a train or subway, however, active noise cancellation can noticeably reduce continuous background noise such as the sound of the carriage or ventilation. Given your use case, I would consider it a practical feature rather than a luxury upgrade.
User:	One issue is that earbuds usually start hurting my ears after a while.
Thought:	***This is actually more important to me than sound quality. My ears are small, and I have bought earbuds before that I stopped using after a week because they were uncomfortable.***
Assistant:	Then I would make fit a deciding factor. The Flex comes with three ear-tip sizes, so I would try the smallest tips first rather than choosing purely based on audio specifications. If comfort is uncertain, I would also avoid paying extra for the Pro until you know that this shape works well for you.
User:	I also switch between my phone and laptop a lot.
Thought:	***I use my phone during the commute and my laptop at work. Reconnecting Bluetooth manually every time is one of those small things that would annoy me every day.***
Assistant:	That makes the Flex a stronger fit because it supports multipoint connectivity. You can keep it connected to both your phone and laptop and switch between them without repeatedly pairing the earbuds. For everyday use, that convenience may matter more than the Pro's incremental improvement in noise cancellation.
User:	So is there any reason I should get the Pro instead?
Thought:	***I think the Flex is enough, but I want reassurance that I am not buying the cheaper model and immediately regretting it.***
Assistant:	The Pro would make sense if stronger noise cancellation or longer battery life were your top priorities and you were comfortable spending $129. Based on what you have described so far—commuting, comfort, switching between devices, and keeping the price reasonable—the Flex already covers the features that are most likely to affect your daily experience. I would not upgrade just for the sake of getting the highest-tier model.
User:	My only concern is still the fit. What if the small tips are uncomfortable too?
Thought:	***I am almost ready to buy, but I do not want to be stuck with another pair of earbuds I cannot wear.***
Assistant:	That is a reasonable concern. The Flex is eligible for a 30-day return in this illustrative example, so you can test the smallest tips during your normal commute instead of judging the fit after only a few minutes. If they remain uncomfortable during longer listening sessions, you can return them rather than forcing yourself to keep a product you will not use.
User:	Okay, the Flex sounds like the right one. Let's go with that.
Thought:	***This feels like the right compromise: under my budget, useful noise cancellation, easier switching between devices, and I can still return it if the fit is bad. I also appreciate that I was not pushed toward the most expensive option.***
Assistant:	Sounds good. I would go with AirBuds Flex at $89, starting with the smallest ear tips. It gives you active noise cancellation for the commute and multipoint connectivity for switching between your phone and laptop, while staying below $100. I can proceed with the Flex selection.

Table 8: A multi-turn case in a commercial customer-service scenario. By observing the user's latent concerns about budget, comfort, device switching, and purchase risk, the privileged assistant progressively resolves the user's decision barriers and guides the interaction toward a successful purchase with high user satisfaction. Product names and specifications are fictional and used only for illustration.