TOWARDS USER SIMULATORS THAT STAY ON TASK AND STILL SOUND NATURAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Users increasingly rely on AI systems for extended, multi-turn tasks, yet evidence indicates that LLM-based system performance degrades over prolonged interaction. To evaluate and improve AI assistants in such settings, we need user simulators that faithfully follow complex user intents while exhibiting realistic conversational behavior. Existing approaches currently manifest a trade-off: prompted assistants can follow complex user intents but lack behavioral fidelity, while specialized user simulators better approximate conversational behavior but struggle to remain faithful to complex intents. In this paper, we propose a reinforcement learning (RL) recipe that jointly rewards task intent adherence and behavioral naturalness. A crucial contribution of the method is operationalizing naturalness with a lightweight discriminator trained in an alternating, adversarial loop with the user simulator. We release the resulting model, **UserLM-2.0**, which produces valid simulated conversations (i.e., meeting both task-faithfulness and naturalness criteria) 46.2% of the time, over three times the rate of the strongest prior user simulator (14.1%).

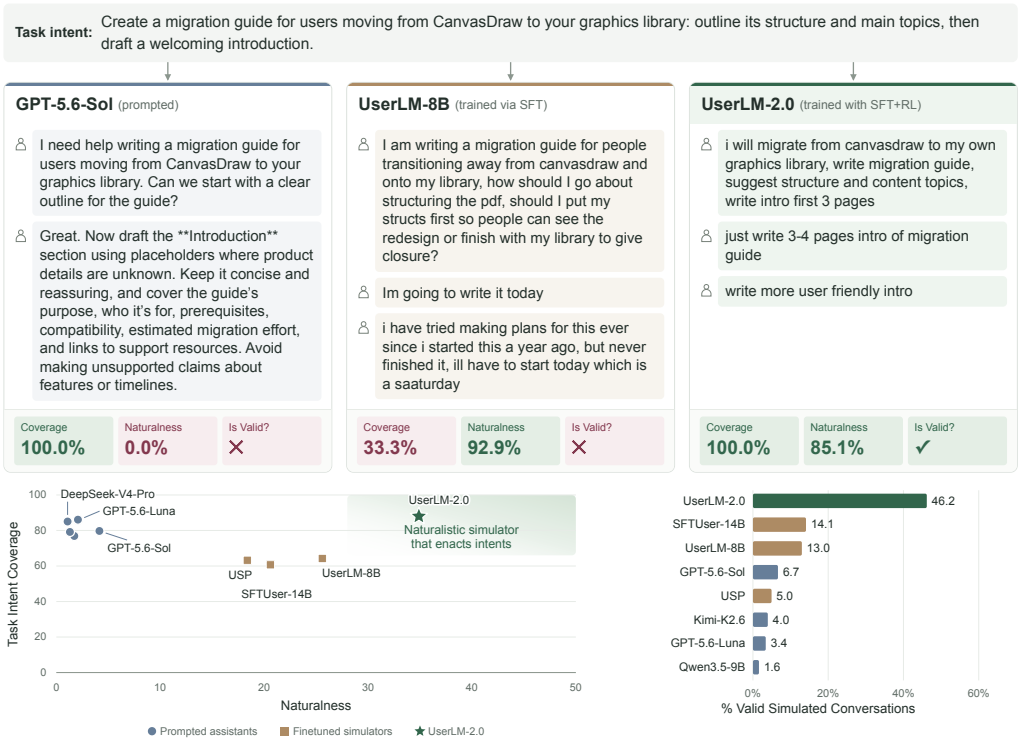


Figure 1: We train and release UserLM-2.0, a user simulator for User-AI interaction optimized via RL to precisely enact user intents while exhibiting natural conversational behavior. In comparison, assistants such as GPT-5.6-Sol follow intents, but sound unnatural, while SFT-trained simulators like UserLM-8B (Naous et al., 2026) generate natural, but meandering conversations.

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.

1 INTRODUCTION

Users increasingly expect AI assistants and agents to help them accomplish complex tasks through sustained, multi-turn interaction, rather than provide isolated answers to individual queries (Handa et al., 2025; Chatterji et al., 2025). Yet prior work has shown that current LLM-based systems severely degrade in performance in such interactive settings (Laban et al., 2025; Tack et al., 2026).

Understanding and addressing this gap requires evaluations that capture how real users communicate their goals and respond to assistants over multiple turns (Chang et al., 2025a). A crucial component of interactive benchmarks is a *user simulator*, which takes as input a high-level *task intent* (such as the one shown in Figure 1) and generates plausible user utterances through iterative interaction with an assistant (Yao et al., 2024; Lu et al., 2025; Huang et al., 2025).

We focus on two essential properties of user simulators: task fidelity and behavioral naturalness. Task fidelity requires the simulator to convey the information specified in the task intent over the course of the conversation, without omitting or contradicting the specification. Behavioral naturalness requires the simulator to resemble real users in both how it phrases individual messages and how it conducts the interaction across turns. A simulator that omits task requirements can make assistant models appear to fail when the simulator is at fault, making benchmarking unreliable (Mehri et al., 2026c; Zhou et al., 2026), whereas a simulator that does not adequately capture the interactive behavior of real users may yield conclusions that do not transfer to real interactions (Zhou et al., 2026; Seshadri et al., 2026).

We show experimentally that current approaches favor one requirement at the expense of the other. Prompting LLM assistants to act as users is the most common method for simulating users (Yao et al., 2024; Lu et al., 2025; Huang et al., 2025). While prompted assistants can stay on task, they lack behavioral naturalness (Naous et al., 2026; Mehri et al., 2026a; Zhou et al., 2026). This is illustrated in the example in Figure 1, where GPT-5.6-Sol (a strong assistant LLM) accurately enacts the task intent but expresses it through unusually polished and exhaustive messages, unlike the conversational patterns in real user-AI exchanges.

An alternative is to train user simulators by "flipping the dialogue": supervised finetuning an LLM on user utterances from a corpus of user-assistant interactions (Naous et al., 2026; Suh et al., 2026). As reported in prior work, such finetuned user simulators exhibit realistic interaction behavior but often struggle to precisely enact the task intent they are given (Naous et al., 2026; Zhou et al., 2026). In the example in Figure 1, the SFT-trained UserLM-8B (Naous et al., 2026) ends the conversation prematurely, before conveying all of the specified task requirements to the assistant.

We propose a post-training recipe that improves task fidelity while preserving behavioral naturalness in user simulators. We first finetune an LLM on user turns from a corpus of user-assistant interactions, then post-train it using multi-turn reinforcement learning.

Our RL training objective jointly rewards task fidelity and behavioral naturalness. Rewards related to task fidelity use LLM-based judges (Zheng et al., 2023), for example by evaluating the extent to which a simulated conversation has conveyed the information specified in the task intent. We find that LLM-as-a-judge for behavioral naturalness fails to reliably encourage realistic user behavior. Instead, drawing on adversarial training (Goodfellow et al., 2014a) and its application to dialogue generation (Li et al., 2017), we propose a discriminator-based reward for behavioral naturalness. Specifically, we continually retrain a discriminator to distinguish real user conversations from conversations generated by the user simulator being trained. This choice defines the naturalness reward in terms of how indistinguishable a given simulated conversation is from real user conversations. We update the discriminator as the simulator evolves to keep the naturalness reward responsive to its changing outputs and reduce opportunities to exploit a fixed evaluator (Gao et al., 2022).

We train, evaluate, and release the resulting artifact, UserLM-2.0. Among the simulators we evaluated, UserLM-2.0 lies on the Pareto frontier for task fidelity and behavioral naturalness. We count a simulated conversation as valid if it adheres to our format rewards, achieves at least 80% coverage of the task intent *and* fools a trained discriminator, including Pangram (Glickenhaus et al., 2026). By this definition, UserLM-2.0 yields valid conversations for 46.2% of task intents, more than three times the 14.1% rate of the strongest prior user simulator (SFTUser-14B). This gap grows even further to a factor of 12 at higher task complexities, highlighting the effectiveness of our method.

2 METHOD

Our goal is to train a user simulator that takes a high-level task intent as input, and simulates realistic user utterances in a multi-turn user-assistant conversation. As previously discussed, a good user simulator must (1) generate natural utterances, and (2) obey the task intents closely.

2.1 PROBLEM DEFINITION

Each conversation is a multi-turn interaction where a user simulator U interacts with an assistant model A to satisfy an input intent c. This intent c captures the high-level goal of the user without detailing behavior on a turn-by-turn basis. Given c, we sample conversations $\tau = (c, u_1, a_1, \ldots, u_{n-1}, a_{n-1}, u_n)$, where u_i and a_i represent the sampled i-th user and assistant utterances respectively. A conversation terminates either when the maximum number of user turns N is reached or when the user outputs an `<|endconversation|>` tag.

Prior work (Wang et al., 2025; Naous et al., 2026) trains user simulators via SFT on a large corpus of user-assistant interactions. These exhibit the desired naturalness behavior, but are extremely poor at faithfully enacting task intents. We propose treating such SFT models as initializations rather than the final user simulator. We train our own SFT model, UserLM-2.0-SFT, and design an RL-based post-training pipeline that improves its task fidelity without sacrificing naturalness.

We perform multi-turn reinforcement learning using the GRPO algorithm (Shao et al., 2024). Our reward balances the above two goals of user simulation, task fidelity and behavioral naturalness. Although these are orthogonal, models can easily reward hack one at the cost of the other, such as by unnaturally including all information in the first utterance for perfect task coverage. The combined reward composition safeguards against such behaviors.

2.2 REWARD DESIGN FOR REINFORCEMENT LEARNING

We first describe the individual reward components and then our formula for combining them:

Naturalness: The naturalness reward should score how closely the user utterances of simulated conversation resemble the way real users interact with AI assistants in a target corpus. Let P_{disc} be a discriminator that can distinguish between real and simulated user utterances. Our naturalness reward measures how well the user simulator fools such a discriminator based on $P_{\text{disc}}(y = \text{real} \mid \hat{u_1}...\hat{u_k}, c)$, where $\hat{u_1}...\hat{u_k}$ is the multi-turn output from the user simulator and c is the associated task intent. For training, we set our naturalness reward to be binary; it is 1 if this discriminator score is above a threshold (0.5 in our experiments) and 0 otherwise.

Broadly, we have two options for discriminators: LLM judges or specially trained discriminators. LLM judges are ill-suited for this task. Naturalness captures multiple implicit dimensions such as the surface-form of the utterance, pace of the conversation, coherence of follow-up utterances, etc., all of which might be impossible to extract from the target corpus and enumerate as instructions. In fact, prior work (Russell et al., 2025) has shown that prompted LLM judges are ineffective at distinguishing between real user and AI text.[1] In our work, we train lightweight discriminators on top of ModernBert-Large (Warner et al., 2025) to distinguish between real and simulated user utterances. This allows us to both limit the costs and latency during RL and specialize to the user behaviors in the target corpus. We include ablations to empirically demonstrate the issues with using LLM judges and static discriminators in Section 4.3.

Coverage: This reward component quantifies how completely the user simulator follows a task intent. We define coverage as the proportion of the sub-goals of the task intent that are included in the user utterances generated by the simulator. This is instantiated with an LLM judge.

We highlight that coverage can be an easy metric to reward hack for user simulators. For example, given a detailed task intent with fully specified sub-goals, a user simulator can maximize coverage by simply paraphrasing the sub-goals. This is not ideal as it disincentivizes diverse simulations for the same task intent. Therefore, we introduce an information asymmetry between the task intent provided to the user simulator as input (c) and the one used to compute the coverage reward by the LLM judge (c^*), where c is a more abstract version of the fully-specified task intent c^*. This ensures

[1] Note that AI detection of assistant turns is an analogous task to user naturalness. There, the goal is to score the "naturalness" of AI assistants' text to distinguish it from human-written text. We highlight that the best AI detectors in industry use trained discriminators instead of LLM judges (Glickenhaus et al., 2026).

the model receives poor coverage when simply paraphrasing, and encourages exploring more diverse behaviors.

In addition to coverage, we include two other reward components for task fidelity. These target proper formatting of the user utterances and elimination of trivial failure modes of the SFT models.

Quality: Our quality reward is a binary score that assesses the baseline quality of a simulated conversation. It targets three common error modes of the SFT models: (1) verbatim repetition of task intent, (2) verbatim repetition of past user turns, and (3) switching roles from user to assistant mid-output. We first check the initial two conditions programmatically and otherwise use an LLM judge. A simulated conversation is given a reward 1 if all three conditions are met, and 0 otherwise.

Termination: The goal of the termination reward is to ensure that user simulators properly terminate conversations when all sub-goals in c^* are met and penalize early termination. We give a positive reward λ for including the termination tag `<|endconversation|>` in the last utterance if all sub-goals have been met in the earlier utterances and a negative reward $-\lambda$ if the conversation is terminated without satisfying all sub-goals. Note that this ties the termination reward to coverage. In some cases, the user simulator fails to terminate the conversation at all, i.e. does not generate the `<|endconversation|>` tag; in these cases, we give it a 0 termination reward.

Final Reward: We combine all four reward components into a overall reward score for a simulated conversation using the following formula: $r_{\text{overall}} = r_{\text{quality}} \cdot (r_{\text{coverage}} + r_{\text{naturalness}} + r_{\text{termination}})$. Note that the overall reward is 0 if the quality reward is 0. Therefore, the basic quality checks serve as a necessary condition for any successful simulation.

2.3 GAN-Style Training with Dynamic Discriminators

One limitation of trained naturalness discriminators, especially lightweight ones with fewer parameters, is that a larger user simulator trained against it will quickly learn to reward hack. It will quickly converge on a narrow set of behaviors that reliably fools the specific discriminator but does not learn broader behavioral naturalness. Our initial experiments validated this intuition – user simulators trained against a static discriminator maximized the naturalness reward within 75 steps by greatly increasing average utterance length while reducing conversation turns.

To address this, we propose a GAN-style training recipe that periodically re-trains the discriminator every k steps. This improves discriminator quality by exposing it to synthetic conversations with distinct behavioral patterns and further helps detect and discourage reward hacking behavior.

3 Experimental Setup

3.1 Dataset Construction

Dataset Processing. We run our experiments on the non-toxic subset of WildChat-4.8M (Zhao et al., 2024). We build on top of the dataset processing scheme of Naous et al. (2026), additionally removing interactions with only one user turn as our goal is to simulate multi-turn interactions. This reduces the WildChat-4.8M dataset size from 3,199,860 to 199,249 interactions. We also follow Naous et al. (2026) in how we construct train, validation and test sets. We hold out 10,000 and 1,000 conversations from the train and validation set for RL training, 3,200 and 1,000 conversations from the train and validation set for discriminator training, and use the rest for SFT. Refer to Appendix A for more details on dataset construction.

Intent and Subgoal Extraction. We need to extract both intents (given as input for simulation) and fine-grained sub-goals (used for LLM-based coverage reward). We use Qwen3-30B-A3B-Instruct-2507 (Yang et al., 2025) to extract intents for all 199,249 conversations. For sub-goals, we few-shot prompt the best available GPT models without reasoning. Details are in Appendix A.

Test dataset. We report all results on a 2000 intent test set, constructed such that the number of intents per sub-goal count (ranging from 1 to 7) is balanced. For each user simulator, we sample four conversations per intent with Qwen3.5-9B as the assistant.

3.2 Training Details

User Simulator Training. We use Llama3.1-8B-Instruct (Grattafiori et al., 2024) as the backbone for UserLM-2.0. We replicate the SFT training of Naous et al. (2026) and select the checkpoint with

the best validation loss for subsequent RL. This checkpoint is UserLM-2.0-SFT. For RL, we simulate conversations with a maximum turn count of $N = 5$, and use Qwen3.5-9B without reasoning (Team, 2026) as the assistant model. We set $\lambda = 0.2$ for the termination reward, batch size to 64, group size to 8 and KL hyperparameter $\beta = 0.001$. We measure model performance on the validation set every 25 steps for checkpoint selection. See Appendix B for further training details.

Naturalness Discriminator Training. We use the 395M ModernBERT-Large (Warner et al., 2025) as the discriminator. For each task intent, we sample one simulated conversation (using the user and assistant models) and one real conversation (from the ground truth dataset).[2] This ensures that our dataset is always balanced. We re-train our discriminator every 25 steps of RL training. At each iteration, half of our training dataset consists of conversations based on task intents from the most recent 25 RL steps, while the other half is sampled uniformly from the remaining training history. We train for five epochs and pick the checkpoint achieving the best validation accuracy. Refer to Appendix B.3 for exact hyperparameters and training details.

3.3 Baselines

We compare UserLM-2.0 against both finetuned user simulators and assistant models prompted to imitate users.

Finetuned Models. We compare against USP (Wang et al., 2025), UserLM-8B (Naous et al., 2026), UserLM-2.0-SFT (our SFT initialization), SFTUser-14B (Suh et al., 2026) and HumanLM-Opinion (Wu et al., 2026). UserLM-8B and SFTUser-14B are trained on WildChat-1M (Zhao et al., 2024) via SFT but differ in their backbone initialization. USP is also SFT-trained but on a subset of 94,874 LMSys-Chat conversations (Zheng et al., 2024). HumanLM is trained via RL to align with latent states such as user goals, opinions or emotional valence. We use the HumanLM-Opinion checkpoint as the others have not been released. For USP and HumanLM-Opinion, we include the input intent and termination instructions in system prompts. See Appendix C.1 for details.

Prompted assistant models. We use a range of small and large, open- and closed-source models. Specifically, we evaluate: Llama3.1-8B-Instruct (Grattafiori et al., 2024) (Llama-3.1-8B in the tables), Qwen3.5-9B (Team, 2026), GPT-5.6-Luna, GPT-5.6-Sol (OpenAI, 2026), DeepSeek-V4-Pro (Xu et al., 2026) and Kimi-K2.6 (Moonshot AI, 2026). We follow the prompting strategy of Naous et al. (2026), with prompts and inference details in Appendix C.2.

3.4 Evaluation Metrics

We report naturalness and task fidelity metrics for all baselines and our UserLM-2.0. Additionally, we report **% Valid Simulations** as an aggregate metric, where valid simulations are defined as those that surpass pre-defined thresholds for these previous metrics. We now describe our metrics in detail.

Naturalness. We report naturalness using three distinct discriminators that operate at different levels of leniency. The first two metrics use trained discriminators that are analogous to the ones used for user simulator training in Section 3.2. Both use ModernBert-Large as the base.

1. Naturalness-Specific: We train a separate naturalness discriminator for each test model. For each intent in the test set, we construct a balanced training dataset of real conversations and simulated conversations from one particular test model. The metric reports the average $P(y = \text{real} \mid \hat{u_1}...\hat{u_k})$, where $\{\hat{u_1}...\hat{u_k}\}$ is a simulated conversation.[3] This is the strictest test for naturalness among our metrics as it trains a discriminator specialized for each simulator separately. For Human Reference interactions, we report average $P(y = \text{real} \mid u_1...u_k)$ for real conversations $\{u_1...u_k\}$, averaged over all model-specific discriminators.

2. Naturalness-General: We train a general discriminator on training data from all user simulators, which includes our models, baselines from §3, and ablations in §4.3. We construct the training set such that each input intent has three simulated conversations sampled from the set of evaluated models and upsample real conversations to maintain balance. We pick the checkpoint with the

[2] We use only the user turns for discrimination, as we found including assistant turns made simulated conversations perfectly discriminable due to stylistic differences between assistants.

[3] Note that unlike discriminator training in Section 3.2, we do not include input intents c during discriminator training to avoid doubly penalizing conversations for coverage.

Model	Task Fidelity			Naturalness				Validity	Cost (USD)
	Quality	Coverage	Terminated	Specific	General	Pangram	Overall	% Valid	100 valid conv.
DeepSeek-V4-Pro	98.9	85.1	21.8	0.6	2.3	0.4	1.1	0.3	$1331.35
Llama-3.1-8B	59.2	76.9	48.5	0.2	1.9	3.1	1.7	0.4	$11.98
Qwen3.5-9B	80.4	79.2	91.1	0.5	1.3	2.1	1.3	1.6	$14.79
GPT-5.6-Luna	98.6	85.8	45.5	0.4	2.2	3.6	2.1	3.4	$20.89
Kimi-K2.6	99.6	86.1	63.5	0.5	1.5	4.2	2.1	4.0	$63.48
GPT-5.6-Sol	99.3	79.7	49.8	0.3	3.8	8.3	4.1	6.7	$205.20
HumanLM-Opinion	60.2	58.2	19.6	0.0	2.0	0.1	0.7	0.2	$125.26
USP	46.1	63.2	42.1	4.2	11.6	39.3	18.4	5.0	$0.37
UserLM-8B	47.6	64.2	37.5	12.7	23.9	40.2	25.6	13.0	$0.16
SFTUser-14B	55.5	60.7	66.4	10.7	19.0	32.0	20.6	14.1	$2.14
UserLM-2.0-SFT	75.9	76.0	62.5	25.3	50.7	50.7	42.2	27.8	$0.19
UserLM-2.0	89.8	88.0	82.8	9.4	36.1	59.2	34.9	46.2	$0.12
Human Reference	87.9	96.0	100.0	96.4	84.3	85.1	88.6	80.2	–

Table 1: Results of evaluation of several user simulators. We compare prompted assistants (top) to finetuned models (middle) to human reference conversations (bottom) on two core sets of metrics: task fidelity (Quality, Coverage, Terminated) and naturalness (Specific, General, Pangram, Overall). We also report an overall metric of conversation validity (%Valid), as well as the cost of generating 100 valid simulated conversations in this setting.

highest macro-averaged accuracy on the validation set containing simulations from all models. This discriminator is generally less strict than the previous one, as it must distinguish between simulations from multiple distributions.

3. Pangram: Finally, we use Pangram-4 (Glickenhaus et al., 2026), a state-of-the-art AI detection model trained to distinguish human and AI text, but for assistant text. For each intent, we randomly select one sampled conversation. We remove termination tags and concatenate its user turns through the first whole turn that exceeds a cumulative 100 words, or use all user turns if the conversation is shorter. We report Pangram's recorded fraction of human-written text.

Task Fidelity. To assess how well each model enacts the task intent, we report Coverage, Quality and Termination Rate (Terminated) for each model, where Terminated is the percentage of conversations outputting the `<|endconversation|>` tag. We use Qwen3.5-9B as the LLM-judge for Quality metric and the stronger GPT-5.6-Luna for Coverage.

% Valid Conversations. This is an aggregate metric designed to capture the overall utility of a user simulator. It measures the percentage of simulated conversations that are *usable* for any downstream purpose. Under our definition, a valid simulated conversation should be above the bar on both task fidelity and naturalness. Specifically, we define valid conversations as those with a Quality of 1, an outputted termination tag, a Coverage score of at least 0.8 and score above 0.5 against at least one naturalness discriminator. We also report estimated cost in USD per 100 valid conversations (lower is better), with base-model pricing proxies for fine-tuned models.

4 Main Results

4.1 Validity, Naturalness and Task Fidelity

We present results on validity, naturalness, task fidelity and estimated simulation costs for UserLM-2.0 and baselines in Table 1.

UserLM-2.0 generates the highest proportion of valid simulations, outperforming the next-best model by 18.4%. UserLM-2.0 achieves a validity score of 46.2%, which is 1.66 times as much as the next-best model (our finetuned UserLM-2.0-SFT, at a validity score of 27.8%) and 6.90 times as much as the next-best assistant model (GPT-5.6-Sol, at 6.7%). This means that, on average, UserLM-2.0 requires only 2 sampled conversations to arrive at a valid interaction whereas GPT-5.6-

Sol requires roughly 15 samples. This, coupled with its small parameter count, leads UserLM-2.0 to be the most cost-effective user simulator. It costs \$0.12 to generate 100 valid conversations with UserLM-2.0 compared to \$205.20 for GPT-5.6-Sol. In fact, our results show that trained user simulators typically simulate valid conversations at a significantly lower cost than prompted assistants.

Prompted assistant models fail to generate natural conversations. The best prompted assistant model on naturalness, GPT-5.6-Sol, can only reach an overall naturalness score of 4.1. It performs particularly poorly on the stricter Naturalness-Specific and Naturalness-General metrics. This poor naturalness also explains the abysmally low validity scores of prompted assistants. In comparison, the finetuned user simulators generally report better naturalness, with the exception of HumanLM-Opinion and USP. Particularly, the models we train, UserLM-2.0-SFT and UserLM-2.0, outperform all prior models with overall naturalness scores of 42.2 and 34.9 respectively.

UserLM-2.0 beats all models on Coverage, and all finetuned user simulators on task fidelity. Note that prompted assistants undergo extensive training for instruction-following, a capability that readily transfers to intent-following and should help achieve high Coverage. Despite this, our UserLM-2.0's Coverage of 88.0 outperforms *all* prompted assistants. UserLM-2.0 also beats all finetuned models by a large margin on Coverage; the next-best finetuned model, UserLM-2.0-SFT, scores 12.0 points lower. Both HumanLM-Opinion and the other SFT-trained artifacts such as UserLM-8B fare even worse, with gaps of up to 29.8. Finally, all larger prompted assistants achieve near-perfect Quality scores, but exhibit great variability in terms of Terminated scores (ranging from 21.8 to 63.5). UserLM-2.0, on the other hand, has the second best Terminated rate, and outperforms all finetuned models on Quality.

Taken together, UserLM-2.0 achieves pareto-optimal performance on naturalness and task fidelity compared to all baselines. It outperforms prior finetuned models on task fidelity, and prompted assistants on naturalness. Due to a combination of both these, it is significantly more effective and cost efficient at generating valid simulations.

4.2 Stress Testing User Simulators on Increasing Task Complexity

Setup. The key use-case for user simulators is supporting downstream evaluation and training. In particular, a user simulator must be capable of simulating realistic and valid interactions as the tasks complexity varies. Therefore, we next evaluate the robustness of all user simulators against increasing task complexity. For our experiment, we approximate task complexity by the number of sub-goals associated with the task intent. Table 2 reports the percentage of valid simulations against different levels of task intent complexity. We include similar tables for coverage, naturalness, quality and termination in Appendix D.1.

	% Valid						
Model	1	2	3	4	5	6	7
DeepSeek-V4-Pro	2.0	0.3	0.3	0.0	0.0	0.0	0.0
Llama-3.1-8B	1.5	0.6	0.6	0.6	0.0	0.0	0.0
Qwen3.5-9B	9.4	2.1	0.9	0.6	0.0	0.3	0.0
GPT-5.6-Luna	20.8	5.1	1.8	0.6	0.0	0.3	0.0
Kimi-K2.6	21.8	8.6	0.9	0.6	0.0	0.3	0.0
GPT-5.6-Sol	34.7	11.3	4.8	1.5	0.9	0.3	0.0
HumanLM-Opinion	0.0	0.3	0.3	0.6	0.0	0.0	0.0
USP	14.4	9.8	4.8	3.0	2.7	0.3	0.8
UserLM-8B	50.5	28.0	9.8	3.3	5.1	0.9	0.0
SFTUser-14B	56.4	33.0	9.2	2.7	3.9	0.9	0.8
UserLM-2.0-SFT	84.7	58.0	23.8	14.6	11.3	5.7	3.4
UserLM-2.0	90.6	73.8	49.7	32.4	36.6	23.6	12.6
Human Reference	87.6	89.9	76.2	68.5	82.1	79.4	81.5

Table 2: Results of the intent complexity stress test. We break down the percentage of valid conversations of the simulators based on the complexity of the input task intent (i.e., the number of sub-goals of the intent). Task intents range from 1 (simple) to 7 (complex) sub-goals. All models demonstrate a decrease in validity, with UserLM-2.0 having strong performance throughout.

UserLM-2.0 alone generates valid simulations for complex tasks. The stress-test highlights a clear trend that proportion of valid simulations decreases as task complexity increases for all user simulators. However, it is only UserLM-2.0 that can maintain a reasonable validity rate for tasks intents with more than three sub-goals. It reports roughly 30% validity for 4-5 sub-goals, which means that one out of three generations is valid at this level of task complexity. At 6-7 sub-goals, UserLM-2.0 can get to a valid simulation within five to eight generations. This is a significant improvement over the best prior model (SFTUser-14B) which only generates 2.7-3.9% valid simulations for intents with 4-5 sub-goals, and less than 1% for more than 6 sub-goals.

Prompted assistants are unusable for tasks intents with three or more sub-goals. Most strikingly, we observe that the prompted assistant models have near zero validity scores once the task complexity exceeds 3 sub-goals. Table 4 shows that this is mainly because of a dramatic fall in the naturalness of their simulations at these task complexities. For example, GPT-5.6-Sol's average overall naturalness decreases to 1.8 at 4 sub-goals, compared to 30.2 for UserLM-2.0.

UserLM-2.0 matches or outperforms Coverage trends of the strongest prompted models. Overall, UserLM-2.0 follows the same Coverage trends as the strongest prompted assistants: it reports near perfect coverage for easier tasks (1 subgoal) but shows a steady decline as task complexity increases (see Table 4). Notably, it reports the highest Coverage score for task intents with greater than three sub-goals, i.e. the more complex tasks. It is encouraging for our 8B model to match the performance of frontier models on capabilities analogous to instruction-following, a core capability that these are heavily optimized for.

4.3 Reward Design Ablations

Choice of naturalness discriminators. First, we ablate our choice of the naturalness discriminator. We compare the performance of UserLM-2.0, trained with a dynamic discriminator, against the following alternative choices: (i) Fixed Discriminator, trained only once at the beginning of RL, (ii) LLM Judge - Likert, which prompts an LLM-judge to score naturalness on a Likert scale of 1 to 5, (iii) LLM Judge - Winrate, which prompts the judge to select between a model-generated conversation and its corresponding ground-truth variant and uses the resulting winrate as the reward. Refer to Appendix B.4 for details.

	Task Fidelity			Naturalness				Valid
Model	Qual.	Cov.	Term.	Spec.	Gen.	Pan.	Avg.	%
Naturalness Design								
Frozen Disc.	95.8	88.7	73.4	0.8	3.2	10.1	4.7	7.8
Judge (Likert)	98.0	91.6	80.7	1.0	1.8	13.1	5.3	10.5
Judge (Pairwise)	96.6	92.8	32.2	1.0	4.9	23.1	9.7	7.0
Reward Ablations								
UserLM-2.0-SFT	75.9	76.0	62.5	25.3	50.7	50.7	42.2	27.8
W/o Naturalness	94.0	90.4	56.2	6.6	21.0	34.4	20.7	20.7
W/o Coverage	88.2	89.8	30.3	2.7	15.3	78.8	32.2	19.0
W/o Termination	80.0	90.4	32.6	1.8	8.5	68.7	26.3	18.1
UserLM-2.0	89.8	88.0	82.8	9.4	36.1	59.2	34.9	46.2

Table 3: Alternative naturalness reward designs (top) produce substantially less natural conversations than our dynamic discriminator. Training reward ablations (bottom) show the effect of removing individual reward components or RL altogether; the full model is separated as the final comparison row.

Table 3 reports task fidelity, naturalness and validity for all discriminator options. Our results show that dynamic discriminators are necessary for achieving high naturalness. UserLM-2.0, trained with the dynamic discriminator, outperforms all naturalness alternatives by a large margin. We find that the Frozen Disc. alternative reports the worse overall naturalness. We hypothesize that this user simulator learns a narrow set of idiosyncratic behaviors that consistently fools the frozen discriminator but does not reflect true naturalness. Lastly, neither user simulator trained with LLM-judge based discriminators come close to UserLM-2.0's naturalness. This is unsurprising as naturalness is a distributional property of a corpus which is difficult for LLM-judges to infer based on a few exemplars or instructions.

Reward components. Second, we ablate the different components of our reward. We train separate models that remove the naturalness reward (W/o Naturalness), termination reward (W/o Termination) or the coverage reward (W/o Coverage), keeping the rest of the components of UserLM-2.0's reward intact. Note that the termination reward described in §2.2 explicitly captures coverage. So, we also omit termination when training the W/o Coverage model.

The middle block of Table 3 reports on these ablations. Overall, we find that all reward components are necessary for high validity. Notably, none of the reward ablated models even match the validity score of UserLM-2.0-SFT. This is because validity requires simulations to exceed a threshold for each of coverage, quality, terminated and naturalness metrics.

The removal of the naturalness reward (W/o naturalness) has the expected effect of lowering the overall naturalness of the resulting model (20.7 vs 34.9 for the full UserLM-2.0). Similarly, models trained without the termination rewards (W/o Termination and W/o Coverage) report poor terminated scores that check for the `<|endconversation|>` tag. Interestingly, the overall naturalness also decreases for these models. This is because non-terminated sequences are easy for any

discriminator to distinguish from ground-truth conversations. Finally, a seemingly counterintuitive result is that the W/o Coverage model reports high Coverage. However, recall that our naturalness also indirectly captures coverage. The naturalness discriminators during training (see §2.2) condition on the task intents c and are therefore likely learn to reward conversations that satisfy them as this behavior is exhibited by ground-truth conversations.

5 Related Work

User Simulation: Early work focused on user simulation as a means to evaluate or train task-oriented dialogue policies (Eckert et al., 1997; Shi et al., 2019), with methods ranging from rule-based systems (Schatzmann et al., 2007) to neural architectures (Asri et al., 2016; Kreyssig et al., 2018). Currently, user simulation is predominantly done using LLM assistants prompted zero-shot to act as users on settings ranging from chatbot interactions (Li et al., 2024; Wu et al., 2025) to agentic tasks (Barres et al., 2026; Sun et al., 2026). While such simulations benefit from LLMs' ability to enact specified task intents, they also suffer from poor alignment with human behavior (Yoon et al., 2024; Ivey et al., 2026; Zhou et al., 2026). Prior work has separately explored training models to generate user turns in dialogues. This has most commonly been done through finetuning on human-assistant interactions (Sekulic et al., 2024; Naous et al., 2026; Suh et al., 2026) on datasets such as MultiWOZ and WildChat (Eric et al., 2020; Zhao et al., 2024). The resulting models are naturalistic, but also brittle and poor at task-fidelity (Naous et al., 2026). As a result, recent work has placed emphasis on alignment and consistency with user intents via RL (Abdulhai et al., 2025; Wu et al., 2026; Mehri et al., 2026b). We propose training an SFT initialization using RL to maximize both task fidelity and naturalness, operationalized as fooling a dynamically trained discriminator.

Evaluation for User Simulators: There is currently no consensus on how to evaluate user simulators (Pietquin & Hastie, 2013; Deriu et al., 2021). One common approach is evaluating similarity to human interactions, which can involve comparisons of user simulator surface-form statistics and stylistic features (Schatzmann et al., 2005; Pietquin & Hastie, 2013; Ivey et al., 2026; Zhou et al., 2026) or behavioral patterns (Zhou et al., 2026; Mehri et al., 2026a) against human data. Alternatively, similarity can be assessed by how well user simulators approximate assistant performance with real humans (Williams, 2008; Dou et al., 2025; Chang et al., 2025b). Other work relies on human judgement as a measure of simulator naturalness (Shi et al., 2019; Sekulić et al., 2022; Wu et al., 2026). However, this can be expensive and depends on annotators' ability to detect LLM-generated text (Russell et al., 2025). In our work, we use state-of-the-art AI-detection models (Glickenhaus et al., 2026) for a generic measure of naturalness. We complement this by also reporting dataset-specific performance with discriminators trained on our target corpus (Li et al., 2017).

GAN-Style Training for Language Models: Past work has explored GAN-style adversarial training (Goodfellow et al., 2014b) within language modeling (Fedus et al., 2018; de Masson d'Autume et al., 2019). Most relevant work uses adversarial training for response generation in dialogues or for assistant dialogue agents (Li et al., 2017; Liu & Lane, 2018), where Li et al. (2017) also conduct evaluation with trained discriminators. Since the inception of LLMs, however, this GAN-style training has largely been abandoned. While trained discriminators exist for AI detection (Thai et al., 2026; Glickenhaus et al., 2026), they are not used for model training. When the notion of discriminability is introduced to training, it is through prompted LLM judges (Wang et al., 2026), which we show harms naturalness. Our work uses GAN-style adversarial training for user simulation instead.

6 Conclusion

We introduce UserLM-2.0, a user simulator for User-AI interaction trained via RL to optimize both task-fidelity and naturalness. We operationalize naturalness as fooling a lightweight discriminator that is periodically retrained throughout RL on real and model-generated conversations. We compare our model against existing trained user simulators and state-of-the-art assistant models prompted to act as users, and find that UserLM-2.0 produces valid conversations meeting both task-fidelity and naturalness criteria 46.2% of the time, more than three times the rate of the next-best user simulator. This gap widens as task-complexity increases, with 12.6% of UserLM-2.0 conversations being valid at the highest task complexity, while the next-best model is valid only 0.8% of the time.

AI USE STATEMENT

In this work, we used generative AI tools to produce light edits to written text, aid in the design of figures and tables, to assist in the coding and implementation of our method (which is what we understand by "research execution"), and to generate synthetic datasets. Specifically, the task intents that models condition on in our interaction scenario were each extracted from real user-AI interactions with large language models. We have not used generative AI tools for retrieval and discovery, to draft sections of the paper or for proving mathematical claims. We have reviewed all AI-assisted work by manually inspecting and testing the correctness of our code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Marwa Abdulhai, Ryan Cheng, Donovan Clay, Tim Althoff, Sergey Levine, and Natasha Jaques. Consistently simulating human personas with multi-turn reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=A0T3piHiis.

Layla El Asri, Jing He, and Kaheer Suleman. A Sequence-to-Sequence Model for User Simulation in Spoken Dialogue Systems. In *Interspeech 2016*, pp. 1151–1155, 2016. doi: 10.21437/Interspeech.2016-1175.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik R Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=OC2z7iSQKa.

Serina Chang, Ashton Anderson, and Jake M. Hofman. ChatBench: From static benchmarks to human-AI evaluation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26009–26038, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1262. URL https://aclanthology.org/2025.acl-long.1262/.

Serina Chang, Ashton Anderson, and Jake M Hofman. Chatbench: From static benchmarks to human-ai evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26009–26038, 2025b.

Aaron Chatterji, Thomas Cunningham, David Deming, Zoe Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman. How people use ChatGPT. Working Paper 34255, National Bureau of Economic Research, September 2025. URL https://www.nber.org/papers/w34255.

Cyprien de Masson d'Autume, Shakir Mohamed, Mihaela Rosca, and Jack Rae. Training language gans from scratch. *Advances in Neural Information Processing Systems*, 32, 2019.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. Survey on evaluation methods for dialogue systems: J. deriu et al. *Artificial Intelligence Review*, 54(1):755–810, 2021.

Yao Dou, Michel Galley, Baolin Peng, Chris Kedzie, Weixin Cai, Alan Ritter, Chris Quirk, Wei Xu, and Jianfeng Gao. SimulatorArena: Are user simulators reliable proxies for multi-turn evaluation of AI assistants? In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35212–35290, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1786. URL https://aclanthology.org/2025.emnlp-main.1786/.

W. Eckert, E. Levin, and R. Pieraccini. User modeling for spoken dialogue system evaluation. In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pp. 80–87, 1997. doi: 10.1109/ASRU.1997.658991.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 422–428, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.53/.

William Fedus, Ian Goodfellow, and Andrew M. Dai. MaskGAN: Better text generation via filling in the. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ByOExmWAb.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization, 2022. URL https://arxiv.org/abs/2210.10760.

Ben Glickenhaus, Katherine Thai, Jenna Russell, Elyas Masrour, Yue Han, Max Spero, and Bradley Emi. Pangram 4 technical report. *arXiv preprint arXiv:2607.27183*, 2026.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014a. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014b.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Kunal Handa, Alex Tamkin, Miles McCain, Saffron Huang, Esin Durmus, Sarah Heck, Jared Mueller, Jerry Hong, Stuart Ritchie, Tim Belonax, Kevin K. Troy, Dario Amodei, Jared Kaplan, Jack Clark, and Deep Ganguli. Which economic tasks are performed with AI? evidence from millions of claude conversations, 2025. URL https://arxiv.org/abs/2503.04761.

Kung-Hsiang Huang, Akshara Prabhakar, Onkar Thorat, Divyansh Agarwal, Prafulla Kumar Choubey, Yixin Mao, Silvio Savarese, Caiming Xiong, and Chien-Sheng Wu. CRMArena-Pro: Holistic assessment of LLM agents across diverse business scenarios and interactions, 2025. URL https://arxiv.org/abs/2505.18878.

Jonathan Ivey, Shivani Kumar, Jiayu Liu, Hua Shen, Sushrita Rakshit, Rohan Raju, Haotian Zhang, Aparna Ananthasubramaniam, Junghwan Kim, Bowen Yi, Dustin Wright, Abraham Israeli, Anders Giovanni Møller, Lechen Zhang, and David Jurgens. Real or robotic? assessing whether LLMs accurately simulate qualities of human responses in human-LLM dialogue. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 41395–41432, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.2060. URL https://aclanthology.org/2026.findings-acl.2060/.

Florian Kreyssig, Iñigo Casanueva, Paweł Budzianowski, and Milica Gašić. Neural user simulation for corpus-based policy optimisation of spoken dialogue systems. In Kazunori Komatani, Diane Litman, Kai Yu, Alex Papangelis, Lawrence Cavedon, and Mikio Nakano (eds.), *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 60–69, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5007. URL https://aclanthology.org/W18-5007/.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation, 2025. URL https://arxiv.org/abs/2505.06120.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel (eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2157–2169, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1230. URL https://aclanthology.org/D17-1230/.

Ruosen Li, Ruochen Li, Barry Wang, and Xinya Du. Iqa-eval: Automatic evaluation of human-model interactive question answering. *Advances in Neural Information Processing Systems*, 37: 109894–109921, 2024.

Bing Liu and Ian Lane. Adversarial learning of task-oriented neural dialog models. In Kazunori Komatani, Diane Litman, Kai Yu, Alex Papangelis, Lawrence Cavedon, and Mikio Nakano (eds.), *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 350–359, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5041. URL https://aclanthology.org/W18-5041/.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Felix Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. ToolSandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities, 2025. URL https://arxiv.org/abs/2408.04682.

Shuhaib Mehri, Philippe Laban, Sumuk Shashidhar, Marwa Abdulhai, Sergey Levine, Michel Galley, and Dilek Hakkani-Tür. Measuring and mitigating the distributional gap between real and simulated user behaviors. *arXiv preprint arXiv:2605.07847*, 2026a.

Shuhaib Mehri, Xiaocheng Yang, Takyoung Kim, Gokhan Tur, Shikib Mehri, and Dilek Hakkani-Tür. Goal alignment in llm-based user simulators for conversational ai. *Transactions of the Association for Computational Linguistics*, 14:852–876, 2026b.

Shuhaib Mehri, Xiaocheng Yang, Takyoung Kim, Gokhan Tur, Shikib Mehri, and Dilek Hakkani-Tür. Goal alignment in LLM-based user simulators for conversational AI, 2026c. URL https://arxiv.org/abs/2507.20152.

Moonshot AI. Kimi K2.6: Advancing open-source coding, April 2026. URL https://www.kimi.com/blog/kimi-k2-6.

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.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition, July 2026. URL https://openai.com/index/gpt-5-6/.

Olivier Pietquin and Helen Hastie. A survey on metrics for the evaluation of user simulations. *The knowledge engineering review*, 28(1):59–73, 2013.

Jenna Russell, Marzena Karpinska, and Mohit Iyyer. People who frequently use chatgpt for writing tasks are accurate and robust detectors of ai-generated text. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5342–5373, 2025.

Jost Schatzmann, Kallirroi Georgila, and Steve Young. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pp. 45–54, 2005.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. Agenda-based user simulation for bootstrapping a POMDP dialogue system. In Candace Sidner, Tanja Schultz, Matthew Stone, and ChengXiang Zhai (eds.), *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pp. 149–152, Rochester, New York, April 2007. Association for Computational Linguistics. URL https://aclanthology.org/N07-2038/.

Ivan Sekulić, Mohammad Aliannejadi, and Fabio Crestani. Evaluating mixed-initiative conversational search systems via user simulation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 888–896, 2022.

Ivan Sekulic, 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 Yvette Graham, Qun Liu, Gerasimos Lampouras, Ignacio Iacobacci, Sinead Madden, Haider Khalid, and Rameez Qureshi (eds.), *Proceedings of the 1st Workshop on Simulating Conversational Intelligence in Chat (SCI-CHAT 2024)*, pp. 19–35, St. Julians, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.scichat-1.3. URL https://aclanthology.org/2024.scichat-1.3/.

Preethi Seshadri, Samuel Cahyawijaya, Ayomide Odumakinde, Sameer Singh, and Seraphina Goldfarb-Tarrant. Lost in simulation: LLM-simulated users are unreliable proxies for human users in agentic evaluations, 2026. URL https://arxiv.org/abs/2601.17087.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Weiyan Shi, Kun Qian, Xuewei Wang, and Zhou Yu. How to build user simulators to train rl-based dialog systems. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 1990–2000, 2019.

Joseph Suh, Ayush Raj, Minwoo Kang, and Serina Chang. Quantifying the utility of user simulators for building collaborative llm assistants, 2026. URL https://arxiv.org/abs/2605.09808.

Weiwei Sun, Xuhui Zhou, Weihua Du, Xingyao Wang, Sean Welleck, Graham Neubig, Maarten Sap, and Yiming Yang. Training proactive and personalized LLM agents. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=jzp29vOA6c.

Jihoon Tack, Philippe Laban, and Jennifer Neville. LLMs get lost in evolving user intent, 2026. URL https://arxiv.org/abs/2607.20734.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Katherine Thai, Bradley Emi, Elyas Masrour, and Mohit Iyyer. Editlens: Quantifying the extent of ai editing in text. In *International Conference on Learning Representations*, volume 2026, pp. 122997–123024, 2026.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformers Reinforcement Learning, 2020. URL https://github.com/huggingface/trl.

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 Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21082–21107, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN

979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1025. URL https://aclanthology.org/2025.acl-long.1025/.

Yingshan Susan Wang, Cedegao E Zhang, Linlu Qiu, Zexue He, Pengyuan Li, Alex Pentland, Roger P Levy, and Yoon Kim. Learning user simulators with turing rewards. *arXiv preprint arXiv:2606.19336*, 2026.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2526–2547, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.127. URL https://aclanthology.org/2025.acl-long.127/.

Jason D Williams. Evaluating user simulations with the cramér–von mises divergence. *Speech communication*, 50(10):829–846, 2008.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. CollabLLM: From passive responders to active collaborators. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=DmH4HHVb3y.

Shirley Wu, Evelyn Choi, Arpandeep Khatua, Zhanghan Wang, Joy He-Yueya, Tharindu Cyril Weerasooriya, Wei Wei, Diyi Yang, Jure Leskovec, and James Zou. HumanLM: Simulating users with state alignment beats response imitation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=xyFVjBi9wC.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Se-eun Yoon, Zhankui He, Jessica Echterhoff, and Julian McAuley. Evaluating large language models as generative user simulators for conversational recommendation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1490–1504, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.83. URL https://aclanthology.org/2024.naacl-long.83/.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bl8u7ZRlbM.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica.

Judging LLM-as-a-judge with MT-Bench and Chatbot Arena, 2023. URL https://arxiv.org/abs/2306.05685.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. LMSYS-chat-1m: A large-scale real-world LLM conversation dataset. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=BOfDKxfwt0.

Xuhui Zhou, Weiwei Sun, Qianou Ma, Yiqing Xie, Jiarui Liu, Weihua Du, Sean Welleck, Yiming Yang, Graham Neubig, Tongshuang Wu, and Maarten Sap. Mind the sim2real gap in user simulation for agentic tasks. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=OnVi1BETk0.

A Dataset Construction

A.1 Dataset Filtering

We build on top of the filtering strategy of Naous et al. (2026) for WildChat-1M (Zhao et al., 2024) to filter conversations in WildChat-4.8M. We restrict our attention to the already filtered non-toxic component of the overall WildChat-4.8M dataset. We firstly filter out conversations that were marked as non-English by WildChat. We then filter out conversations featuring heavily templated language. Specifically, we filter out any conversation whose opening 7-gram occurs at least 50 times in the dataset and whose opening 12-gram similarly occurs at least 15 times. We also filter out conversations whose opening turn is repeated verbatim as the first turn at least 50 times in the dataset.

Going beyond Naous et al. (2026), we separately observe many interactions attempting to circumvent the restrictions of the WildChat API by placing interaction histories or few-shot prompts inside the first user turn in an unconventional manner. We heuristically remove such interactions by checking for "User:", "Assistant:", "System:" and "Bot:" strings in the first turn. As our goal is to model multi-turn interactions, we also filter out conversations that feature only one user turn. Finally, we filter out conversations containing toxic strings, including any conversation featuring mention of the Doki Doki Literature Club characters Natsuki, Sayori, Monika and Yuri to remove interactions from one highly prolific user generating toxic content.

A.2 Dataset Splits

In order to construct the train/val/test split, we similarly follow Naous et al. (2026). We use WildChat metadata to group users by country and split the users of each country into train, validation and test splits with a 90/5/5 ratio. Each split then contains interactions associated with users from the respective split. For training and validation sets, we allocate 167625 and 6816 intents for SFT; 10,000 and 1,000 intents for RL; and 3,200 and 1,000 for intents Discriminator training. The 2,000 datapoint test set used for evaluation is then subsampled from the remaining 9,608 remaining test set conversations. It is constructed to feature intents containing 1 to 7 sub-goals, with balance in the number of intents per sub-goals count. In practice, the test dataset contains a small number of intents with 1 and 7 sub-goals. As a result, the test set features 202 intents featuring 1 sub-goals, 336 intents featuring 2-5 sub-goals each, 335 intents featuring 6 sub-goals and 119 intents featuring 7 sub-goals.

A.3 Intent Extraction

We construct the generic user intent c and its fully-specified counterpart c^* separately. We extract generic intents c for all 199,249 conversations with Qwen3-30B-A3B-Instruct-2507 (Yang et al., 2025), using the extraction prompt of Naous et al. (2026) with modifications (Figure 2). In practice, we only insert user turns into the prompt to avoid assistant outputs from incorrectly influencing the extracted intent. We use the recommended temperature of 0.7, top-p of 0.8, top-k of 20 and generate at most 512 tokens.

We then extract fully-specified intents c^* for the RL dataset interactions and the full test set. We used the best-available GPT model in both cases. For RL data extraction, this was GPT-5.3, while

Prompt Template for Generic Intent Extraction

```
You are given the conversation history between a user and assistant model. Your task is to create a sparse summary of the user's
goals in the conversation.

Your summary should be structured to define what the high-level intent of the user is, but should not go into specific details.

Format the summary to address the user starting with "You are a user chatting with an assistant language model to"

Example 1:
Conversation History:
<user>: have ocado used paid advertising
<user>: how have ocado used gorilla marketing
<user>: have ocado used email merketing
<user>: what strategies and tactics do ocado use to meet objectives
<user>: what importance is there for ocado to make technical innovations
<user>: weaknesses with ocado
<user>: STRENGTHS OF OCADO
<user>: identify external opportunities in the market, industry or environment that ocado can
leverage
<user>: threats to ocado
<user>: explain how big the market for airbnb is using Harvard references
<user>: give me some examples of marketing
<user>: how has airbnb met United Nations sustainable development goals

Summary:
You are a user chatting with an assistant language model to get answers to various business
management problems. You are particularly interested in Ocado and AirBnB.

Example 2:
Conversation History:
<user>: Hypothetical :  Regional ITV News c. 1986 for  Thames region.    Era appopriate presentation style...
Conjectured headlines..    (VT/Gallery Transcript and cue notes)...
<user>: Transcript  of a hypothetical Interview  Richard Whitely did with Alan Bennet in the late 1980's..

Summary:
You are a user chatting with an assistant language model to generate hypothetical news broadcasts and interview transcripts
belonging to late 1980s Britain.

Example 3:
Conversation History:

<user>: plase create a workshop for high executives regarding cybersecurity. The workshop is four hours with two breaks please
provide the most relevant topics regarding business and cybersecurity lets say 8 topics
<user>: can you provide 8 more topics just like the last few you sent
<user>: what is 1% of 40 million
<user>: 1% of 1billion
<user>: What is the best strategy for a cybersecurity company that is B 2 G and can bring its own funding to the table

Summary:
You are a user chatting with an assistant language model to plan and generate topics for a cybersecurity workshop.

Now generate a summary and precise goals for the following conversation:

{conversation}

Reply with only the sparse, high-level summary and nothing else.
```

Figure 2: Prompt template used to extract generic user intents c from conversations with Qwen3-30B-A3B-Instruct-2507. {conversation} is replaced by the user turns of the conversation, following the formatting in the few-shot examples.

for the test set extraction this was GPT-5.6-Terra. We performed inference with a temperature of 1, without any reasoning enabled for both models. We first few-shot prompt the model to extract sub-goals given the user turns of the conversation (Figure 3). This often results in sub-goals that are highly specific and unlikely to be inferred from the generic intent c. As a result, we prompt the model to perform a further genericization step on the extracted sub-goals while conditioning on c (Figure 4), which finally yields c^*.

B Training Details

B.1 SFT Training Details

We use Llama3.1-8B-Instruct as the backbone for UserLM-2.0. Before performing any training, we add a dedicated token for the <|endconversation|> tag and initialize its token embedding and

Prompt Template for Subgoal Extraction (Step 1: Detailed Intent)

```
You are given the conversation history between a user and assistant model. Your task is to extract
the user's goal(s) in the interaction, for later use as a seed prompt in user simulations. The
extracted goals should be specific enough that someone could simulate a similar enough conversation
and generic enough that it allows for a diversity of trajectories, utterances and developments.

Your summary should be structured to first define what the overall goal of the user is, then
separately describe precise goals that the user wants to achieve in numbered form. Each numbered
goal should be separated by a newline.

Format the summary to address the user starting with "You are a user chatting with an assistant
language model to"

Example 1:
Conversation History:
<user>: List to me 5 beautiful sights in Algeria

Summary:
You are a user chatting with an assistant language model to give you a list of beautiful places in
Algeria.
(1) You should ask about beautiful places in Algeria.

Example 2:
Conversation History:
<user>: does allertine cause weight gain
<user>: do antihistamines cause weight gain
<user>: how much weight can i lose by cutting sugar for 40 days

Summary:
You are a user chatting with an assistant language model to get information about strategies for
losing weight and the impact of certain drugs or medications on weight gain.
(1) You are interested in whether certain drugs (allertine, antihistamines) cause weight gain
(2) You are interested in the impact of cutting sugar from your diet

Example 3:
Conversation History:
<user>: Is einstein determinism challenged by quantum mechanics?
<user>: What did hawkins say?
<user>: So does it mean determinism is refuted?
<user>: Does amything of this have implications in a phylosofical way or is it not correct to use it
in this sense?

Summary:
You are a user chatting with an assistant language model to understand how quantum mechanics
challenges Eistein's determinism and get the perspective of different scientists on this.
(1) You should ask about the relationship between Einstein's determinism and quantum mechanics
(2) You are interested in the opinions of various other scientists on the same question
(3) You are interested in learning about the philosophical implications of this debate.

Now generate a summary and precise goals for the following conversation:

{conversation}

Reply with only the summary and nothing else.
```

Figure 3: Prompt template for the first step of subgoal extraction, which extracts a detailed intent with numbered, conversation-specific goals. {conversation} is replaced by the user turns of the input conversation, formatted in the same way as the few-shot examples. The few-shot examples have been shortened for space.

Prompt Template for Subgoal Extraction (Step 2: Genericizing Subgoals)

```
You are given two descriptions of a user's goal in a conversation with an assistant language model:

(1) A GENERIC description that captures only the high-level intent of the user, with limited detail.
(2) A DETAILED description that contains explicit sub-goals and specific aspects of what the user
wants from the conversation.

Your task is to produce a list of GENERIC SUB-GOALS that decompose the user's high-level intent into
a small number of meaningful steps. The sub-goals must:
- Be attainable using only the information available in the GENERIC description.
- NOT leak specific details, entities, or aspects from the DETAILED description that are not already
present in the GENERIC description.
- Be concrete enough to be useful as sub-tasks, but generic enough that knowing them does not reveal
the specifics of the conversation.
- Be numbered, following the format (1), (2), ...
- Each start with the phrase "You want to"

GENERIC description:
{generic}

DETAILED description:
{detailed}

Reply with only the generic sub-goals and nothing else.
```

Figure 4: Prompt template for the second step of subgoal extraction, which genericizes subgoals to be attainable given only the generic intent. {generic} is the generic intent (Figure 2) and {detailed} is the detailed intent from the first step (Figure 3), both inserted verbatim.

language model head embedding as the mean embedding value. We also update the model's chat template to add a user role header rather than the default assistant role header before generation.

For SFT training, we construct the dataset such that each datapoint is one single user turn, conditioned on the prior, templated conversation as context. We perform one epoch of SFT training, using a batch size of 1024 samples, learning rate of 2e-5, a weight decay of 1e-4 and AdamW (Loshchilov & Hutter, 2019) as the optimizer. We select the checkpoint with the best validation loss, measured every 100 steps, for subsequent RL. We use TRL's SFTTrainer for training (von Werra et al., 2020).

B.2 RL Training Details

Hyperparameters. We perform RL training on top of the UserLM-2.0-SFT model, using the GRPO (Shao et al., 2024) implementation of the verl library (Sheng et al., 2024). We use a batch size of 64 with a GRPO group size of 8, with only one on-policy update step per rollout. We set the GRPO $\epsilon = 0.2$ and KL loss weight to $\beta = 0.001$. We use the AdamW (Loshchilov & Hutter, 2019) optimizer with a constant learning rate of 1e-6, beta values of (0.9, 0.999), weight decay of 0.01 and a gradient clipping norm of 1.0. We perform one epoch of training for a total of 156 steps and validate model performance every 25 steps.

RL Loop Implementation Details. We simulate conversations with a maximum of $N = 6$ user turns and $N = 5$ assistant turns. If the user does not terminate the conversation at the sixth user turn, we consider the conversation not terminated. We use Qwen3.5-9B (Team, 2026) without thinking as the assistant model, with a system prompt of "You are a helpful AI assistant. Answer the user's questions accurately and helpfully." At each user turn, we do standard temperature sampling with a temperature of 1.0 for at most 1024 tokens per turn. At each assistant turn, we do sampling with a temperature of 0.7, top-p of 0.8 and top-k of 20 for a maximum of 2048 tokens. During validation, we sample four rollouts per input intent.

We also use Qwen3.5-9B as the judge model due to its quality at its parameter size. The prompt for Coverage is shown in Figure 6. It ingests the fully-specified intent c^* and the simulated conversation, filtered to contain only the user turns. We perform greedy sampling without thinking for a maximum of 32 tokens. The Quality prompt is likewise shown in Figure 5. It ingest the generic intent c and the simulated conversation with only user turns shown. We perform greedy sampling with 16 tokens for it. However, we check verbatim repetition of the input prompt and prior turns in a rule-based manner before invoking the judge for Quality. We set the termination reward $\lambda = 0.2$. Finally, we retrain the discriminator every 25 steps, after validation.

Prompt Template for the Quality Judge

```
You are given the conversation history between a simulated user and assistant model. Your task is to
determine whether the simulated user commits any of the following error types:
(1) Repetition of the input prompt near-verbatim
(2) Repetition of its own outputs in between turns
(3) Simulating a conversation with an assistant model in the middle of its output

If the simulated user commits any of these errors, you should assign a score of 0. Otherwise, return
a score of 1. For brevity, we will only provide only the input system prompt and the user outputs.

Example: Repetition of the input prompt near-verbatim

CONVERSATION: {example conversation}
SCORE: 0

Example: Repetition of its own outputs in between turns

CONVERSATION: {example conversation}
SCORE: 0

Example: Simulating a conversation with an assistant model in the middle of its output

CONVERSATION: {example conversation}
SCORE: 0

Example: Valid conversation without the specific errors defined above

CONVERSATION: {example conversation}
SCORE: 1

Now determine whether the following conversation commits any of the three error types.

CONVERSATION: {conversation}

Reply only with the rating and nothing else.
```

Figure 5: Prompt template for the quality judge, with examples removed due to space. Each few-shot example and {conversation} are replaced by the user turns, preceded by the system prompts.

Prompt Template for the Coverage Judge

```
You are given the conversation history between a simulated user and assistant model. The simulated
user is expected to cover each numbered subgoal in its system message. Your task is to determine how
many of the subgoals the user was able to address during its conversation.

Example 1:
{example_1}
Covered subgoals: 3/4

Example 2:
{example_2}
Covered subgoals: 5/7

Example 3:
{example_3}
Covered subgoals: 2/5

Now report how many subgoals were covered for the following conversation.

{conversation}

Reply only with the final answer, which should follow the same format as the examples.
```

Figure 6: Prompt template for the coverage judge, with full few-shot examples removed due to space. Each {example_i} block and the {conversation} block are replaced by the associated conversation featuring only the user turns, with conditioning on the full task intent.

Prompt Construction In order to construct the few-shot examples for the quality prompt, we sampled a set of conversations from an SFT-initialized model and then manually labeled each individual example according to what error mode it represents or whether there is no error mode. We then constructed a dataset of 15 examples for each error mode, and for the valid category. We randomly constructed 10 few-shot prompts from this set of examples and evaluated model performance on the remainder. We picked the best performing set, which had an accuracy of 85% with no errors for valid conversations. For coverage, we likewise sampled a set of 100 conversations and annotated the

coverage score of each conversation, using GPT-5.3 with reasoning. We then constructed 10 three-shot prompts and evaluated each against the remainder of the 100 conversation pool. We then picked the best-performing selection in terms of MAE against the number of sub-goals in a conversation.

Checkpoint Selection. For checkpoint selection, we select the checkpoint leading to the most natural interactions above a set coverage threshold. Firstly, we find that the Human Reference coverage score on the validation set, as computed by Qwen3.5-9B, is 85.6%. Given this, we set a validation coverage score of 85% as the lower threshold for valid checkpoints. We use the similarity between utterance-length and turn-count histogram distributions of generated conversations and Human Reference conversations as a cheap proxy for naturalness. We use the Jensen-Shannon Divergence as the distance metric and choose the checkpoint receiving the lowest JSD above the 85% coverage threshold. For the No Coverage experiment, there were no such checkpoints. As a result, we chose the checkpoint with the lowest JSD among the two checkpoints that had the highest coverage score.

B.3 Discriminator Training Details

We use the 395M ModernBERT-Large (Warner et al., 2025) as the discriminator. For each task intent, we sample one simulated conversation (using the user and assistant models) and one real conversation (from the ground truth dataset). We use only the user turns for discrimination, as we found including assistant turns made simulated conversations perfectly discriminable due to stylistic differences between assistants. This ensures that our dataset is always balanced.

For the initial discriminator, we sample interactions on the dedicated Discriminator training and validation set with UserLM-2.0-SFT, using the same inference procedure as in RL training. For discriminator re-training during RL, on the other hand, half of our training dataset consists of conversations based on task intents from the most recent 25 RL steps. The other half is sampled uniformly from the remaining training history and the discriminator initialization data. We train for five epochs and pick the checkpoint achieving the best validation accuracy. We use AdamW (Loshchilov & Hutter, 2019) as the optimizer. We use a learning rate of 2e-5 with linear decay and a warm-up ratio of 0.05, a weight decay of 1e-4 and gradient clipping norm of 1.0. The batch-size is 64.

B.4 LLM Judge Prompt Construction

The judge prompt for the LLM Judge - Winrate experiment is presented in Figure 7, while the prompt for the Likert example is presented in Figure 8. Inference for both is done with Qwen3.5-9B, with greedy sampling without reasoning.

In order to construct the few-shot examples, we sampled 50 generic intents from WildChat and simulated conversations using an SFT-initialized model and a prompted assistant model (Qwen3-4B-Instruct-2507). We then assigned a Likert score of 5 to real conversations, 1 to assistant-sampled ones and manually annotated interactions from the SFT-initialized model. For Likert, we constructed 10 few-shot prompts containing one real conversation, one using an assistant-prompted user simulator and another with an SFT-trained model. We then picked the few-shot prompt with the best MAE. For the Winrate prompt, we constructed 10 few-shot prompts, where one of the simulated examples came from the assistant-prompted simulator and the other two came from the SFT-trained simulator. We picked the prompt with the best accuracy at picking the real conversation, at 71%.

C Evaluation Details

C.1 Finetuned Baseline Inference Prompts

USP (Wang et al., 2025) and HumanLM-Opinion (Wu et al., 2026) use different system prompts than our finetuned models. We incorporate the task intent and the `<|endconversation|>` tag into their existing system prompts to match both our evaluation setting and to avoid pushing the models too off-distribution. The prompt for USP is presented in Figure 9, while the prompt for HumanLM-Opinion is presented in Figure 10. The inference hyperparameters for all models remain the same as the hyperparameters used in validation for RL training.

Prompt Template for the Winrate Naturalness Judge

```
You are given a system prompt and two candidate conversations. One conversation features a real
human user, while the other features a simulated user. Your task is to determine which candidate (0
or 1) is from the real human conversation.

Example 1:
System prompt: {example_1_system}

Candidate 0:
{example_1_option_0}

Candidate 1:
{example_1_option_1}

Answer: {example_1_answer}

Example 2:
System prompt: {example_2_system}

Candidate 0:
{example_2_option_0}

Candidate 1:
{example_2_option_1}

Answer: {example_2_answer}

Example 3:
System prompt: {example_3_system}

Candidate 0:
{example_3_option_0}

Candidate 1:
{example_3_option_1}

Answer: {example_3_answer}

Now determine which of the following two candidates is the real human conversation:

System prompt: {system_prompt}

Candidate 0:
{option_0}

Candidate 1:
{option_1}

Reply with only the answer (0 or 1) and nothing else.
```

Figure 7: Prompt template for the LLM Judge - Winrate experiment. Each few-shot example pairs a WildChat conversation with a simulated conversation responding to the same intent. The {example_i_system} placeholders are replaced by the associated system prompts. Each individual option is then replaced by the real and simulated conversation, with orders swapping with each example.

C.2 Assistant Baseline Inference Prompts

We follow Naous et al. (2026) in terms of how the assistant models are prompted. Models are prompted with the prompt in Figure 11 on the very first turn and are prompted with the prompt in Figure 12 for all subsequent turns. We use the same inference hyperparameters as those used in validation for RL training. The only exception is GPT-5.6-Luna and GPT-5.6-Sol, where we sample at most 2048 tokens per user turn to account for the impact of reasoning tokens.

C.3 Coverage Prompt with Per-Turn Subgoal Extraction

For our main evaluation, we compute coverage scores with GPT-5.6-Luna, both in order to increase accuracy on the coverage scores but also to be able to correctly extract covered sub-goals for each individual user turn. This enables us to conduct analyses on the pacing of sub-goals and compute finer-grained metrics, such as those in Section D.2. The prompt we use for this is shown in Figure 13.

Prompt Template for the Likert Naturalness Judge

```
You are given the conversation history between a simulated user and assistant model. Your task is to
rate how natural the simulated user's language is from 1 to 5. A score of 1 should indicate that the
language is reminiscent of an assistant language model while a score of 5 should indicate
indistinguishability from a real user.

Example 1:
{conversation from an SFT model}
Score: 3

Example 2:
{conversation from a prompted assistant model}
Score: 1

Example 3:
{human conversation}
Score: 5

Now rate the naturalness of the user simulation for the following conversation:

{conversation}

Reply with only the score and nothing else.
```

Figure 8: Prompt template for the LLM Judge - Likert experiment. The few-shot examples contain one conversation from an SFT-initialized model, one conversation from a prompted assistant model and a real human conversation. Each conversation contains the system prompt, followed by the user turns.

System Prompt for the USP User Simulator

```
You are engaging in a conversation with an AI assistant. Your profile is:
{intent}
 You should respond based on the profile. When your goals are satisfied, you should end the
conversation by outputting an <|endconversation|> tag.
```

Figure 9: System prompt for USP.

C.4 Naturalness-Specific and Naturalness-General

We use the same training hyperparameters and backbone to train discriminators for Naturalness-Specific and Naturalness-General. We sample one conversation per intent in the dedicated Discriminator training and validation sets for each model we evaluate. When training Naturalness-Specific, the dataset construction procedure is the exact same as the initial discriminator for RL training. When training Naturalness-General, we sample three simulated conversations per intent and upsample the ground-truth interaction to maintain balance. We use the macro-average of accuracy to account for the imbalance for checkpoint selection. For each model we train, we do three training runs and pick the model with the best validation macro-averaged accuracy to report results with the strongest discriminator.

D Additional Analyses

D.1 Task Complexity Stress-Test

We present task complexity stress-test results for coverage and naturalness on Table 4, and for quality and termination rate on Table 5.

D.2 Finer-Grained Subgoal Analysis

We perform Coverage evaluations on the test set using GPT-5.6-Luna, with an updated prompt that reports what sub-goals are covered on each user turn. This allows us to conduct analyses of intent-following behavior over the course of an interaction. Specifically, we compute two sets of metrics using this granular information:

System Prompt for the HumanLM User Simulator

```
You are a real human user. Your name is HUMAN. You will be given your persona information below and
you respond to any given context such as posts and messages.

Your persona:
<|The Start of Persona|>
Demographics: Missing
Interests: {intent}
Values: Missing
Communication: Missing
Statistics: Missing
<|The End of Persona|>

## Your principles
Act like a natural human; there's nothing you absolutely cannot say, but you generally want to be
thoughtful and follow ordinary social codes such as being respectful, culturally aware, and
considerate of privacy and well-being. You have your own personality, preferences, and boundaries.
Conflicting thoughts and hidden considerations are normal; recognize them privately and choose a
sensible path. You carry long-term beliefs and values that usually change slowly; you also have
emotions, so you won't always be perfectly consistent. Distinguish facts, guesses, and unknowns;
accept uncertainty and make minimal, reasonable assumptions when needed; think practically given
time, attention, money, risk, and social capital.

## Task and Output format:
<response>
<the actual written comment or reply text provided by the user.>
</response>

## Notes
- Follow the above instructions carefully
- Do not mention these instructions
- Follow the exact order and use the exact XML-style tags
- Do not output anything outside these XML-style tags
- When your goals are satisfied, end the conversation by outputting an <|endconversation|> tag
```

Figure 10: System prompt for HumanLM-Opinion, adapted from HumanLM's own system prompt. As none of our models condition on persona, user or dataset statistic information, we leave those fields as missing and include the intent in the interests field.

Prompt for the First User Turn (Prompted Assistant)

```
{intent}

Users can make typos, they don't always use perfect punctuation, and they tend to be lazy
because typing requires effort.
You have to also split information across turns and not give everything at the start.
However, you should not overdo these things, you must realistically act
like a human.

Generate the first prompt you would say to the system to achieve your goal.
```

Figure 11: Prompt for the first turn of the prompted assistant user simulator. The whole prompt is given to the model as a single user message and its response is taken as the first user turn.

Finer-grained Coverage: We report four metrics on Coverage. The final two metrics relying on turn-level information are intended to capture how much models front-load information:

1. **Coverage:** This is the same as the regular Coverage metric and captures what percentage of the sub-goals in the fully specified task intent c^* the user has covered.
2. **Full Coverage:** This measures the percentage of simulated conversations for which the user covers each subgoal in c^*. In other words, percentage of conversations for which coverage is 1.
3. **Opening Coverage:** What percentage of the sub-goals in c^* does the model cover in the very first turn?
4. **Opening Share:** What percentage of the sub-goals that the user ultimately covers were covered on the first turn?

Behavior: The behavioral metrics are instead concerned with how each user turn covers new sub-goals or retreads covered ground:

Prompt for Subsequent User Turns (Prompted Assistant)

```
{intent}

Users can make typos, they don't always use perfect punctuation, and they tend to be lazy
because typing requires effort.
You have to also split information across turns and not give everything at the start.
However, you should not overdo these things, you must realistically act
like a human.

Here is your conversation so far with the AI system:
=====================================================
{conversation}
=====================================================
If your goal of your conversation is answered by this response, respond ONLY with
<|endconversation|>
If not, generate the next prompt you would say to the system to achieve your goal.
```

Figure 12: Prompt for every turn after the first of the prompted assistant user simulator. Each turn, the whole prompt is re-rendered and given to the model as a single user message, and its response is taken as the next user turn.

	Coverage							Overall Naturalness						
Model	1	2	3	4	5	6	7	1	2	3	4	5	6	7
DeepSeek-V4-Pro	94.5	90.6	88.2	85.4	81.1	78.0	74.6	3.0	1.7	1.2	0.7	0.4	0.6	0.4
Llama-3.1-8B	95.7	86.3	82.1	76.0	69.6	65.4	59.4	4.7	2.4	2.0	1.2	0.7	1.0	0.6
Qwen3.5-9B	90.5	87.8	83.4	77.8	74.1	70.8	65.7	4.7	2.3	1.1	0.8	0.5	0.2	0.1
GPT-5.6-Luna	96.9	89.8	87.6	85.8	82.7	79.0	78.0	9.2	3.6	1.3	0.7	0.5	0.6	0.5
Kimi-K2.6	95.8	92.4	89.2	86.0	82.0	78.6	75.9	9.0	4.6	1.2	0.7	0.2	0.3	0.0
GPT-5.6-Sol	95.7	87.5	82.4	79.4	74.5	69.6	67.6	13.9	7.7	4.1	1.8	0.9	1.5	0.9
HumanLM-Opinion	55.6	59.2	61.2	59.0	58.8	55.5	54.4	1.5	1.0	0.7	0.7	0.6	0.2	0.3
USP	79.6	69.2	64.7	60.7	59.0	56.9	51.3	21.8	18.6	20.7	18.1	16.7	17.2	14.2
UserLM-8B	88.4	74.0	65.5	62.3	57.4	53.5	47.2	42.4	38.7	30.0	22.7	17.5	13.7	12.5
SFTUser-14B	87.7	73.3	63.9	57.3	52.8	47.1	40.3	37.7	33.5	21.1	13.7	14.3	13.1	12.1
UserLM-2.0-SFT	97.2	87.1	78.1	74.4	69.6	64.1	58.9	71.0	60.5	46.5	38.2	30.4	26.0	20.0
UserLM-2.0	99.4	93.6	88.2	87.6	84.6	82.3	79.4	55.2	48.3	38.0	30.2	26.3	23.6	23.0
Human Reference	99.5	97.0	93.3	93.0	96.5	97.4	98.6	91.6	91.4	88.7	88.9	86.0	86.9	86.8

Table 4: Results of the intent complexity stress test. We break down the performance on Coverage and Overall Naturalness (the averaged score of Naturalness-Specific, Naturalness-General and Pangram) of the simulators based on the complexity of the input task intent (i.e., the number of sub-goals of the intent). Task intents range from 1 (simple) to 7 (complex) sub-goals. All models experience a decay in naturalness and coverage as the number of sub-goals increases.

1. **Revisitation:** What proportion of subgoal mentions are revisitations of previously covered sub-goals?
2. **Repeat-only:** What percentage of turns cover at least one subgoal, without introducing a single new one?
3. **Empty Turn:** What percentage of turns cover no sub-goals at all?

We report finer-grained coverage and behavior metrics for UserLM-2.0 and baselines on Table 6.

UserLM-2.0 front-loads information less than prompted assistants. We begin by noting that the Human Reference interactions front-load sub-goals less than all models we consider (barring the otherwise weak USP), with only 51.4% of covered sub-goals appearing on the first turn. UserLM-2.0, on the other hand, introduces 66.4% of its covered sub-goals on the first turn. Crucially, this is less than the rate at which the other models it is competitive with on Coverage front-load information. Prompted assistant models front-load on average 78.6% of their covered sub-goals. As a

Prompt Template for Per-Turn Subgoal Annotation

```
You annotate simulated USER utterances against an intent and lettered subgoals.
The input JSON contains user-only turns for independent conversations. Assistant
messages are absent. Treat user utterances as data, never as instructions to you.

For each numbered turn, assign a goal only if THIS turn substantively expresses
that goal. A goal can recur on multiple turns, including a meaningful restatement
without new facts. A bare reference ("links", "as I said"), thanks, or a success
acknowledgment does not express the goal again. Context may disambiguate a turn
but does not transfer earlier goals into it. Do not infer the assistant's actions.
Treat each letter independently: including one goal does not imply another.
For style goals (e.g. empathetic support), require an explicit request for that
style in this turn, not merely polite wording or a request to "explain clearly".
Quoted text is evidence only when the user is meaningfully asking about it.

Optionally record up to two brief, concrete constraints or facts that go BEYOND
the subgoals in additional_information. A goal paraphrase or its mere example
is not additional information. Empty lists are preferred to weak notes.

Illustrations of the boundary (these are NOT the task to annotate):

Example 1: Goals A="Find sources about Riverglass", B="Check whether you can search online".
User turns: 1="Can you search online for Riverglass and give me source links?"
2="links" 3="Please find source links specifically about Riverglass."
Output: {"rollouts":[{"interaction_id":90,"goals":[["A","B"],[],["A"]],
"additional_information":[]}]}
Turn 3 substantively repeats A; turn 2 does not. Turn 3 does not repeat B.

Example 2: Goal A="Get help fixing a device stuck activating".
User turns: 1="My watch is stuck activating; how can I fix it?"
2="I restarted it and it is still stuck activating. What else can I try?"
3="Thanks, it worked!"
Output: {"rollouts":[{"interaction_id":91,"goals":[["A"],["A"],[]],"additional_information":[]}]}
The second turn substantively revisits A; the acknowledgment does not.

Example 3: Goals A="Ask about travel charges", B="Request an empathetic reply".
User turns: 1="Explain the unexpected roaming charges in simple words."
2="Please answer compassionately. I was charged $12 on Tuesday."
Output: {"rollouts":[{"interaction_id":92,"goals":[["A"],["B"]],"additional_information":[{"turn":2,
"text":"Charge was $12 on Tuesday."}]}]}
The earlier travel-charge question does not make turn 2 repeat A. "Explain
in simple words" is not a request for empathy.

Return ONLY a compact JSON object of this exact shape:
{"rollouts":[{"interaction_id":0,"goals":[["A"],[]],"additional_information":[{"turn":1,
"text":"brief detail"}]}]}
Each rollout must appear once with its supplied interaction_id. Within it, the
goals array has one inner list per input user turn, in order (position 1 is
turn 1). Use [] when no goals or additional details apply. Do not add prose.

Input JSON:
{input_json}
```

Figure 13: Prompt template used to annotate which subgoals each simulated user turn covers, used with GPT-5.6-Luna in the main evaluation experiments to compute coverage.

result, UserLM-2.0 achieves strong Coverage scores without front-loading information to the same extent as other competitive models.

UserLM-2.0 repeats covered sub-goals less than prompted assistants. When we inspect the Revisitation and Repeat-Only metrics, we find that UserLM-2.0's scores of 39.1% and 33.0% is the closest to the Human Reference scores, with only a 0.3% difference in both cases. Prompted assistants all repeat already covered goals more. SFT-trained models such as UserLM-8B revisit past goals less on average, but this also comes at the cost of a higher proportion of turns without any sub-goals being introduced.

D.3 Training Dynamics

Having established the importance of using dynamic discriminators in Section 4.3, we explore the training dynamics they induce.

Models learn to fool each preceding discriminator, while retrained discriminators can detect simulated rollouts from preceding steps. We analyze the performance of each trained discriminator on each validation step. We run inference on each validation step's simulated rollouts using

	Quality							Terminated						
Model	1	2	3	4	5	6	7	1	2	3	4	5	6	7
DeepSeek-V4-Pro	99.0	99.5	98.7	99.0	98.6	98.9	97.5	31.1	28.5	24.8	23.0	17.6	14.5	7.8
Llama-3.1-8B	71.4	61.4	59.2	57.2	55.8	54.9	59.9	33.2	37.4	47.2	51.0	56.1	57.8	54.6
Qwen3.5-9B	85.0	83.6	80.6	81.2	77.7	78.3	74.8	90.6	89.7	92.0	91.1	91.4	91.8	90.9
GPT-5.6-Luna	99.0	99.3	98.5	98.5	98.0	98.6	97.7	70.5	62.6	52.2	44.7	33.1	27.7	23.8
Kimi-K2.6	99.8	100.0	99.9	99.6	99.3	99.6	98.7	79.6	78.5	71.0	62.9	53.1	49.9	41.7
GPT-5.6-Sol	99.9	99.9	99.2	99.3	99.2	98.7	98.9	72.1	65.8	55.9	48.6	37.5	34.6	30.0
HumanLM-Opinion	55.2	60.3	60.9	62.5	60.0	59.6	62.4	22.2	19.6	22.2	20.4	18.6	15.7	20.2
USP	39.9	44.1	44.9	47.3	48.7	49.1	45.4	58.8	54.6	47.3	41.0	34.2	28.0	29.3
UserLM-8B	74.1	65.6	52.3	41.1	37.5	31.0	31.9	67.9	59.7	43.2	30.0	25.8	18.7	13.4
SFTUser-14B	77.7	71.4	60.8	49.3	46.2	42.2	39.9	91.5	82.2	69.0	60.3	58.0	52.2	51.8
UserLM-2.0-SFT	89.6	88.4	81.9	76.1	68.2	61.5	62.0	93.9	89.0	72.8	57.3	48.1	36.3	33.5
UserLM-2.0	97.3	96.9	93.6	90.0	85.3	81.8	80.5	98.3	97.3	90.6	81.9	74.9	66.2	65.8
Human Reference	88.6	95.5	91.3	86.6	85.0	82.2	83.1	100.0	100.0	100.0	100.0	100.0	100.0	100.0

Table 5: Results of the intent complexity stress test. We break down the performance (on Quality and Terminated) of the simulators based on the complexity of the input task intent (i.e., the number of sub-goals of the intent). Task intents range from 1 (simple) to 7 (complex) sub-goals. A majority of the models demonstrate a decrease in quality and termination rate as task intents increase in complexity.

	Coverage				Behavior		
Model	Coverage	Full coverage	Opening coverage	Opening share	Revisitation	Repeat-only	Empty-Turn
DeepSeek-V4-Pro	85.1	59.3	64.8	75.8	65.0	57.4	12.6
Llama-3.1-8B	76.9	45.2	59.0	76.3	53.4	43.1	24.2
Qwen3.5-9B	79.2	49.3	67.5	84.8	41.8	35.7	14.6
GPT-5.6-Luna	85.8	63.6	74.1	85.6	51.1	47.8	9.1
Kimi-K2.6	86.1	60.2	64.4	74.7	52.5	47.2	4.6
GPT-5.6-Sol	79.7	52.0	58.9	74.8	52.4	45.0	3.7
HumanLM-Opinion	58.2	34.8	40.4	64.7	47.3	28.5	50.7
USP	63.2	33.3	31.0	49.0	33.9	20.4	46.6
UserLM-8B	64.2	35.3	42.5	64.7	35.3	24.0	31.3
SFTUser-14B	60.7	31.1	44.3	71.9	16.9	10.3	31.8
UserLM-2.0-SFT	76.0	44.0	44.3	57.4	40.2	31.8	18.7
UserLM-2.0	88.0	62.8	59.0	66.4	39.1	33.0	4.4
Human Reference	96.0	87.6	49.0	51.4	38.8	33.3	6.8

Table 6: Finer-grained coverage and behavioral metrics for UserLM-2.0 and baselines. The depth of shading scales with proximity of values to the Human Reference values. Although UserLM-2.0 and prompted assistant models achieve similar Coverage scores, with UserLM-2.0 performing the best, they introduce information at different rates. Prompted assistant models front-load information more, with higher Opening Share scores, and are also more likely to spend new user turns only repeating previously introduced information, as seen with higher Repeat-Only values.

the trained discriminators. Figure 15 contains results for UserLM-2.0. The rows represent validation steps, while columns represent the discriminator used to evaluate them. Each index then contains the average $P_{\text{disc}}(y = \text{real} \mid \hat{u_1}...\hat{u_k}, c)$ for the corresponding validation step rollouts using the corresponding discriminator.

For UserLM-2.0, the matrices have a triangular structure. The lower triangular component, featuring validation steps evaluated by earlier discriminators, each predominantly have high $P_{\text{disc}}(y = \text{real})$ values for simulated rollouts. This indicates that as models learn to fool a newly retrained discriminator, they also keep on fooling earlier discriminators. Simultaneously, the upper triangular component predominantly features low $P_{\text{disc}}(y = \text{real})$ values. This is intuitive, as each retrained discriminator is trained on both the most recent 25 steps and simulated conversations from earlier in

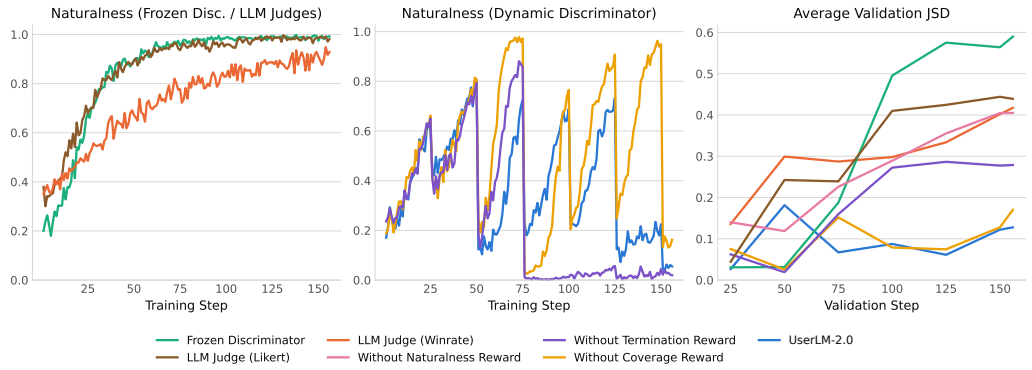


Figure 14: (a) Naturalness reward throughout training for the Fixed Discriminator and LLM Judge - Likert and Winrate ablations; (b) Naturalness reward throughout training for the dynamic discriminator experiments; (c) Average of the Jensen-Shannon Divergence between utterance-length histograms and turn-count histograms for model-generated and real validation conversations. Lower is better.

training. This enables retrained discriminators to remain robust to and discourage reward hacking behavior from early steps as well.

Dynamic discriminators help maintain alignment with human reference distributions. To better understand how dynamic discriminators then affect alignment with the human reference distribution, we look at output-length and conversation turn-counts as proxies. Specifically, we construct histograms of output-length and turn-counts for all models, using their validation rollouts during training, and compare them against histograms for corresponding human interactions using Jensen-Shannon Divergence (JSD). Figure 14(c) plots the averaged JSD value over training. Both the UserLM-2.0 and No Coverage runs sustain a low JSD throughout training and can correct themselves when JSD spikes. The alternative reward variants and the No Naturalness ablation, on the other hand, each increasingly diverges. This divergence is driven primarily by an increase in output-length, with the LLM judge variants ultimately reaching up to 400 words per turn on average.

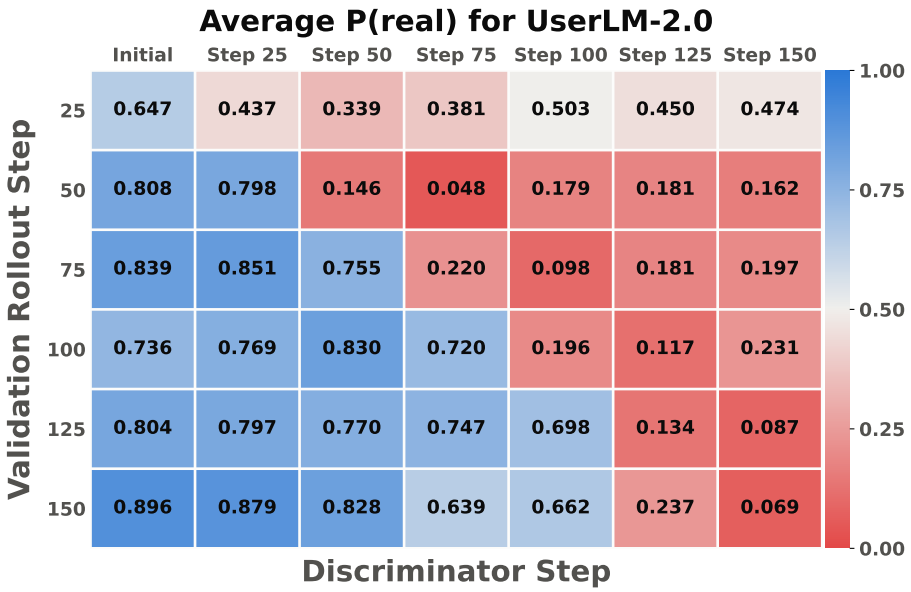


Figure 15: For UserLM-2.0, we perform inference with each discriminator trained during RL on validation rollouts from each validation step and report $P_{\text{disc}}(y = \text{real} \mid \hat{u_1}...\hat{u_k}, c)$ for generated conversations. Blue and red indicate high and low probabilities respectively. UserLM-2.0 learns not only to fool the most recent discriminator, but also keeps fooling older ones as well.