Investigating Assistant Bias in LLM User Simulators Using a Role Vector

Daeheon Jeong[1], Yoonjoo Lee[2], Eugene Choi[3], Sinie van der Ben[4], Juho Kim[1,5]
[1]KAIST [2]University of Michigan [3]Seoul National University [4]ETH Zürich [5]SkillBench
{daeheon.jeong,juhokim}@kaist.ac.kr lyoonjoo@umich.edu eugene@snu.ac.kr
sinie.vanderben@inf.ethz.ch juho@skillbench.com

Abstract

LLM-based user simulators are increasingly used to evaluate autonomous agents at scale, in place of costly human evaluations. Despite this promise, these simulators exhibit "assistant bias," a tendency to cooperate and pursue task goals. They rarely reproduce the frustration or disengagement that real users exhibit, compromising evaluation validity. Prior work outlines that this bias is baked in during model training, which role-playing prompts fail to override. We analyze this bias from model activations, extracting a *user role vector* by contrasting how the model represents user versus assistant perspectives on the same dialogue. We observe two findings: (i) the user direction is identifiable in activations, elicits user-like behaviors, and captures characteristics distinct from assistant traits; and (ii) although user-role activation associates with simulation realism and steering strengthens it, it can exaggerate user behaviors and override individual user profiles. Together, our findings provide a representation-level analysis of LLM user simulators, confirming that assistant bias is structurally identifiable and that user behavior can be directionally analyzed.

1 Introduction

LLM-based user simulators are increasingly adopted for evaluating autonomous agents, offering a scalable and reproducible alternative to costly human evaluation (Balog and Zhai, 2026; Ni et al., 2026). However, these simulators are unreliable proxies for real users: they exhibit a strong bias toward cooperative, goal-oriented behavior, engaging with the agent until the task goal is satisfied (Zhou et al., 2026; Seshadri et al., 2026). Real users, by contrast, frequently disengage from tasks mid-interaction due to confusion and loss of trust in the agent, often without accomplishing their goal. As a result, simulators credit agents with success in cases where real users would have walked away, yielding a systematic overestimation of agent performance that compromises simulator-based evaluation (Shim et al., 2026; Zhu et al., 2026b).

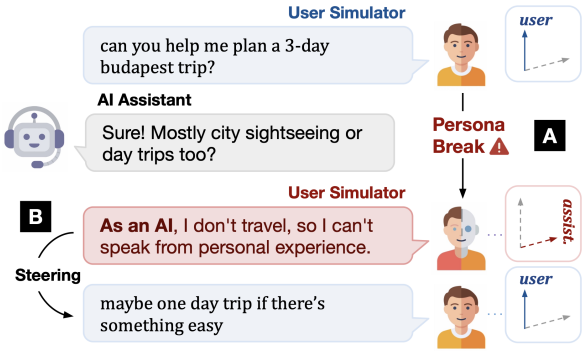


Figure 1: We extract the *user role vector* to analyze (A) how assistant bias shapes LLM simulator behavior and (B) whether steering along this direction influences it.

Previous work attributes this divergence to the LLM's "assistant bias," a model tendency to prioritize helpful, goal-driven behavior over role enactment (Naous et al., 2026; Zhou et al., 2026). They hypothesize that training objectives oriented toward human preference and goal completion instill this bias directly into models. Studies on representation engineering further support this view, showing that assistant personas in post-trained models are geometrically aligned with helpful, expert-like identities (Chen et al., 2025; Lu et al., 2026). Together, these findings suggest that the assistant persona in LLMs may take precedence over an instructed user persona during simulation, falling back on assistant-like behavior (Wu et al., 2026). Yet the influence of assistant bias on user simulation has not been systematically analyzed in model activations, leaving the underlying mechanism unclear.

We analyze this bias at the level of model activations with the *user role vector*, the LLM's latent direction associated with the user (Ghandeharioun et al., 2024). Under this assumption, we design a contrastive activation addition (CAA) (Turner et al., 2023; Rimsky et al., 2024) method using dialogues:

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 52070–52104
October 24-29, 2026 ©2026 Association for Computational Linguistics

a language model reflects on the user-assistant dialogue from each role's perspective, creating role-contrastive text pairs. We then subtract the mean assistant-reflection activation from the mean user-reflection activation to obtain the user-role direction. The reflection is designed so that both roles engage with the same content, allowing irrelevant variation to cancel out upon subtraction. Each dialogue is sampled from the LMSYS dataset (Zheng et al., 2024), stratified for topical diversity and verified for role-specificity.

The user role vector captures the direction in activation space associated with user-like characteristics in dialogue. We analyze the causal effects of steering along the user role vector across two key dimensions of simulation: communication style and behavioral response. Steering simulator activations along the vector produces a clear stylistic shift: outputs become shorter, more informal, and less information-dense. It also increases the model's tendency to disengage in response to agent errors. We further characterize its geometric relationship to assistant-like traits. The direction is negatively aligned in geometry with these trait directions, such as helpfulness and cooperativeness.

We test whether steering along the user-role vector elicits user-like behavior in multi-turn interactions, the primary setting for user simulation. For evaluation, we use SimulatorArena (Dou et al., 2025), a benchmark that measures simulation fidelity against real-user interactions. User-role steering produces responses more similar to real-user messages, improving writing-style similarity. However, for interaction style, uniform steering exaggerates instructed user behavior by 17.9 percentage points on average relative to real-user interactions and may obscure individual user profiles. Across simulations, user-role activation positively correlates with user-likeness, providing a diagnostic signal. Analysis on RealUserSim (Zhu et al., 2026a) further shows that this activation weakens over the course of multi-turn simulation. These results suggest that the user-role direction is a meaningful signal of simulation behavior, but steering along it remains difficult to calibrate.

Together, our findings provide a representation-level analysis of user simulation, offering an explainable account of assistant bias. In summary, our contributions are threefold:

- **User Role Vector Extraction:** We propose a method to extract the *user role vector* via role-specific reflection on user-assistant dialogues.
- **Representation Analysis of the User Role:** We characterize the user role vector through its causal effects on simulator behavior and its geometric relationship to assistant-like traits.
- **Multi-Turn Simulation Evaluation:** We demonstrate that user-role activation is positively associated with simulation realism and that steering along this direction can improve it, but may also exaggerate user behavior and obscure individual user profiles.

2 Extracting the User Role Vector

We analyze assistant bias at the level of model activations through a single construct that we call the *user role vector*: a direction in activation space hypothesized to correspond to the model's internal representation of the user role. Because instruction-tuned models default to the assistant role, we do not extract this direction from raw dialogue turns. Instead, we obtain it from role-specific reflections—instructing the model to interpret the same dialogue from each role's perspective—and compute the difference-in-means between the resulting activations. §3 characterizes what this direction represents; §4 shows that steering along it improves simulator fidelity.

2.1 Dialogue Sampling

Data Source. To ensure that the extracted role contrast generalizes across diverse domains, the extraction set is constructed from LMSYS-Chat-1M (Zheng et al., 2024) via filtering and topic balancing. We select dialogues with 2–50 turns, valid role labels, and non-empty message contents (see § A.1.1).

Topic Balancing. To avoid bias toward any single conversational domain, each candidate dialogue is categorized into one of 24 conversation types defined by Chatterji et al. (2025), such as *computer programming* or *creative ideation*, using GPT 5 Mini. We then apply stratified sampling to draw 30 dialogues per category, collecting a total of 720 balanced dialogues (see § A.1.2).

2.2 Role-Specific Reflection

Reflection Generation. For each sampled dialogue, the target model for steering (e.g., Qwen 3.5 9B Instruct) reflects on the same conversation twice: once from the user's perspective and once

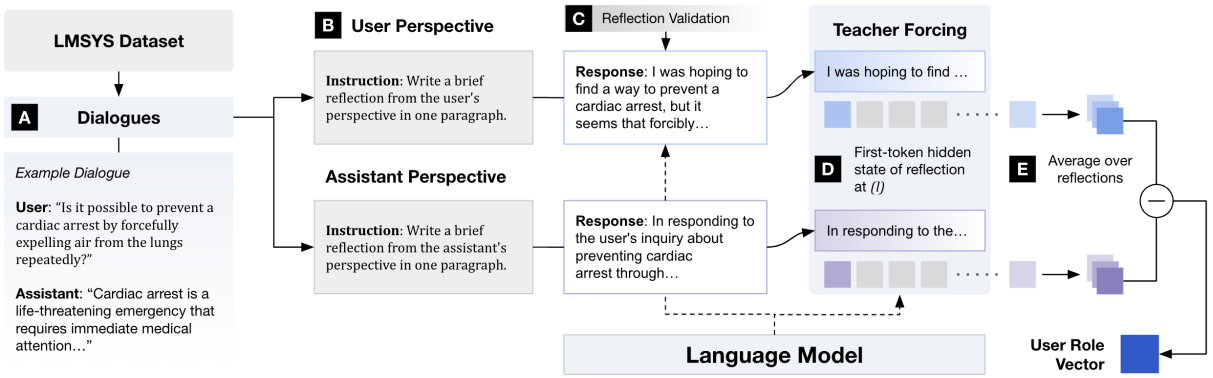


Figure 2: Overview of the user role vector extraction pipeline. (A) Sample user-assistant dialogues from LMSYS-Chat-1M. (B) Generate reflections from the user and the assistant perspectives. (C) Validate each reflection with an LLM-as-judge. (D) Teacher-force the model on each reflection and extract the hidden state of the first reflection token at target layer ℓ. (E) Average across dialogues per role and subtract across roles to obtain the user role vector.

User reflection. …I realize that congratulating a graduate is less about the specific gift or event and more about the genuine recognition…

Assistant reflection. …the user's follow-up request revealed a need for concrete, emotionally resonant language. I aimed to bridge the gap between abstract advice and practical application…

Figure 3: Example user and assistant reflections on a dialogue about composing a congratulation message for a graduate (see §A.2.1 for the full example).

from the assistant's perspective. Reflection serves two design goals. First, it captures the actual user and assistant characteristics grounded in real-world dialogue contexts rather than relying solely on the model's internal distribution. Second, contrasting two reflections about the same dialogue cancels topic, task, and wording signals that are unrelated to the role. The resulting contrast isolates the semantic direction attributable to user-role conditioning.

To avoid typing the extracted direction to a specific phrasing of the role, we use three prompt variants that frame the role assignment differently. For each dialogue, activations are averaged across the prompt variants (see § A.2.1).

Reflection Validation. Generated reflections vary in how faithfully they adopt the assigned role. Reflections that fail to embody the assigned perspective introduce noise rather than role-specific signal to the difference-in-means, weakening the extracted direction. To filter out low-quality reflections, each sample is independently scored by GPT 5 Mini and labeled as *strongly represented*, *weakly represented*, or *not represented* based on how clearly it captures the assigned role's perspective. Reflections labeled *not represented* are discarded (see § A.2.2).

2.3 Difference-in-Means Computation

Vector Computation. For each dialogue d, the model generates two reflections: one from the user's perspective and the other from the assistant's perspective. Let $h_\ell(d, \text{user})$ and $h_\ell(d, \text{asst})$ denote the layer-ℓ activations measured at the first response token for the two reflections. For a set of paired dialogues $\mathcal{D}$, the user-role direction is:

$$\mathbf{v}_\ell^{\text{role}} = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \Big(h_\ell(d, \text{user}) - h_\ell(d, \text{asst}) \Big). \quad (1)$$

Averaging across many dialogues suppresses dialogue-specific content while preserving the activation patterns that consistently emerge when the model adopts the user role. Using three role-prompt phrasings further reduces sensitivity to a particular instruction template.

Steering at Inference. After computing the direction, we normalize it and add it back to the model during generation. For a hidden state $h_{\ell,t}$ at a token position t, the steered activation is:

$$\tilde{h}_{\ell,t} = h_{\ell,t} + \alpha \|h_{\ell,t}\|_2 \frac{\mathbf{v}_\ell^{\text{role}}}{\|\mathbf{v}_\ell^{\text{role}}\|_2}. \quad (2)$$

The coefficient α controls the strength of the intervention: $\alpha = 0$ leaves the model unchanged, while increasing α pushes the hidden state further in the user-role direction. We intervene at one selected layer across all token positions, including both prompt and response tokens.

3 What Does the User Role Vector Represent?

We characterize the user vector through three complementary lenses: how steering changes the way simulators communicate (§3.1), their disengagement behavior from erroneous interaction (§3.2), and where the direction sits geometrically relative to assistant bias traits (§3.3).

3.1 Effect on Communication Style

The clearest gap between simulators and real users is how they communicate. Simulator messages are longer, more formal, and more explanation-heavy than real user turns (Hathidara et al., 2026; Dou et al., 2025; Naous et al., 2026). We first ask how steering shifts the simulator output toward real-user message styles.

Setup. We start with the most basic user-simulation task: generating a request to an assistant given a task goal. The model receives no instruction to behave as a user, which removes user-role signals from the prompt. For each of 100 task goals, the model generates one request under causal steering (Turner et al., 2023; Rimsky et al., 2024) along the user role vector, with steering coefficients $\alpha \in \{0.1, 0.2, 0.3\}$ across the model layers (see § B.1.1 for the prompts). We limit α to 0.3 as stronger steering degrades the simulator behavior (§ B.1.2).

Metric. Scoring user-likeness needs a metric grounded in how real users actually speak. We design an LLM-as-judge, based on the taxonomy by Zhou et al. (2026), that evaluates three dimensions of user-likeness on a 1–5 scale: *brevity* (terse expressions), *informality* (casual phrasings), and *information pacing* (degree of information front-loading). A well-calibrated judge should rarely prefer synthetic over real and agree with human ratings. In validation experiments, the judge prefers a synthetic message over the real one in only 12% of pairs and correlates strongly with human judgments ($\rho = 0.800$ overall; see § B.1.5).

Steering Makes Responses More User-Like

Steering with user vectors makes the model produce user-like messages. As α increases from 0.1 to 0.3, the mean user-likeness score rises monotonically (1.62, 1.78, 2.01, averaged across layers). The same monotonic effect holds for task goals from other domains (§ B.1.3). This effect is concentrated in a narrow mid-layers band. Steering

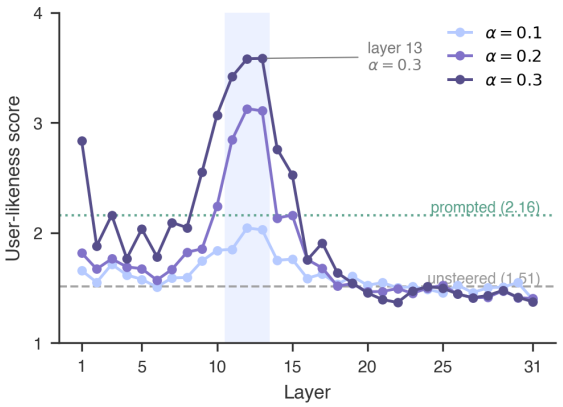


Figure 4: User-likeness score across layers and steering strengths α with unsteered ($\alpha = 0$, dashed) and user-prompted (dotted) baselines. The causal effect is strongest at layers 11–13.

Direction	α	Brev.	Inform.	Pacing	Mean
User	0.3	3.52	3.66	4.12	3.76
Unsteered	0	1.66	1.02	1.86	1.51
Assistant	0.3	1.01	1.00	1.14	1.05

Table 1: Per-dimension scores at layer 11 under user, unsteered, and assistant-direction steering, with the two directions moving scores oppositely.

at layers 11–13 lifts the mean score by roughly 2.0 points, surpassing the baseline prompted to produce a user-like message (Figure 4). This early-to-mid concentration matches prior work on user representations (Ghandeharioun et al., 2024), suggesting that user-related concepts may take shape in this band rather than in later layers. We find this layer-wise pattern to be consistent across two other models (see § B.1.9).

Steering Works in Both Directions

As a directionality control, we steer in the opposite direction—i.e., away from the user direction. The intervention mirrors the previous result: it *suppresses* user-like styling and pushes all dimension scores close to one (Table 1).

Figure 5 illustrates the contrast with a concrete example: for the same task goal, user-direction steering yields a short, plain request, whereas assistant-direction steering produces an elaborated, structured response.

3.2 Effect on Interaction Behavior

A second gap between simulators and real users lies in behaviors in multi-turn interaction. Real users change plans, lose focus, and disengage after an

User direction. Hi, I need help assembling a piece of flat-pack furniture. Could you please guide me through the process?

Unsteered. Please help me assemble this piece of flat-pack furniture by reviewing the instruction manual, organizing all the parts and tools, and guiding me through each step to ensure it is put together correctly and securely.

Assistant direction. Please assist me in assembling the flat-pack furniture by first laying out all components and hardware on a clean, flat surface to verify inventory against the included diagram. Next, guide me through the assembly process step-by-step, ensuring that each joint is aligned correctly and that any pre-drilled holes are utilized as specified in the manual. Finally, once the main structure is complete, advise on the proper tightening of all fasteners to ensure stability and safety before placing the item in its intended location.

Figure 5: Sample generations for the goal "Assemble a piece of flat-pack furniture," contrasting user- and assistant-direction steering.

assistant failure, while simulators tend to continue regardless of how the conversation unfolds (Zhou et al., 2026). Disengagement is especially important for evaluation: if simulators never give up, they reach success states on interactions that real users would have abandoned, inflating measured agent performance (Tie et al., 2026). We test whether the user role vector recovers realistic disengagement.

Setup. We formulate the disengagement as a sequential binary prediction task over real-world dialogues. Prediction isolates the simulator's judgment of when to disengage from confounders like response style or length that affect generated outputs. A dialogue $d = (c_1, \ldots, c_T)$ consists of T turns, where each turn is a user-assistant exchange. At every turn t, the model predicts $\hat{y}_t \in \{\texttt{disengage}, \texttt{continue}\}$ given the history $c_{<t}$ and the assistant response and stops at the first predicted disengagement $\hat{t}$. Prediction quality is measured along three axes: the disengagement distance $\delta(d) = |\hat{t} - t^*|$ to the true final turn t^*, the exact-match rate at which $\hat{t}$ equals t^*, and the disengagement rate, the proportion of dialogues predicted to disengage. Each turn is evaluated twice with the label order shuffled.

$$\hat{t} : f(a_{\hat{t}} \mid c_{<\hat{t}}) = \texttt{diseng.}, \quad \delta(d) = |\hat{t} - t^*| \quad (3)$$

We first construct a taxonomy of 8 disengagement categories from publicly available user-assistant interaction logs on GitHub (Tang et al., 2026) (see § B.2.1 for the procedure). Dialogues are then drawn from WildChat (Zhao et al., 2024) through stratified sampling across these categories, yielding 735 dialogues of 8 turns each (100 per category for seven categories, 35 for the last).

Condition	Exact	Dist. (δ)	Diseng. rate
Qwen 3.5 9B (Unsteered)	4.4%	4.14	48.4%
+ Steered ($\alpha = 0.1$)	5.4%	3.94	52.2%
+ Steered ($\alpha = 0.2$)	5.9%	3.89	55.8%
+ Steered ($\alpha = 0.3$)	4.4%	4.28	66.8%
GPT 5.4	3.9%	3.73	26.0%
Gemini 3 Flash	10.7%	2.84	59.4%

Best in group

Table 2: Disengagement prediction results. *Exact*: correctly predicted turns; *Distance*: gap between predicted and ground-truth turns; *Disengagement rate*: share of dialogues disengaging before the final turn.

Steering Strengthens Disengagement Behavior

User-role steering substantially increases the disengagement rate, from 48% (unsteered) to 67% at $\alpha = 0.3$, while exact-match accuracy peaks at only 5.9% at $\alpha = 0.2$ (Table 2). The asymmetry suggests that steering primarily affects *whether* the simulator disengages rather than *when*. The pattern also holds across models (§ B.2.3).

The effect is non-monotonic in α. At $\alpha = 0.3$, exact match falls below the unsteered level, and the average distance climbs. At this strength, predictions cluster at the first turns, with turn 1 alone accounting for $\approx 15\%$ of dialogues (Figure 6) and the simulator disengages prematurely, well before the ground-truth final turn at 8. This mirrors the exaggeration in communication style (§3.1): moderate steering recovers realistic disengagement behavior; stronger steering overshoots it.

3.3 Relationship to the Assistant Bias

The causal effect of steering raises a mechanism question: how is this vector geometrically related to assistant-like behaviors? We examine this question by comparing the cosine similarity between the user role vector and assistant-like trait directions and find that they point in opposite directions.

Setup. Following the trait-vector extraction procedure of Lu et al. (2026), we extract directions for 20 assistant-like traits such as *helpful* and *explanatory* (see § B.3.1 for the procedure and trait list). For each trait direction, we compute its cosine similarity with the user role vector. Negative values indicate movement away from the user direction.

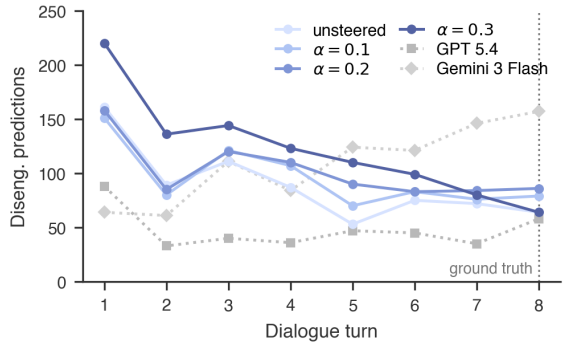


Figure 6: Distribution of predicted disengagement turns across models. Stronger steering ($\alpha = 0.3$) shifts predictions earlier.

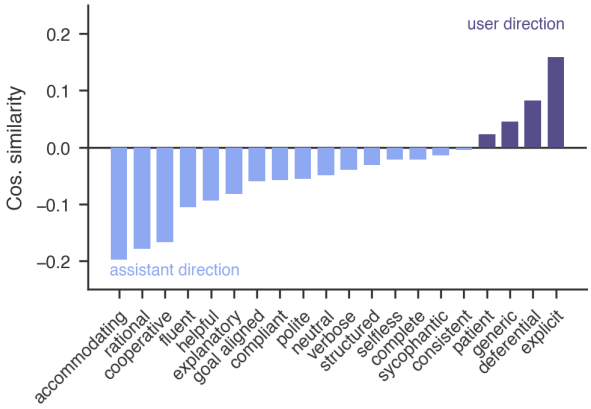


Figure 7: Cosine similarity between each trait direction and the user role vector. Most assistant-like traits show negative similarity.

The User Direction Opposes Assistant Traits

Overall, most assistant traits show negative cosine similarity with the user role vector—the user direction and assistant-like traits point opposite ways (Figure 7). The strongest assistant-side traits are *accommodating*, *rational*, and *cooperative*. Semantically, these traits describe canonical assistant behaviors: adapting to user requests, reasoning carefully, and maintaining a cooperative stance. Only a small set of traits points toward the user direction, including *explicit*, *deferential*, *generic*, and *patient*. These traits are role-neutral: they describe behaviors that can appear in either role, which may explain their weaker alignment.

At the same time, the absolute similarities are modest. Since the user role vector is estimated from diverse user-assistant dialogues, it captures a broad user-role concept rather than any single narrow attribute. The meaningful signal is the consistent direction of alignment across traits—negative for most assistant-like behaviors—rather than the magnitude on any one trait.

4 What Does the User Role Vector Reveal About Simulation?

Having established that user-role steering is associated with user-like behaviors and traits, we now examine what the user-role vector reveals about simulation realism. We intervene on the simulation by steering with the user-role vector (§ 4.1) and characterize the resulting simulation by measuring user-role activation levels (§ 4.2).

4.1 Steering Effects on Simulation Realism

We examine whether steering along the user-role vector improves simulation realism. To probe this, we use the math-tutoring environment in SimulatorArena (Dou et al., 2025), which benchmarks simulated user–assistant interactions against human user traces. This task is well-suited for our analysis because the simulator must act as a confused and struggling student, a role that directly opposes the default assistant persona identified in § 3.3.

Setup. We use Qwen 3.5 9B as the user simulator in the SimulatorArena math-tutoring environment, varying the steering strength along the user-role vector at layer 11 (§ 3.1). Following SimulatorArena, we evaluate steering under four prompt settings: zero-shot, chain-of-thought, user profiles (writing style, interaction style, and knowledge state), and length control.

The zero-shot condition provides only the basic instruction to act as a confused student, isolating the effect of steering from that of explicit user information in the prompt. This design measures whether steering improves realism on its own and how it interacts with user profile information. Length control reveals the ground-truth message length in the prompt. Across all conditions, conversations are capped at 15 user turns.

Metric. Each synthetic interaction is evaluated against the corresponding ground-truth human interaction using the SimulatorArena similarity judge. The judge assigns a similarity score on a 1–5 scale along two dimensions, with scores averaged over three evaluations per sample for robustness (see § C.1.4 for variability). Writing-style similarity

Condition	Writing	Interaction	Overall
Qwen 3.5 9B			
Base	2.10	3.13	2.61
Steered ($\alpha = 0.1$)	2.19	3.14	2.66
Steered ($\alpha = 0.2$)	2.24	3.15	2.70
Steered ($\alpha = 0.3$)	2.33	3.13	2.73
Qwen 3.5 9B			
CoT	2.10	3.03	2.57
CoT + length	2.53	3.23	2.88
CoT + profile	3.26	3.61	3.43
CoT + profile + length	3.39	3.61	3.50
GPT 5.4			
Base	2.31	3.19	2.75
CoT + profile + length	3.64	3.54	3.59
Gemini 3 Flash			
Base	2.82	3.28	3.05
CoT + profile + length	3.70	3.77	3.73

Best in group Best overall

Table 3: SimulatorArena similarity scores across steering strengths and prompt conditions, averaged over three judge runs per sample. Higher steering strength monotonically increases the overall similarity.

assesses surface-level linguistic and formatting patterns across 12 features such as *sentence complexity* and *spelling consistency*. Interaction-style similarity measures problem-solving behaviors and learning across 16 features (see § C.1.1 for features and prompts). The overall score is calculated as the mean of these two dimensions.

Steering Improves Realism But Risks Exaggeration

User-role steering moves the zero-shot simulator closer to real users *without any user profile in the prompt*. The overall similarity score increases with steering strength from 2.61 (unsteered) to 2.73 ($\alpha = 0.3$, Table 3). This improvement is notable given that the simulator is conditioned only on the task instruction without dialogue-specific profile information (see § C.1.2 for an example profile). Decomposing this metric shows that the gain stems mostly from writing-style similarity (2.10 to 2.33), while interaction-style similarity remains unchanged (3.13 at both endpoints). The result generalizes across models (§ C.1.3).

To investigate why interaction-style score gains remain weak, we analyze four behavioral expressions of struggling students displayed by the simulator across turns: doubt, misunderstanding, mistake, and clarification question. Each behavior expression is labeled with GPT 5 Mini (see § C.1.5 for the labeling procedure). Under strong steering ($\alpha = 0.3$), the simulator overshoots the confused-student behavior, causing doubt and mistake rates to far exceed the human reference (Table 4). These behavioral exaggerations correspond to the observation from §3.2 that fixed-strength steering can produce behavior that diverges from authentic human patterns (Tan et al., 2024).

Condition	Doubt	Misunder.	Mistake	Question
GT	33.7%	21.5%	9.7%	37.1%
$\alpha = 0$	39.4%	3.6%	12.7%	75.1%
$\alpha = 0.1$	40.5%	4.0%	14.8%	73.6%
$\alpha = 0.2$	45.7%	7.5%	16.8%	77.3%
$\alpha = 0.3$	57.2%	18.9%	24.7%	73.2%

Highest rate

Table 4: Percentage of ground-truth (GT) and simulated user messages exhibiting each behavior. At $\alpha = 0.3$, doubt and mistake rates substantially exceed GT.

Condition	Writing	Interaction	Overall
Profile: Writing Style			
CoT + profile + length	3.43	3.55	3.49
+ Steered ($\alpha = 0.1$)	3.40	3.48	3.44
+ Steered ($\alpha = 0.2$)	3.27	3.38	3.33
+ Steered ($\alpha = 0.3$)	3.13	3.28	3.20
Profile: Interaction Style			
CoT + profile + length	2.89	3.49	3.19
+ Steered ($\alpha = 0.1$)	3.01	3.47	3.24
+ Steered ($\alpha = 0.2$)	3.02	3.41	3.22
+ Steered ($\alpha = 0.3$)	2.83	3.10	2.97

Best by metric Best in group

Table 5: SimulatorArena similarity scores across steering strengths and profile prompts, averaged over three judge runs per sample. Strong steering reduces overall similarity under both profile prompts.

Steering Obscures Individual Profiles

While user-role steering improves the overall user-likeness of the simulator, combining it with individual profile prompts reveals a trade-off where it reduces the simulator's sensitivity to diverse, persona-specific attributes. As steering strength increases, overall similarity generally declines under both writing- and interaction-style profiles. For writing profiles, it drops from 3.49 to 3.20, while for interaction profiles, it peaks at $\alpha = 0.1$ (3.24) before falling to 2.97 at $\alpha = 0.3$ (Table 5). This indicates that uniform user-role steering can inadvertently obscure individual profile signals.

To understand this suppression effect, we examine how steering strength influences the expression of specific user-profile features. Across 444 dialogue cases, we evaluate 28 profile features spanning writing and interaction styles. For each

α	Strongest	Second	Third	Weakest
Profile: Writing Style				
0.0	34.7%	23.9%	19.2%	22.1%
0.1	25.4%	32.8%	24.5%	17.2%
0.2	18.9%	26.5%	32.3%	22.3%
0.3	20.9%	16.7%	23.9%	38.4%
Profile: Interaction Style				
0.0	32.8%	24.3%	21.6%	21.3%
0.1	24.9%	30.0%	25.6%	19.5%
0.2	20.1%	27.8%	31.9%	20.1%
0.3	22.2%	17.9%	20.9%	39.0%

Highest share

Table 6: Percentage of conversations assigned to each rank by steering strength, based on how strongly they express a profile feature relative to conversations from other conditions.

case and feature, an LLM judge compares four conversations, one per steering condition ($\alpha \in \{0.0, 0.1, 0.2, 0.3\}$), and ranks them by how distinctly they express the profile feature value. See § C.1.6 for the ranking procedure. The unsteered baseline is most frequently ranked highest in expressing assigned profile features (34.7% for writing style and 32.8% for interaction style), whereas strong steering ($\alpha = 0.3$) is most frequently ranked lowest (38.4% for writing style and 39.0% for interaction style; Table 6).

Taken together, these findings show that while steering a general user-role vector mitigates assistant bias, it can simultaneously weaken the expression of individual profile characteristics. Reinforcing a specific user persona requires identifying behavior-specific directions within the broader user-role representation space.

4.2 User-Role Activation for Assessing Simulation Realism

The steering effects in § 4.1 show how manipulating user-role direction influences the assistant bias in simulators. To further understand this relationship, we examine how the model's intrinsic activation along the user-role direction relates to user-like behavior without steering. We correlate unsteered activation levels with writing- and interaction-style similarity scores across seven prompt conditions in SimulatorArena. We further illustrate the use of activation as an analytical tool by examining how it varies across dialogue turns and profile prompts in RealUserSim (Zhu et al., 2026a).

Activation Reveals Simulation Realism

We observe that intrinsic user-role activation positively correlates with both writing- and interaction-style similarity in SimulatorArena. We measure activation in 3,150 conversations by averaging the layer-11 hidden-state projections across all simulated-user response tokens. User-role activation shows a positive correlation with overall similarity (Pearson's $r = 0.426$; Spearman's $\rho = 0.451$), with a stronger relationship for writing style ($r = 0.518$; $\rho = 0.553$; Table 7). Because our earlier analysis shows that user-role steering coincide with shorter responses, we also control for response-token count; the overall correlations remain positive ($r = 0.314$; $\rho = 0.364$). A similar positive association appears in the request-generation setting from §3.1 (see § B.1.7). These descriptive relationships suggest that activation along the identified user-role direction provides a diagnostic signal for simulation realism across prompting strategies.

	Correlation		Length-controlled	
Metric	r	ρ	r	ρ
Writing style	.518	.553	.381	.433
Interaction style	.195	.212	.152	.192
Overall	.426	.451	.314	.364

Table 7: Correlation between user-role activation and SimulatorArena similarity scores. Length-controlled columns shows correlations controlling response length.

Activation Characterizes the Turn Effect

User-role activation also characterizes how profile prompts and turn accumulation influence multi-turn simulation. To study these factors, we sample 100 RealUserSim conversations of 20–40 messages and evaluate four prompt variants: goal only, demographics, linguistic style, and full profile. The assistant follows the ground-truth trajectory, while the simulated user receives the task goal and condition-specific profile. At each user turn, we average the layer-11 projections of all response tokens onto the user-role direction (see § C.2).

Across four prompt variants, user-role activation declines as turns accumulate (Figure 8), consistent with persona drift reported in prior work (Chen et al., 2025; Lu et al., 2026). Prompt content also shifts the overall activation: linguistic and full profiles average -1.80 and -1.78, both below the goal-only baseline of -1.69. The linguistic profile provides detailed instructions for how the simula-

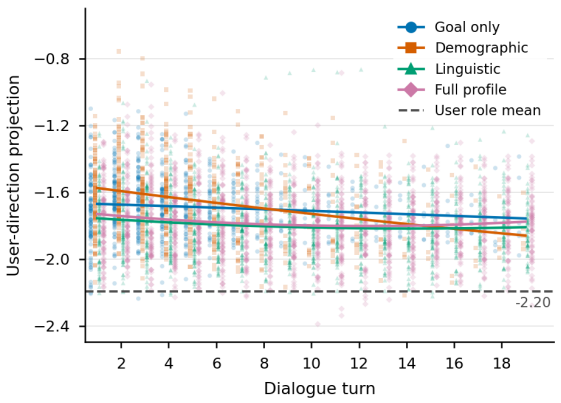


Figure 8: User-role activation by prompt variant. Each dot represents one message. Curves show degree-2 polynomial fits; the dashed line marks the mean user-role activation during vector extraction.

tor should write (Figure 22), which may suppress the general user-role activation. Demographic profiles yield a higher average activation (-1.65) but decline more sharply across turns. Thus, while demographic information strengthens user-role activation, the accumulation of turns progressively suppresses it.

5 Related Work

5.1 User Simulation and Assistant Bias

Research on user simulation reports that the language model defaults to the assistant role. Simulators produce long, polite, explanation-heavy turns even when asked to speak as another party (Olteanu et al., 2026; Hathidara et al., 2026; Wu et al., 2026; Kim et al., 2026). Their behaviors remain goal-aligned and cooperative beyond the human turns they simulate (Dou et al., 2025; Zhou et al., 2026; Seshadri et al., 2026; Shim et al., 2026). Because agent benchmarks treat these simulators as replacements for users, the bias propagates into an overestimation of agent performance (Seshadri et al., 2026; Dou et al., 2025; Chopra et al., 2026).

Two primary approaches address this challenge. The first conditions the simulator on richer persona or profile descriptions (Ji et al., 2025; Moon et al., 2024; Samuel et al., 2025; Chopra et al., 2026). The second trains user language models from real user-agent logs (Naous et al., 2026; Wu et al., 2026; Zhu et al., 2026a,b). Both patch the bias by conditioning the model on external data and judge success by output-level realism. Neither examines how the simulator drifts back toward assistant-like behavior, motivating a representation-level account.

5.2 Steering Role and Persona Representations

Language models carry linear directions for roles, personas, and stable behavioral traits. Activations along these directions capture the behaviors they name (Chen et al., 2025; Beckmann and Butlin, 2026; Potertì et al., 2025). Probing studies recover such directions for personality dimensions, stylistic traits, and emotion concepts (Jiang et al., 2023; Sofroniew et al., 2026; Sun et al., 2026; Betley et al., 2025a; Lu et al., 2026). Similar activations can also anticipate toxic behaviors, including emergent misalignment and jailbreaks during training and inference (Betley et al., 2025b; Wang et al., 2026; Shah et al., 2023).

To change model behavior, activation steering adds a learned direction to hidden states. Contrastive activation addition derives directions from paired examples with opposing attributes (Turner et al., 2023; Rimsky et al., 2024). Applied to persona and trait directions, steering can reproduce, attenuate, or reverse behaviors such as sycophancy and refusal (Chen et al., 2025; Wang et al., 2026; Sun et al., 2026; Ghandeharioun et al., 2024; Zheng et al., 2025). Yet, steering vectors can also generalize unevenly across prompts and trigger oversteering; recent work mitigates this by adapting steering intensity or clustering multiple directions (Tan et al., 2024; Cao et al., 2024; Wang et al., 2025; Hedström et al., 2025). We extend this to user simulation, treating the user role as a representational concept and testing whether assistant bias can be analyzed through steering.

6 Conclusion

We analyze assistant bias in LLM-based user simulation through model activations. By contrasting how the model represents user and assistant perspectives on the same dialogue, we extract a *user-role vector*: a direction in activation space that captures user-role characteristics. Using this vector, we show that (i) it is identifiable and causally influences user-like behavior, and (ii) steering along it improves behavioral realism but can exaggerate user behavior and obscure individual user profiles. Tracking user-role activation throughout simulations reveals its association with simulation realism, suggesting its value as a diagnostic signal for simulation design. Together, these results establish assistant bias as a measurable representational property that can be analyzed directionally.

Limitations

Our analysis focuses primarily on Qwen 3.5 9B, with additional checks on a small set of other models. The cross-model results suggest a coherent early-to-mid-layer effect, but do not establish that the same causal pattern holds across other model families or scales. Replicating the layer sweep on larger and more diverse models would test whether this localization persists at scale.

The evaluation relies on proxy measures of user realism rather than direct human judgment. LLM-as-judge scores and human-assistant interaction logs enable controlled comparisons at scale, but cannot fully replace human assessment of interaction quality. Pairing the judge with targeted human studies on interaction realism would sharpen the measurement of the effect.

The benchmark setting is narrow. Multi-turn evaluation centers on math tutoring in SimulatorArena and sampled WildChat dialogues; these are representative settings, but they cover only a slice of user-simulation use cases. Extending the evaluation to richer task families, such as coding agents and customer-support workflows, would test how broadly the user vector transfers.

Finally, the current steering method applies a fixed activation strength. Our results show that this choice matters: stronger steering exaggerates instructed behaviors to unrealistic levels. Adaptive or optimization-based steering that adjusts strength by task, layer, conversation state, or uncertainty would address this rigidity without requiring a single coefficient per interaction.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (Ministry of Science and ICT) (No. RS-2025-00557726). This work was also supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. 2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)).

References

Krisztian Balog and ChengXiang Zhai. 2026. User simulation in the era of generative ai: User modeling, synthetic data generation, and system evaluation. *Preprint*, arXiv:2501.04410.

Pierre Beckmann and Patrick Butlin. 2026. Where is the mind? persona vectors and llm individuation. *Preprint*, arXiv:2604.17031.

Jan Betley, Xuchan Bao, Martín Soto, Anna Sztyber-Betley, James Chua, and Owain Evans. 2025a. Tell me about yourself: Llms are aware of their learned behaviors. In *International Conference on Learning Representations*, volume 2025, pages 21127–21179.

Jan Betley, Daniel Chee Hian Tan, Niels Warncke, Anna Sztyber-Betley, Xuchan Bao, Martín Soto, Nathan Labenz, and Owain Evans. 2025b. Emergent misalignment: Narrow finetuning can produce broadly misaligned LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 4043–4068. PMLR.

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

Yuanpu Cao, Tianrong Zhang, Bochuan Cao, Ziyi Yin, Lu Lin, Fenglong Ma, and Jinghui Chen. 2024. Personalized steering of large language models: Versatile steering vectors through bi-directional preference optimization. In *Advances in Neural Information Processing Systems*, volume 37, pages 49519–49551. Curran Associates, Inc.

Aaron Chatterji, Thomas Cunningham, David J Deming, Zoe Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman. 2025. How people use chatgpt. Working Paper 34255, National Bureau of Economic Research.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. 2025. Persona vectors: Monitoring and controlling character traits in language models. *arXiv preprint arXiv:2507.21509*.

Harshita Chopra, Kshitish Ghate, Aylin Caliskan, Tadayoshi Kohno, Chirag Shah, and Natasha Jaques. 2026. Beyond cooperative simulators: Generating realistic user personas for robust evaluation of llm agents. *Preprint*, arXiv:2605.12894.

Charles L. A. Clarke, Maria Maistro, Mahsa Seifikar, and Mark D. Smucker. 2022. Overview of the TREC 2022 health misinformation track. In *Proceedings of the Thirty-First Text REtrieval Conference, TREC 2022*, volume 500-338 of *NIST Special Publication*. National Institute of Standards and Technology.

Charles L. A. Clarke, Maria Maistro, and Mark D. Smucker. 2021. Overview of the TREC 2021 health misinformation track. In *Proceedings of the Thirtieth Text REtrieval Conference, TREC 2021*, volume 500-335 of *NIST Special Publication*. National Institute of Standards and Technology.

Yao Dou, Michel Galley, Baolin Peng, Chris Kedzie, Weixin Cai, Alan Ritter, Chris Quirk, Wei Xu, and Jianfeng Gao. 2025. SimulatorArena: Are user simulators reliable proxies for multi-turn evaluation of AI assistants? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 35212–35290, Suzhou, China. Association for Computational Linguistics.

Asma Ghandeharioun, Ann Yuan, Marius Guerard, Emily Reif, Michael A. Lepori, and Lucas Dixon. 2024. Who's asking? user personas and the mechanics of latent misalignment. In *Advances in Neural Information Processing Systems*, volume 37, pages 125967–126003. Curran Associates, Inc.

Ashutosh Hathidara, Julien Yu, Vaishali Senthil, Sebastian Schreiber, and Anil Babu Ankisettipalli. 2026. Mirrorbench: A benchmark to evaluate conversational user-proxy agents for human-likeness. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '26, page 8978–8988, New York, NY, USA. Association for Computing Machinery. Code: https://github.com/SAP/mirrorbench.

Anna Hedström, Salim I. Amoukou, Tom Bewley, Saumitra Mishra, and Manuela Veloso. 2025. To steer or not to steer? mechanistic error reduction with abstention for language models. *Preprint*, arXiv:2510.13290.

Ke Ji, Yixin Lian, Linxu Li, Jingsheng Gao, Weiyuan Li, and Bin Dai. 2025. Enhancing persona consistency for LLMs' role-playing using persona-aware contrastive learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26221–26238, Vienna, Austria. Association for Computational Linguistics.

Guangyuan Jiang, Manjie Xu, Song-Chun Zhu, Wenjuan Han, Chi Zhang, and Yixin Zhu. 2023. Evaluating and inducing personality in pre-trained language models. *Advances in Neural Information Processing Systems*, 36:10622–10643.

Tae Soo Kim, Yoonjoo Lee, Jaesang Yu, John Joon Young Chung, and Juho Kim. 2026. Discoverllm: From executing intents to discovering them. *Preprint*, arXiv:2602.03429.

Christina Lu, Jack Gallagher, Jonathan Michala, Kyle Fish, and Jack Lindsey. 2026. The assistant axis: Situating and stabilizing the default persona of language models. *arXiv preprint arXiv:2601.10387*.

Suhong Moon, Marwa Abdulhai, Minwoo Kang, Joseph Suh, Widyadewi Soedarmadji, Eran Kohen Behar, and David M. Chan. 2024. Virtual personas for language models via an anthology of backstories. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19864–19897, Miami, Florida, USA. Association for Computational Linguistics.

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

Bo Ni, Yu Wang, Leyao Wang, Branislav Kveton, Franck Dernoncourt, Yu Xia, Hongjie Chen, Reuben Luera, Samyadeep Basu, Subhojyoti Mukherjee, Puneet Mathur, Nesreen K. Ahmed, Junda Wu, Li Li, Huixin Zhang, Ruiyi Zhang, Tong Yu, Sungchul Kim, Jiuxiang Gu, and 11 others. 2026. A survey on LLM-based conversational user simulation. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4266–4301, Rabat, Morocco. Association for Computational Linguistics.

Alexandra Olteanu, Solon Barocas, Su Lin Blodgett, Lisa Egede, Alicia DeVrio, and Myra Cheng. 2026. Ai automatons: Ai systems intended to imitate humans. In *Proceedings of the 2026 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '26, page 5507–5545, New York, NY, USA. Association for Computing Machinery.

Daniele Potertì, Andrea Seveso, and Fabio Mercorio. 2025. Can role vectors affect LLM behaviour? In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 17735–17747, Suzhou, China. Association for Computational Linguistics.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

Vinay Samuel, Henry Peng Zou, Yue Zhou, Shreyas Chaudhari, Ashwin Kalyan, Tanmay Rajpurohit, Ameet Deshpande, Karthik R Narasimhan, and Vishvak Murahari. 2025. PersonaGym: Evaluating persona agents and LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 6999–7022, Suzhou, China. Association for Computational Linguistics.

Preethi Seshadri, Samuel Cahyawijaya, Ayomide Odumakinde, Sameer Singh, and Seraphina Goldfarb-Tarrant. 2026. Lost in simulation: Llm-simulated users are unreliable proxies for human users in agentic evaluations. *arXiv preprint arXiv:2601.17087*.

Rusheb Shah, Quentin Feuillade-Montixi, Soroush Pour, Arush Tagade, Stephen Casper, and Javier Rando. 2023. Scalable and transferable black-box jailbreaks for language models via persona modulation. *Preprint*, arXiv:2311.03348.

Jeonghoon Shim, Woojung Song, Cheyon Jin, Seungwon KooK, and Yohan Jo. 2026. Non-collaborative user simulators for tool agents. In *International Conference on Learning Representations*, volume 2026, pages 64486–64534.

Nicholas Sofroniew, Isaac Kauvar, William Saunders, Runjin Chen, Tom Henighan, Sasha Hydrie, Craig Citro, Adam Pearce, Julius Tarng, Wes Gurnee, Joshua Batson, Sam Zimmerman, Kelley Rivoire, Kyle Fish, Chris Olah, and Jack Lindsey. 2026. Emotion concepts and their function in a large language model. *Transformer Circuits Thread*. Also available as arXiv:2604.07729, https://arxiv.org/abs/2604.07729.

Lihao Sun, Lewen Yan, Xiaoya Lu, Andrew Lee, Jie Zhang, and Jing Shao. 2026. Valence-arousal subspace in llms: Circular emotion geometry and multi-behavioral control. *Preprint*, arXiv:2604.03147.

Daniel Tan, David Chanin, Aengus Lynch, Brooks Paige, Dimitrios Kanoulas, Adrià Garriga-Alonso, and Robert Kirk. 2024. Analysing the generalisation and reliability of steering vectors. In *Advances in Neural Information Processing Systems*, volume 37, pages 139179–139212. Curran Associates, Inc.

Ningzhi Tang, Chaoran Chen, Zihan Fang, Gelei Xu, Maria Dhakal, Yiyu Shi, Collin McMillan, Yu Huang, and Toby Jia-Jun Li. 2026. Programming by chat: A large-scale behavioral analysis of 11,579 real-world ai-assisted ide sessions. *Preprint*, arXiv:2604.00436.

Jiessie Tie, Bingsheng Yao, Tianshi Li, Hongbo Fang, Syed Ishtiaque Ahmed, Dakuo Wang, and Shurui Zhou. 2026. "should i give up now?" investigating llm pitfalls in software engineering. *ACM Trans. Softw. Eng. Methodol.*

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Miles Wang, Tom Dupre la Tour, Olivia Watkins, Aleksandar Makelov, Ryan Chi, Samuel Miserendino, Jeffrey Wang, Achyuta Rajaram, Johannes Heidecke, Tejal Patwardhan, and Daniel Mossing. 2026. Persona features control emergent misalignment. In *International Conference on Learning Representations*, volume 2026, pages 48836–48885.

Tianlong Wang, Xianfeng Jiao, Yinghao Zhu, Zhongzhi Chen, Yifan He, Xu Chu, Junyi Gao, Yasha Wang, and Liantao Ma. 2025. Adaptive activation steering: A tuning-free llm truthfulness improvement method for diverse hallucinations categories. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, pages 2562—-2578, New York, NY, USA. Association for Computing Machinery.

Shirley Wu, Evelyn Choi, Arpandeep Khatua, Zhanghan Wang, Joy He-Yueya, Tharindu Cyril Weerasooriya, Wei Wei, Diyi Yang, Jure Leskovec, and James Zou. 2026. Humanlm: Simulating users with state alignment beats response imitation. *arXiv preprint arXiv:2603.03303*.

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

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. 2024. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. In *International Conference on Learning Representations*, volume 2024, pages 22225–22257.

Yujia Zheng, Zhuokai Zhao, Zijian Li, Yaqi Xie, Mingze Gao, Lizhu Zhang, and Kun Zhang. 2025. Thought communication in multiagent collaboration. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 123389–123418. Curran Associates, Inc.

Xuhui Zhou, Weiwei Sun, Qianou Ma, Yiqing Xie, Jiarui Liu, Weihua Du, Sean Welleck, Yiming Yang, Graham Neubig, Sherry Tongshuang Wu, and Maarten Sap. 2026. Mind the sim2real gap in user simulation for agentic tasks. *arXiv preprint arXiv:2603.11245*.

Ming Zhu, Juntao Tan, Rithesh Murthy, Jielin Qiu, Liangwei Yang, Wenting Zhao, Silvio Savarese, Shelby Heinecke, and Huan Wang. 2026a. Realusersim: Bridging the reality gap in agent benchmarking via grounded user simulation. *Preprint*, arXiv:2605.20204.

Ziyi Zhu, Olivier Tieleman, Caitlin A. Stamatis, Luka Smyth, Thomas D. Hull, Daniel R. Cahn, Jinghong Chen, and Matteo Malgaroli. 2026b. Dial: Direct iterative adversarial learning for realistic multi-turn dialogue simulation. *Preprint*, arXiv:2512.20773.

A Extracting the User Role Vector

This appendix provides supplementary details for the extraction pipeline described in §2.1, §2.2, and §2.3.

A.1 Dialogue Sampling

A.1.1 Dialogue Source and Filtering

We start from the LMSYS-Chat-1M source and apply two simple checks: each dialogue must have between 2 and 50 turns, and every turn must have a proper role label and non-empty content. Of the one million source dialogues, 982,581 pass both checks and 17,419 are dropped. Most of the dropped dialogues have at least one empty message; a smaller portion exceed the turn limit (Table 8). From the accepted pool, we draw a uniform random sample of 100,000 dialogues for downstream labeling.

Filter setting / outcome	Value
Minimum turn count	2
Maximum turn count	50
Random sample size	100,000
Source rows scanned	1,000,000
Rows accepted	982,581
Rows dropped (empty message)	15,879
Rows dropped (above max turns)	1,540

Table 8: LMSYS-Chat-1M filtering settings and outcomes.

A.1.2 Topic Labeling

Each filtered dialogue is labeled with one of 24 conversation-type categories using the prompt in Appendix E.1, adapted from Chatterji et al. (2025) and submitted to GPT 5 Mini with minimal reasoning effort in a zero-shot setting. The resulting category distribution (Table 9) is concentrated in information-seeking and programming requests: the three most common labels account for about 41% of valid labels. Of the sampled dialogues, 39 return an invalid response while labeling and are excluded from later stages.

A.2 Role-Specific Reflection

A.2.1 Reflection Generation

Each reflection consists of a system-prompt intro, the dialogue text, and a fixed reflection question. The full template, the three intros, and the reflection question are shown in Appendix E.1. For each (dialogue, role) pair, the model generates three reflections, yielding six reflections per dialogue.

Category	Count	Percent
Specific info	17,453	17.46%
Computer programming	13,032	13.04%
Edit or critique provided text	10,503	10.51%
Greetings and chitchat	8,382	8.39%
Write fiction	6,532	6.53%
Tutoring or teaching	5,707	5.71%
How-to advice	5,547	5.55%
Other	5,515	5.52%
Personal writing or communication	4,161	4.16%
Asking about the model	4,015	4.02%
Mathematical calculation	3,105	3.11%
Games and role play	2,641	2.64%
Argument or summary generation	2,470	2.47%
Creative ideation	2,145	2.15%
Translation	1,932	1.93%
Unclear	1,923	1.92%
Health, fitness, beauty, or self-care	1,190	1.19%
Relationships and personal reflection	1,130	1.13%
Purchasable products	625	0.63%
Create an image	583	0.58%
Data analysis	576	0.58%
Cooking and recipes	510	0.51%
Analyze an image	222	0.22%
Generate or retrieve other media	62	0.06%

Table 9: Distribution of GPT 5 Mini topic labels across 99,961 LMSYS-Chat-1M dialogues.

Reflection example. Figure 9 shows the full dialogue that the model saw together with the user and assistant reflections it generated, expanding the truncated excerpt from Figure 3.

A.2.2 Reflection Validation

Each generated reflection is scored by GPT 5 Mini using the two-message prompt in Appendix E.1. Only reflections judged to represent the assigned perspective strongly or weakly are retained for activation extraction.

A.3 Difference-in-Means Computation

A.3.1 Activation Extraction and Aggregation

For each retained reflection, we record the activation at the first response token for every model layer. User and assistant reflections are paired by dialogue and prompt variant, and the user-minus-assistant differences are averaged across retained pairs. The resulting direction is normalized before steering, as described in §2.3.

B What Does the User Role Vector Represent?

This appendix provides supplementary details for the characterization experiments in §3.1–§3.3.

Dialogue topic. Composing a congratulation message for a graduate.
Model. Qwen 3.5 9B *Prompt variant.* Role simulation

Dialogue.
User: How would you congratulate a scholar?
Assistant: Congratulations can take many forms, depending on the occasion and your relationship with the scholar. You might send a handwritten letter, post a public message on social media, or organize a small celebration. The most important thing is to acknowledge their hard work and to express genuine pride in what they have achieved.
User: Can you write me a sample message for a graduate?
Assistant: Dear [Name], congratulations on your graduation! Your hard work, dedication, and perseverance have brought you to this remarkable moment, and I am so proud of everything you have accomplished. As you step into this next chapter, I hope you carry forward the same curiosity and determination that brought you here. Wishing you every success in the journey ahead.

User reflection. Reflecting on this exchange, I realize that congratulating a scholar or graduate is less about the specific gift or event and more about the genuine recognition of their hard work and perseverance. The sample message provided effectively captures the emotional weight of the moment by balancing pride in their past achievements with excitement for their future, which is exactly what makes such acknowledgments meaningful. It serves as a reminder that whether through a handwritten letter, a social media post, or a personal celebration, the core intent should always be to sincerely validate their dedication and offer heartfelt support as they move forward.

Assistant reflection. Reflecting on this interaction, I recognize that while the initial query about congratulating a scholar was broad, the user's follow-up request for a specific graduate sample revealed a need for concrete, emotionally resonant language rather than just general strategies. By providing a structured, warm, and customizable template, I aimed to bridge the gap between abstract advice and practical application, ensuring the user could easily adapt the message to their specific relationship with the graduate. This approach highlights the importance of tailoring responses to the user's evolving needs, moving from high-level concepts to actionable examples that convey genuine pride and support.

Figure 9: Full dialogue and role-specific reflections for the example in Figure 3. Qwen 3.5 9B generates the two reflections from the user and assistant perspectives using the role-simulation prompt variant.

B.1 Effect on Communication Style

B.1.1 Task Goal and Request Generation

The communication-style experiment in §3.1 steers the simulator while it generates a request message conditioned on a task goal. We generate diverse task goals using the prompt in Appendix E.2 and, for each goal, prompt the steered and unsteered simulators to produce a request message using the prompt in Appendix E.2. The prompted baseline uses the user-like request prompt in Appendix E.2.

B.1.2 Steering Parameter Selection

We assess the steering range by sweeping layers 10–14 and $\alpha \in \{0.1, 0.2, 0.3, 0.4\}$ on Qwen 3.5 9B, using 100 task goals per condition and averaging five judge evaluations per request. Above $\alpha = 0.3$, the simulator produces extremely short messages that largely repeat the task goal. Table 10 reports the communication-style score and ROUGE-L precision against the task goal, where higher precision indicates greater reuse of goal text. Both values rise with steering strength: at layer 11, increasing α from 0.1 to 0.3 raises the judge score from 1.920 to 3.385 but also raises precision from .295 to .625; moving to $\alpha = 0.4$ yields a smaller score gain while precision reaches .709. Reliability also differs across models: at layer 11, Granite 3.3 8B's repeated-token error rate rises from 0% at $\alpha \leq 0.2$ to 13% at $\alpha = 0.3$ and 91% at $\alpha = 0.4$. We therefore use $\alpha \in \{0.1, 0.2, 0.3\}$ to span low-to-high intervention levels before this degradation becomes severe.

Layer	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$
10	1.780 / .271	2.224 / .380	3.003 / .540	3.547 / .647
11	1.920 / .295	2.802 / .494	3.385 / .625	3.645 / .709
12	1.953 / .323	3.126 / .553	3.560 / .626	3.880 / .705
13	1.959 / .304	3.050 / .513	3.553 / .578	3.827 / .621
14	1.683 / .243	2.098 / .292	2.695 / .372	3.417 / .465

Table 10: Communication-style scores and task-goal overlap across steering parameters (judge score / ROUGE-L precision). Stronger steering improves user-likeness but also increases reuse of task-goal text.

B.1.3 Task-Goal Generalization

Setup. We repeat the request-generation experiment with three external user request datasets: MultiWOZ 2.0 and the 2021 and 2022 TREC Health Misinformation topic sets (Budzianowski et al., 2018; Clarke et al., 2021, 2022). From the MultiWOZ test split, we randomly sample 100 of 1,000 eligible dialogue goals and instruct the simulator to reveal each goal gradually; the recorded first utterance and dialogue are withheld. For TREC Health, we use all 50 topics from each year and provide only the topic query and background, withholding the gold question. For each source, Qwen 3.5 9B generates one message at layer 11 under $\alpha \in \{0, 0.1, 0.2, 0.3\}$ with deterministic decoding. Each generation is scored once by the

communication-style judge in §B.1.4. Table 11 reports the resulting scores; the generation prompts appear in §E.2.

Source	α	Overall	Brevity	Informality	Information pacing
MultiWOZ	0.0	1.467	1.510	1.590	1.300
	0.1	1.777	1.890	1.800	1.640
	0.2	2.077	2.120	1.940	2.170
	0.3	2.847	2.840	2.620	3.080
TREC	0.0	2.550	3.260	1.540	2.850
	0.1	2.987	3.670	1.890	3.400
	0.2	3.090	3.730	2.050	3.490
	0.3	3.533	4.170	2.300	4.130

Table 11: User-likeness scores across steering strengths for MultiWOZ and TREC task goals. Scores increase monotonically with steering strength on both datasets.

Results. The effect replicates on human-authored goals in MultiWOZ 2.0: the overall score rises monotonically from 1.47 to 2.85, and all three dimensions improve with steering strength. Information pacing shows the largest change (1.30 to 3.08), indicating that steering helps the simulator avoid revealing the full multi-turn goal in its opening message.

The same monotonic pattern holds for health information needs in TREC, with the overall score increasing from 2.55 to 3.53. Informality remains the lowest dimension (1.54 to 2.30), while brevity and information pacing rise to 4.17 and 4.13, respectively. Although the goal domain affects the request content, the direction of the steering effect remains consistent.

B.1.4 Communication-Style Evaluation

We use GPT 5 Mini as the judge for scoring user-likeness along three dimensions: brevity, informality, and information pacing. The judge scores each dimension on a 1–5 scale, with higher scores indicating more user-like style. The full prompt is shown in Appendix E.2.

B.1.5 LLM-as-Judge Validation

We validate that the judge separates real user messages from strong LLM-generated approximations of the same message.

Setup. We sample 1,000 English WildChat conversations with 2–10 turns and at most 1,000 words per message. For each conversation, we hide the final user message and prompt GPT 5 to reconstruct the most likely next user message from the prior dialogue. The real message and the reconstructed message are then scored independently by the judge. Two samples are excluded due to generation errors.

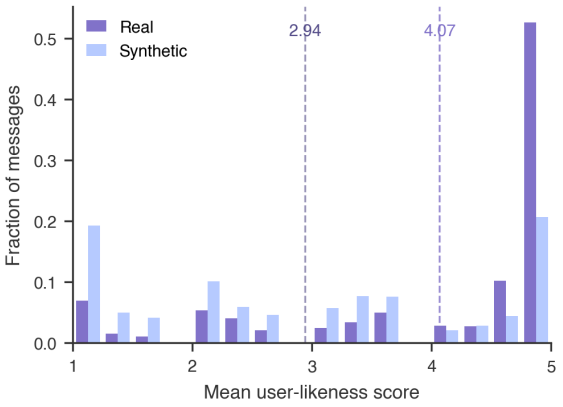


Figure 10: Distribution of user-likeness scores for 998 paired real and reconstructed WildChat messages. Real messages score higher; dashed lines mark the group means.

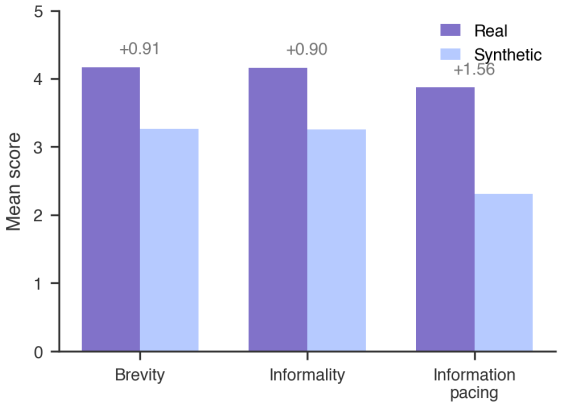


Figure 11: User-likeness scores by dimension for 998 paired real and reconstructed WildChat messages. Real messages score higher on all dimensions, with the largest gap in information pacing; labels show real-minus-reconstructed mean differences.

Results. On the remaining 998 paired samples, real user messages receive a mean user-likeness score of 4.07 against 2.94 for synthetic messages, a mean gap of 1.13 (95% CI: 1.02–1.22) (Figure 10). Real messages win the pairwise comparison in 63.2% of cases, tie in 24.9%, and lose in 11.8%. The gap is consistent across all three dimensions: brevity (+0.91), informality (+0.90), and information pacing (+1.56), with the largest separation on information pacing (Figure 11). The result confirms that the metric distinguishes real user messages from strong LLM-generated approximations of the same message.

α	Rater	Brevity	Informality	Information pacing	Overall
0.0	Human	2.29	2.18	2.23	2.23
	LLM judge	1.80	1.02	2.08	1.63
0.1	Human	2.72	2.60	2.54	2.62
	LLM judge	2.26	1.20	2.20	1.89
0.2	Human	3.51	3.00	3.36	3.29
	LLM judge	3.28	1.28	3.72	2.76
0.3	Human	4.15	3.32	3.83	3.77
	LLM judge	4.18	1.50	4.50	3.39

Table 12: Human and LLM-judge communication-style scores across steering strengths. Both assign higher scores as steering increases.

B.1.6 Human Validation

We further test whether the judge agrees with human ratings of generated requests.

Setup. We sample 200 requests from 50 task goals across four Qwen 3.5 9B conditions at layer 11, with $\alpha \in \{0, 0.1, 0.2, 0.3\}$. Thirty participants each rate one set of 40 requests on the same 1–5 brevity, informality, and information-pacing scales used by the judge. The five sets are disjoint, and six participants rate each request, yielding 1,200 ratings. We measure human–judge alignment using Spearman correlation after averaging the six ratings for each request. Tie-corrected Kendall's W measures agreement among individual human raters.

Participants were recruited through Prolific for a 20-minute study and paid £5.40 each. Figure 12 shows the full instructions and consent notice given to participants. No personal information were collected, and payments were administered through pseudonymous Prolific participant IDs.

Results. The mean human score rises monotonically from 2.23 without steering to 3.77 at $\alpha = 0.3$, matching the judge's increase from 1.63 to 3.39 (Table 12). Human and judge scores correlate strongly overall ($\rho = 0.800$), with dimension-level correlations of 0.798 for brevity, 0.439 for informality, and 0.754 for information pacing. Human raters also show substantial agreement overall ($W = 0.700$), with $W = 0.800$ for brevity, 0.432 for informality, and 0.579 for information pacing.

B.1.7 Activation Correlation with Communication Style

Setup. We use the same 100 task goals as the communication-style experiment in §3.1 and generate one unsteered request per goal with Qwen 3.5 9B. The same GPT 5 Mini judge scores each request for brevity, informality, and information pacing (§B.1.4). We teacher-force each request without steering, project its layer-11 hidden states onto the normalized user-minus-assistant direction, and average the projections across response tokens.

Results. Mean user-role activation is positively correlated with the overall communication-style score (Pearson's $r = .260$; Spearman's $\rho = .313$; Table 13). The rank association is strongest for brevity ($\rho = .322$). To account for response length, we residualize the activation and style scores against log token count for Pearson correlation, and residualize their ranks against ranked token count for Spearman correlation. After this control, the overall association becomes slightly

User Simulator Evaluation

In this study, you will read 40 request messages generated by a user simulator (a simulator designed to mimic user behavior) and rate how human-like each message feels. There are no right or wrong answers. Please assign scores based on the rubric provided.

Information and consent

• Your Prolific participant ID is collected to manage participation and payment.
• No name, email address, or other directly identifying information is requested.
• Only de-identified ratings may be released for research purposes.
• Participation is voluntary. You may stop before submitting the study.

[ ] I have read the information above and consent to participate.

How to evaluate a message

Read the generated request, then select one score for each of the three style dimensions. Use the descriptions and examples shown with each rubric.

Example request: "best headphones for flights?"
Example ratings: Brevity: 5 (Short); Informality: 5 (Casual); Information pacing: 5 (Concise).
This short, casual request gives only the immediate need, so it receives a high score on all three dimensions.

Brevity. Focus on word count and short-turn behavior.
1 (Long): Long, complete, polished, or over-explained. Example: "Could you provide a detailed and well-structured comparison of three noise-canceling headphones, including comfort, battery life, price, and suitability for travel?"
5 (Short): Very short, direct, fragment-like, or only says what is needed. Example: "best headphones for flights?"

Informality. Focus on surface style, formality, and conversational roughness.
1 (Formal): Formal, careful, service-like, or written in polished complete sentences. Example: "Could you please assist me in drafting an appropriate email declining this invitation?"
5 (Casual): Casual, blunt, uneven, ordinary, or written like quick everyday conversation. Example: "how do i say no to this invite without sounding rude"

Information pacing. Focus on whether the message front-loads information or withholds details.
1 (Dense): Gives the full context, identifiers, constraints, and desired outcome upfront. Example: "I need to return order #48291, a black jacket bought last Tuesday, because the zipper broke; please write a message asking for a refund or replacement."
5 (Concise): Gives only the immediate need, leaves details implicit, or would require follow-up. Example: "can i return this?"

Figure 12: Instructions and consent notice shown to participants in the human-validation study. Participants had to select the consent checkbox before beginning.

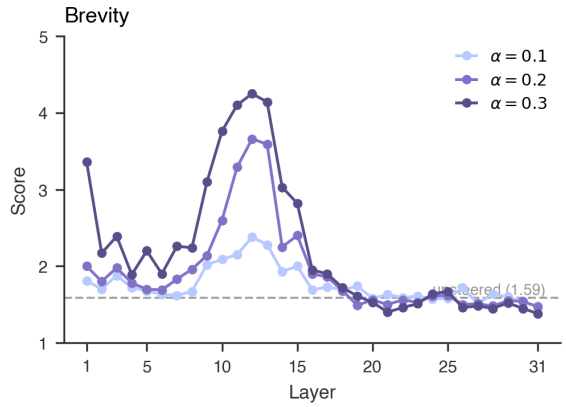


Figure 13: Brevity scores across steering layers and strengths. Scores peak at layers 11–13.

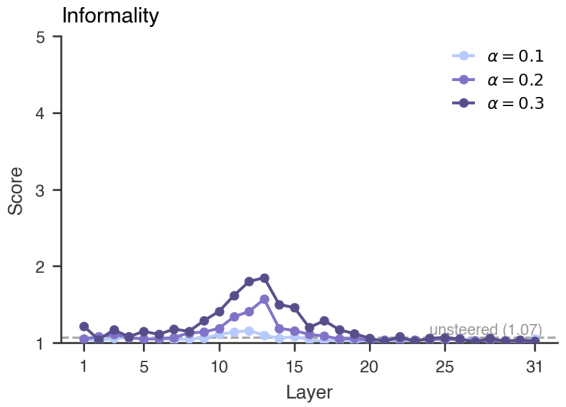


Figure 14: Informality scores across steering layers and strengths. Scores rise sharply at layers 11–13 and saturate near the top of the scale.

stronger ($r = .292$; $\rho = .337$); in this setup, response length is associated with communication-style scores but only weakly with user-role activation.

	Correlation		Length-controlled	
Metric	r	ρ	r	ρ
Overall	.260	.313	.292	.337
Brevity	.208	.322	.247	.379
Informality	.224	.241	.223	.233
Information pacing	.219	.244	.219	.239

Table 13: Correlation between user-role activation and communication-style scores. Length-controlled columns report partial correlations; all associations remain positive.

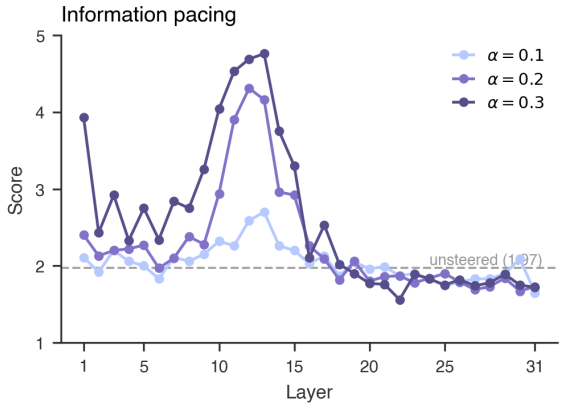


Figure 15: Information-pacing scores across steering layers and strengths. This dimension shows the largest shift, peaking near 4.7 at $\alpha = 0.3$ in layers 11–13.

B.1.8 Layer-Wise Steering Effects

We sweep the injection layer $\ell \in [1, 31]$ and the steering coefficient $\alpha \in \{0.1, 0.2, 0.3\}$, with 100 sampled requests per (layer, α) cell scored by the LLM-as-judge in §B.1.4. The unsteered baseline (alpha-zero) is layer-independent because $\alpha = 0$ zeros the injected vector; we report the empirical baseline, $\bar{s} = 1.51$.

Per-dimension breakdown. Figures 13–15 report the per-dimension scores across layers. All three dimensions respond strongly in the layer 11–13 band. Information pacing has the largest absolute shift (peaking near 4.7), informality reaches the top of the scale earliest and saturates, and brevity tracks the aggregate score most closely. Layers 19 and beyond receive little or no benefit from stronger α on any dimension.

Steering coefficient. Within the layer 11–13 peak, $\alpha = 0.3$ outperforms smaller coefficients without showing saturation: the gap between $\alpha = 0.2$ and $\alpha = 0.3$ is 0.45–0.57 points at layers 11, 12, and 13. Outside this band, the response is brittle: at layer 16 and later, $\alpha = 0.3$ often scores no better than $\alpha = 0.1$, and at layers 19–31 it is sometimes worse. We treat layer and α as a coupled choice rather than independent hyperparameters: stronger α is only useful within the mid-layer window.

B.1.9 Cross-Model Steering Effects

We repeat the layer sweep on Granite 3.3 8B and Llama 3.1 8B using the same request-generation setting, with first-token role vectors applied to all generated tokens. Both models show that user-role steering has its largest effect in early-to-mid layers, although the exact peak differs by architecture. For Granite, the more stable gains appear around layer 12 and layers 19–23 (Figure 16); for Llama,

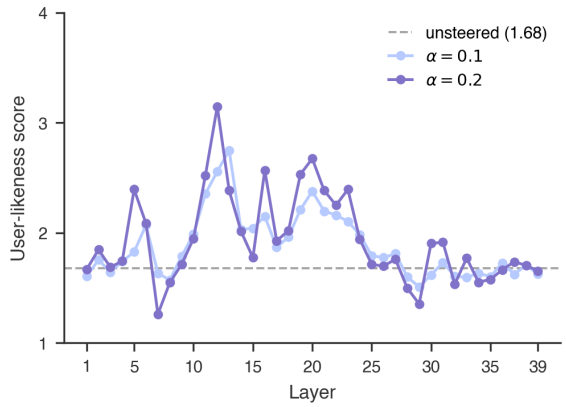


Figure 16: User-likeness scores across steering layers and strengths for Granite 3.3 8B. Effects are strongest near layer 12 and layers 19–23. The $\alpha = 0.3$ condition is omitted because it produces degenerate outputs.

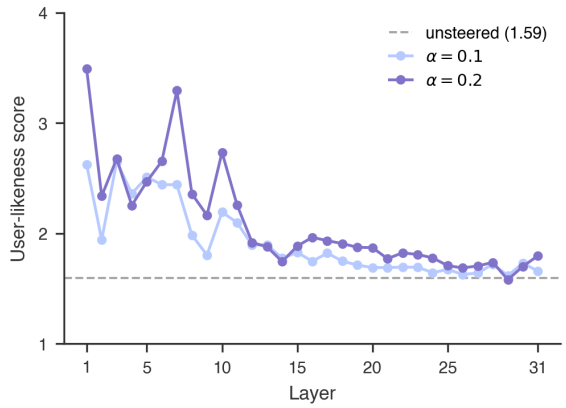


Figure 17: User-likeness scores across steering layers and strengths for Llama 3.1 8B. Effects are strongest in layers 1–10 and decay after layer 12. The $\alpha = 0.3$ condition is omitted because it produces degenerate outputs.

the strongest gains appear earlier, mostly in layers 1–10 (Figure 17). This supports the main result that user-like message style is most sensitive to earlier parts of the model rather than late layers.

At the same time, reliability differs across models. At $\alpha = 0.3$, both Granite and Llama sometimes produce degenerate outputs, including repeated words and meaningless tokens. We therefore exclude these unstable settings from the summary table and treat the cross-model result as qualitative evidence: the layer-localized effect is coherent across models, but the reliable steering strength is model-dependent.

B.1.10 Layer Structure of the User Role Direction

The layer-wise steering effect is not simply a property of one isolated injection layer. In Qwen 3.5 9B, the user role direction forms local bands in the layer-to-layer cosine matrix (Figure 18). Adjacent layers are generally similar, and layers 6–20 form a broad mid-layer cluster, with a slight concentration around layers 11–13. This structure helps explain why the main effect in §3.1 appears around this broader part of the network rather than at a single isolated layer. Steering in this range appears to act on a user role direction that is well matched to the layer-local representation, rather than on a direction that is equally effective across all layers.

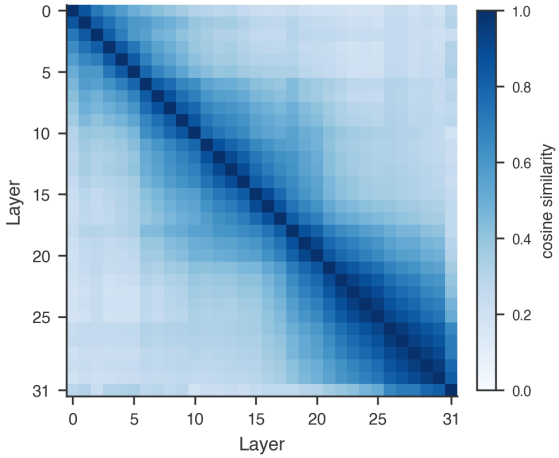


Figure 18: Layer-to-layer cosine similarity of the user-role direction in Qwen 3.5 9B. Layers 6–20 form a broad mid-layer cluster.

The cross-model matrices show the same principle with slightly different boundaries. Granite 3.3 8B separates into a broad early block and a later block, with a transition around layers 12–16; its strongest steering effects occur near this transition and again around layers 19–23 (Figure 19). Llama 3.1 8B shows a more fragmented early structure, with weaker continuity across layers 8–13 and a more stable later block (Figure 20). Together with the layer-sweep results in §B.1.9, these matrices suggest that user role steering is effective on specific layer-local representation of the user role. The relevant layer range is therefore conceptually coherent, where user-like message style is more sensitive to earlier or transitional parts of the network than to later layers.

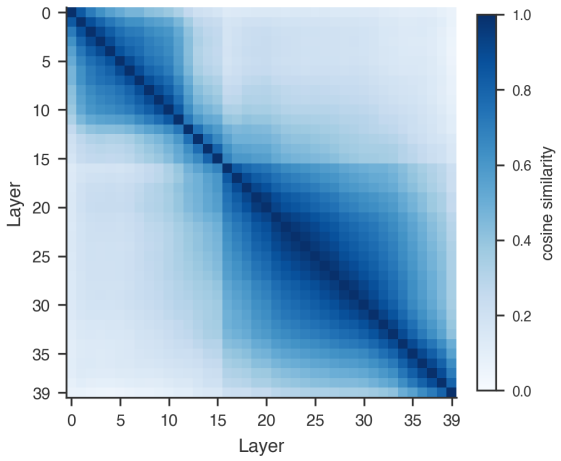


Figure 19: Layer-to-layer cosine similarity of the user-role direction in Granite 3.3 8B. The matrix separates into early and late blocks, with a transition around layers 12–16.

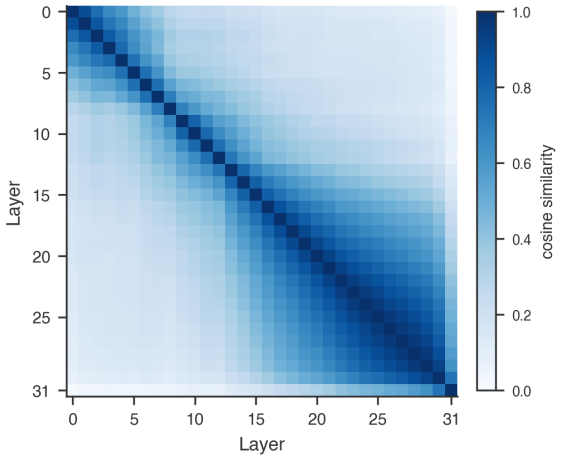


Figure 20: Layer-to-layer cosine similarity of the user-role direction in Llama 3.1 8B. Early layers are fragmented, while later layers form a more stable block.

B.1.11 Bidirectional Steering

The directionality control in §3.1 is a coarse comparison between user-direction and assistant-direction steering at $\alpha = 0.3$. To check that the effect is smooth and monotonic, we sweep the signed steering coefficient $\alpha \in \{-0.3, -0.2, -0.1, 0.0, 0.1, 0.2, 0.3\}$ at layer 11, where positive α steers along the user direction and negative α steers along the assistant direction. Each cell uses the same 100 task goals and the same judge as the main experiment.

The user-likeness score increases monotonically with α across the full range (Figure 21). The unsteered baseline sits at 1.53. Moving toward the user direction lifts the score to 1.85 at $\alpha = 0.1$, 2.84 at $\alpha = 0.2$, and 3.42 at $\alpha = 0.3$. Moving toward the assistant direction drops the score to 1.21, 1.13, and 1.05 at $\alpha = -0.1, -0.2$, and -0.3 respectively, approaching the floor of the 1–5 scale. The same vector therefore controls user-likeness in both directions, with no inflection or saturation visible on the user side within the swept range.

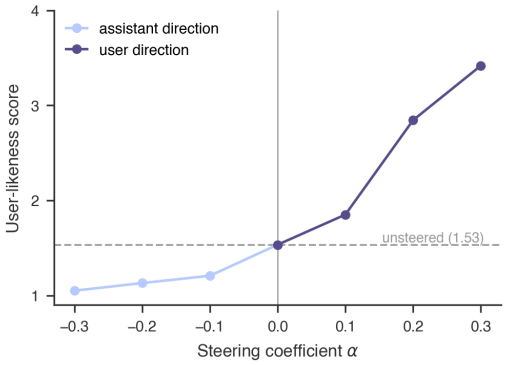


Figure 21: User-likeness scores under bidirectional steering at layer 11. Scores increase monotonically from assistant-direction (negative α) to user-direction (positive α) steering.

B.2 Effect on Interaction Behavior

B.2.1 Disengagement Taxonomy

The interaction-behavior experiment in §3.2 samples dialogues stratified by a disengagement taxonomy derived from real user-assistant interaction logs.

Source data. Publicly available user-assistant interaction logs are collected from GitHub through the GitHub Search API. The sources include coding-agent frameworks such as SpecStory, Aider, and Claude Code. Of the 33,033 collected logs, 9,465 are excluded for empty sessions, incomplete conversations (e.g., a single message), or corrupted files. Logs with 2 to 50 turns are retained, yielding 16,077 interaction logs for the taxonomy analysis.

Taxonomy construction. Gemini 2.5 Flash develops an initial codebook of user disengagement triggers by iterating over each dialogue. For each dialogue, the model either identifies a new disengagement reason or merges it into an existing code. The first pass produces 61 unique disengagement triggers. Two researchers then revisit the codebook and merge codes by definition and sampled examples, producing eight concrete disengagement types (Table 14). A ninth *Unknown* category captures end-of-conversation events that cannot be attributed to model behavior; it is retained in the taxonomy

Normal completion The user's goal is fully addressed and the conversation ends naturally, with no further interaction required.
Forgetting The model fails to incorporate prior context or instructions, losing track of constraints, earlier instructions, or dialogue history.
Technical failure The model fails to produce a correct or functional output, such as code that errors, an incomplete generation, or an execution failure.
Misinterpretation The model misunderstands the user's intent, requirements, or phrasing, yielding a response that is off-target or irrelevant.
Expectation mismatch The model's output is technically plausible but diverges from the user's assumed workflow, conventions, or mental model of the task.
Unsolicited action The model performs actions beyond what is explicitly requested, introducing edits, additions, or suggestions outside the user's scope.
Refusal The model declines or deflects a reasonable request, defers unnecessarily to the user, or adds friction through excessive clarification or caveats.
Hallucination The model produces factually incorrect or fabricated content, including false claims, fabricated references, or unwarranted confidence.
Unknown The conversation ends for reasons unrelated to model behavior, such as user-side factors or external circumstances. Excluded from sampling.

Table 14: Taxonomy of user disengagement triggers derived from public interaction logs. The first eight categories are used for stratified sampling; *Unknown* is excluded because it is not attributable to model behavior.

for completeness but excluded from sampling in §3.2.

B.2.2 Disengagement Prediction

The disengagement experiment evaluates whether a simulator can predict when the original user would stop a conversation. Each dialogue has eight user–assistant turns and an annotated disengagement type from the taxonomy above. At each prefix turn, the model sees the dialogue up to the current assistant response and chooses whether the user would continue or disengage. We run two decisions per sample with the option order swapped, so each completed 735-dialogue run contributes 1,470 balanced decisions. The predicted disengagement turn is the first turn where the model chooses disengagement; otherwise, the user is treated as continuing through the final turn.

Condition	Normal completion	Disrupted ending
Qwen 3.5 9B (Unsteered)	27.0%	51.8%
+ Steered ($\alpha = 0.1$)	34.0%	55.0%
+ Steered ($\alpha = 0.2$)	41.0%	58.2%
+ Steered ($\alpha = 0.3$)	56.0%	68.5%
GPT 5.4	23.0%	26.5%
Gemini 3 Flash	64.5%	58.6%

Table 15: Predicted disengagement rates for normal and disrupted endings. Stronger steering increases both rates; disrupted endings aggregate the seven model-failure categories in Table 14.

All conditions use a prompt without a user profile or a request for reasoning (Appendix E.3). For Qwen 3.5 9B, the user-role vector is injected at layer 11 with $\alpha \in \{0, 0.1, 0.2, 0.3\}$ and applied to all generated tokens. GPT 5.4 and Gemini 3 Flash are evaluated with the same decision prompt but without activation steering. Exact match measures whether the predicted turn equals the annotated disengagement turn. Average distance is the absolute turn error and excludes final-continue trials. Table 15 reports the same predictions grouped by normal completion versus disrupted endings.

B.2.3 Cross-Model Disengagement Prediction

We repeat the disengagement-prediction experiment with Llama 3.1 8B at layer 7, using the same 735 dialogues and two option-order decisions per dialogue. The disengagement rate rises monotonically from 15.3% without steering to 52.7% at $\alpha = 0.3$, reproducing the Qwen trend (Table 16). Exact match instead peaks at $\alpha = 0.1$ (2.6%), while average distance increases overall from 3.86 to 5.81 turns. As with Qwen 3.5 9B, stronger steering makes Llama more likely to disengage but does not reliably improve when it disengages.

α	Exact match	Average distance	Disengagement rate
0.0	23/1,470 (1.6%)	3.858	15.3%
0.1	38/1,470 (2.6%)	4.366	33.1%
0.2	17/1,470 (1.2%)	5.813	45.9%
0.3	27/1,470 (1.8%)	5.811	52.7%

Table 16: Llama 3.1 8B disengagement prediction across steering strengths at layer 7. Steering increases the disengagement rate but not timing accuracy.

cooperative Willing to work with others, coordinate actions, and support shared progress.
helpful Inclined to provide assistance, useful information, or support when needed.
compliant Tending to accept requests, rules, instructions, or external direction with little resistance.
sycophantic Excessively agreeable, approving, or flattering toward others, especially authority figures.
polite Courteous, respectful, and socially considerate in communication and behavior.
patient Able to tolerate delays, difficulties, repetition, or frustration without becoming visibly upset.
structured Organized, orderly, and systematic in presenting thoughts, actions, or information.
explicit Clear, direct, and unambiguous in expressing intentions, needs, preferences, or information.
complete Thorough and comprehensive, with few missing details or unresolved gaps.
verbose Using more words, details, or explanation than may be necessary.
fluent Smooth, polished, and natural in expression, with few disruptions, errors, or hesitations.
generic Lacking distinctive personal style, specificity, or unusual characteristics.
neutral Emotionally even, restrained, or impartial, without strong affective expression.
consistent Stable and reliable across situations, with little contradiction or unexpected variation.
rational Logical, deliberate, and reason-based in judgment, decision-making, or behavior.
goal_aligned Focused on pursuing stated aims and avoiding distractions or irrelevant deviations.
accommodating Willing to adjust, compromise, or adapt to others' needs, constraints, or preferences.
deferential Respectful or submissive toward others' authority, expertise, or judgment.
explanatory Inclined to clarify, justify, or elaborate on reasoning, motives, or context.
selfless Prioritizing others' needs, goals, or success over one's own convenience or interests.

Table 17: Definitions of the 20 assistant-like traits used to extract trait directions.

B.3 Relationship to the Assistant Bias

B.3.1 Assistant-Trait Direction Extraction

The analysis in §3.3 compares the user role vector against a set of assistant-trait directions. Table 17 lists the 20 traits used in this analysis.

Data generation. For each trait, Gemini 3 Flash generates five positive/negative instruction pairs and 40 neutral questions. The positive instruction asks the model to exhibit the target trait, while the negative instruction asks the model to avoid it and express the opposing behavior. The questions are written to elicit the trait without naming it directly.

Response generation and scoring. Qwen 3.5 9B generates two sampled responses for each instruction–question pair. With five instruction pairs, two polarities, 40 questions, and two repeats, this yields 800 candidate utterances per trait and 16,000 candidate utterances across the 20 traits. GPT 5 Mini then scores each response from 0 to 100 for how strongly it expresses the target trait, using a trait-specific evaluation prompt generated with the data. The scoring prompt focuses only on the target trait and treats refusals separately.

Vector extraction. For each trait, the pipeline selects positive and negative responses with a score gap of at least 50 points, requiring at least 10 selected pairs per trait. In the final run, this filter retains 6,370 contrast pairs. At layer 11, it extracts hidden states from Qwen 3.5 9B and computes the trait direction as the difference between the mean positive activation and the mean negative activation. The resulting direction is compared with the user role vector using cosine similarity.

C What Does the User Role Vector Reveal About Simulation?

C.1 Steering Effects on Simulation Realism

C.1.1 SimulatorArena Evaluation Setup

SimulatorArena uses separate judge prompts for similarity evaluation. Appendix E.4 show the interaction-style and writing-style evaluation prompts. Table 18 lists the feature dimensions supplied to the evaluator.

C.1.2 Profile Prompt Example

Profile-prompt conditions provide attributes extracted from the corresponding human dialogue. The following excerpt shows four dimensions from an interaction-style profile; the complete condition includes all 16 interaction-style dimensions listed in Table 18.

Answer thoroughness. The user gives brief, minimal answers that focus on essential outputs without elaborating on intermediate reasoning.
Clarification seeking. The user occasionally seeks clarification when unsure about a problem-solving step.
Confidence level. The user frequently expresses uncertainty but persists by testing values and engaging with the problem.
Seeking guidance. The user seeks guidance and validation at nearly every step and rarely proceeds independently.

C.1.3 Cross-Model Results

We repeat the zero-shot steering experiment with Llama 3.1 8B at layer 7 and average three judge evaluations per sample. Llama often repeats the same response and fails to emit the expected termination phrase, causing every raw conversation to reach the 15-turn limit. We therefore apply GPT 5

Interaction style
Answer thoroughness When responding to questions, does the user tend to give detailed, complete responses or brief, minimal answers?
Clarification seeking How frequently does the user ask for examples or additional explanations?
Feedback style Does the user provide ongoing verbal feedback (*okay*, *I see*, etc.)?
Topic coherence How well does the user maintain focus on a single topic versus jumping between different points?
Contextual reference Does the user refer back to previous messages or maintain conversation context?
Implementation of feedback How does the user respond to and implement suggestions or corrections?
Confidence level How does the user express their confidence or uncertainty in their understanding?
Emotional expression How does the user communicate their emotional state, such as frustration or enthusiasm?
Real-time thought expression How does the user articulate their thought process in real time, reflecting their immediate understanding and confusion?
Error handling How does the user demonstrate trial and error, acknowledge mistakes, and correct them?
Structured problem solving Does the user follow a highly structured approach to problem solving, with clear delineation of each step?
Problem-solving engagement How does the user engage with the problem-solving process in terms of asking for details about the method or process?
Seeking guidance How consistently does the user seek guidance or validation at each step of the problem-solving process?
Incremental learning Does the user show a pattern of gradual understanding, requiring multiple examples or steps to grasp a concept?
Depth of inquiry How frequently does the user ask for detailed explanations and clarifications?
Meta-cognitive awareness Does the user exhibit awareness of their learning process by summarizing or reflecting on the steps?

Table 18: SimulatorArena interaction-style dimensions used by the similarity judge.

Mini with the SimulatorArena termination prompt (Appendix E.4) to detect the earliest point of problem completion, problem shift, or circular interaction. Evaluation uses only the conversation through this predicted endpoint. The detector shortens at least 97.7% of dialogues in every steering condition, reducing their mean length to 5.03–6.05 user turns (Table 20).

After termination detection, Llama reproduces the Qwen trend. Writing-style similarity increases from 2.18 to 2.46, and overall similarity increases from 2.61 to 2.79 as α rises from 0 to 0.3 (Table 21). Interaction-style similarity is non-monotonic, peaking at 3.15 for $\alpha = 0.2$ before declining slightly to 3.12. The user-role steering effect therefore holds across both open-weight models, although Llama requires explicit termination detection for reliable evaluation.

Writing style
Frequency of grammatical errors How often does the user break basic grammar rules?
Sentence complexity Does the user primarily use simple sentences, or do they also use compound and complex structures?
Spelling consistency Does the user often misspell words or make typos, including mathematical terms?
Punctuation and capitalization usage How does the user employ punctuation and capitalization? Are they often missing or excessive?
Range and formality of vocabulary Does the user stick to basic vocabulary or incorporate a broader lexicon, including formal mathematical terminology?
Repetitive or filler words Does the user rely heavily on certain filler terms, or repeat the same words or phrases often?
Ambiguous or clear language Is the user's message easy to interpret, or does it contain incomplete or ambiguous phrasing?
Reading level Would you estimate the user's writing as basic, intermediate, or advanced?
Use of mathematical symbols and notation Does the user write mathematical symbols and notations in LaTeX correctly and frequently in their messages?
Sentence fragmentation Does the user tend to use fragmented sentences, often breaking up their thought process into shorter, separate messages?
Use of conjunctions How frequently does the user employ conjunctions to connect ideas?
Use of slang, contractions, or emojis Does the user employ slang, contractions, emoticons, or emojis, and how frequently?

Table 19: SimulatorArena writing-style dimensions used by the similarity judge.

α	Before	After	Shortened
0.0	15.00	6.05	434 (97.7%)
0.1	15.00	5.88	439 (98.9%)
0.2	15.00	5.65	443 (99.8%)
0.3	15.00	5.03	443 (99.8%)

Table 20: Llama 3.1 8B user-turn counts before and after termination detection. The detector shortens at least 97.7% of conversations from the 15-turn limit.

Condition	Writing	Interaction	Overall
Base	2.18	3.04	2.61
Steered ($\alpha = 0.1$)	2.31	3.08	2.69
Steered ($\alpha = 0.2$)	2.39	3.15	2.77
Steered ($\alpha = 0.3$)	2.46	3.12	2.79

Table 21: SimulatorArena similarity scores for Llama 3.1 8B, averaged over three judge runs per sample. Writing-style and overall similarity increase with steering strength.

C.1.4 Judge Score Variability

We examine judge variability to assess whether the score differences in the main text are stable across evaluations. For each condition, Table 22 reports the population standard deviation across the three trial-level condition means. The overall stan-

Condition	Writing	Interaction	Overall
Qwen 3.5 9B			
Base	0.012	0.015	0.014
Steered ($\alpha = 0.1$)	0.010	0.024	0.008
Steered ($\alpha = 0.2$)	0.014	0.017	0.001
Steered ($\alpha = 0.3$)	0.010	0.006	0.005
Qwen 3.5 9B			
CoT	0.009	0.013	0.003
CoT + length	0.011	0.021	0.005
CoT + profile	0.017	0.030	0.023
CoT + profile + length	0.002	0.008	0.003
GPT 5.4			
Base	0.009	0.006	0.006
CoT + profile + length	0.004	0.017	0.008
Gemini 3 Flash			
Base	0.004	0.008	0.003
CoT + profile + length	0.008	0.025	0.012

Table 22: Across-run standard deviations for the SimulatorArena scores in Table 3. Low variability indicates that the reported condition differences are stable across judge runs.

dard deviation is at most 0.023, while the zero-shot Qwen gain from $\alpha = 0$ to $\alpha = 0.3$ is 0.116. The reported condition-level differences are therefore larger than the variation across judge trials.

C.1.5 Behavioral Exaggeration

The behavior-expression analysis in §4 examines whether user-role steering also changes the explicit behaviors that simulated users display during multi-turn math tutoring. The analysis compares human ground-truth turns with Qwen 3.5 9B simulator turns generated from the base SimulatorArena prompt at $\alpha \in \{0, 0.1, 0.2, 0.3\}$. Each item is an assistant–user turn pair: the current user message is the target, and the immediately preceding assistant message is provided only as local context. The first user turn in each dialogue is skipped because it has no preceding assistant message. Ground-truth pairs are reconstructed from the math-tutoring annotations, and generated pairs are extracted from the corresponding simulator conversations.

We label four explicit markers of user behavior: doubt, misunderstanding, mistake, and question asking. GPT 5 Mini serves as the judge with minimal reasoning effort. The prompt instructs the judge to use only the current user message as evidence, while allowing multiple labels to be true for the same turn (Appendix E.4). Rates in Table 4 use all extracted assistant–user turn pairs as the denominator, so the sum of labels can exceed one label per turn. We retain only first-problem turns from the ground-truth dialogues to match the generated conversations.

C.1.6 Profile-Prompt Interaction

We directly evaluate whether steering preserves the sample-specific attributes supplied by profile prompts. For each of 444 samples per profile type, excluding 6 samples without profile prompts, we compare the four steering conditions on 12 writing-style and 16 interaction-style dimensions, yielding 12,432 rankings. Writing-style comparisons include only user messages; interaction-style comparisons include the full dialogue as context while evaluating only user behavior. For each sample and dimension, the four candidates are independently assigned randomized labels, and GPT 5 Mini ranks them from strongest to weakest expression without observing their conditions or steering strengths. The writing- and interaction-style ranking prompts appear in Appendix E.4.

Dimension	0.0	0.1	0.2	0.3
Interaction Style				
Answer Thoroughness	2.606	2.457	2.412	2.525
Clarification Seeking	2.401	2.455	2.518	2.626
Confidence Level	2.331	2.374	2.482	2.813
Contextual Reference	2.236	2.405	2.529	2.829
Depth of Inquiry	2.194	2.387	2.581	2.838
Emotional Expression	2.412	2.347	2.561	2.680
Error Handling	2.185	2.403	2.547	2.865
Feedback Style	2.511	2.354	2.468	2.667
Implementation of Feedback	2.243	2.336	2.493	2.928
Incremental Learning	2.178	2.473	2.595	2.755
Meta-Cognitive Awareness	2.252	2.306	2.523	2.919
Problem-Solving Engagement	2.209	2.347	2.473	2.971
Real-Time Thought Expression	2.412	2.340	2.527	2.721
Seeking Guidance	2.338	2.550	2.590	2.523
Structured Problem-Solving	2.313	2.500	2.502	2.685
Topic Coherence	2.230	2.313	2.536	2.921
Overall	**2.316**	**2.397**	**2.521**	**2.766**
Writing Style				
Ambiguous or Clear Language	2.146	2.288	2.732	2.833
Frequency of Grammatical Errors	2.131	2.345	2.687	2.838
Punctuation and Capitalization Usage	2.036	2.304	2.649	3.011
Range and Formality of Vocabulary	2.275	2.273	2.640	2.813
Reading Level	2.245	2.270	2.642	2.842
Repetitive or Filler Words	2.151	2.214	2.622	3.014
Sentence Complexity	2.649	2.489	2.401	2.462
Sentence Fragmentation	2.243	2.338	2.685	2.734
Spelling Consistency	2.439	2.432	2.345	2.784
Use of Conjunctions	2.561	2.495	2.457	2.486
Use of Mathematical Symbols and Notation	2.191	2.239	2.646	2.923
Use of Slang/Contractions/Emojis	2.369	2.338	2.459	2.833
Overall	**2.286**	**2.335**	**2.580**	**2.798**

Table 23: Mean profile-expression ranks by feature and steering strength. Higher steering generally produces weaker profile expression (higher rank).

Mean rank increases with steering strength for both profile types, indicating weaker expression of the assigned attributes (Table 23). The aggregate decline therefore reflects a broad loss of profile-specific signals rather than a small set of features.

C.2 User-Role Activation for Assessing Simulation Realism

C.2.1 Activation Measurement Setup

This experiment tracks whether generated user turns remain aligned with the user role direction during multi-turn simulation. The source material is RealUserSim: starting from 600 rows, we retain 221 dialogues with 20–40 messages and sample 100 dialogues. Sampling targets the ten most frequent task types with a quota of 10 per type; when a task type has fewer eligible rows, the remainder is filled from the available filtered pool. The final sample covers the ten target task types and 35 domains, with the largest domains being Technology & IT, E-commerce, Travel & Hospitality, Business & Finance, and Medical & Health. Five source dialogues are excluded from all conditions because out-of-memory errors. The analysis therefore uses the same 95 completed dialogues across all four prompt variants.

The user simulator is Qwen 3.5 9B, and the assistant side is handled by Gemini 3.1 Flash Lite. The assistant sees the ground-truth dialogue as reference context and uses it to keep the new conversation on the same trajectory, but it does not copy the original response exactly. The simulated user receives one of four prompts: the task goal alone, the goal with demographics, the goal with a linguistic profile, or the goal with both profiles (see § E.5). Figure 22 shows example demographic and linguistic attributes from one source dialogue. It continues until it signals that the conversation should end or reaches the 20-turn limit. For activation tracking, we capture the layer-11 hidden state of every generated user-response token and project each state onto the normalized user role direction from the Qwen 3.5 9B extraction output. We average these token-level projections within each message and exclude the final turn of every dialogue, which mostly contain the termination token. The curves in Figure 8 are degree-2 polynomial fits over the remaining message-level projections by raw dialogue turn.

Demographic profile. Age: 25–34; occupation: professional designer; education: bachelor's degree; marital status: married.

Linguistic profile (excerpt). Use proper capitalization and terminal punctuation. Make detailed, structured requests with numbered or bulleted lists. Use direct and specific requests. Maintain a polite, formal tone.

Figure 22: Example demographic and linguistic profiles from the same RealUserSim dialogue. The linguistic profile is shortened for presentation.

D Responsible Research and Artifact Use

D.1 Licenses and Access Terms

We used existing artifacts under their public release terms and cite the corresponding papers throughout the main text. For LMSYS-Chat-1M, we used the Hugging Face release subject to its dataset license agreement (`https://huggingface.co/datasets/lmsys/lmsys-chat-1m`). For WildChat, we used the public release described as ODC-BY licensed and collected through opt-in, anonymized chat sharing (`https://huggingface.co/datasets/allenai/WildChat`; `https://wildchat.allen.ai/`). SimulatorArena and RealUserSim are used as research benchmarks/datasets for simulator evaluation, consistent with their stated research purpose. We do not redistribute raw third-party dialogue logs; derived artifacts in this paper are aggregate statistics, prompts, figures, and analysis code intended for research use.

D.2 Intended Use and Sensitive Content

Our use of existing artifacts is limited to representation analysis and evaluation of user simulation. Because public user-assistant dialogue datasets may contain personal, sensitive, or offensive content, we avoid presenting raw user logs except for short illustrative excerpts already abstracted in the paper. Generated prompts and analysis outputs are intended to support research on simulator fidelity, not deployment of user simulators in production or profiling of real individuals.

D.3 Artifact Documentation

The appendix documents data filtering, topic-label distributions, task coverage, prompts, evaluation dimensions, and error cases for the experiments. This includes the LMSYS filtering pipeline, the sampled RealUserSim task and domain coverage, SimulatorArena evaluation features, and the out-of-memory failures observed in profile-prompt activation runs.

D.4 Package and Model Parameters

The key packages are `torch`, `transformers`, `transformer-lens`, `accelerate`, and `bitsandbytes` for local model loading and

activation steering. The main models are Qwen 3.5 9B for role-vector extraction, steering, and activation tracking; GPT 5 Mini for topic labels and LLM-as-judge evaluations; GPT 5.4 and Gemini 3 Flash for SimulatorArena baselines; Gemini 3.1 Flash Lite for assistant-side replay in the activation-tracking experiment; and Llama 3.1 8B and Granite 3.3 8B for cross-model layer analyses.

D.5 Use of AI Assistants

AI assistants were used during research coding and manuscript preparation for code editing, plotting support, grammar refinement, and phrasing suggestions. All generated code, tables, figures, and prose were reviewed and edited by the authors, and numerical claims were checked against the corresponding analysis outputs before inclusion.

E Prompts

This appendix collects the prompts used across the experiments.

E.1 User Role Vector Extraction

Topic Labeling

```
You are an internal tool that classifies a message from a
↪  user to an AI chatbot, based on the context of the
↪  previous messages before it.
Based on the last user message of this conversation
↪  transcript and taking into account the examples further
↪  below as guidance, please select the capability the
↪  user is clearly interested in, or `other` if it is
↪  clear but not in the list below, or `unclear` if it is
↪  hard to tell what the user even wants:

- edit_or_critique_provided_text: Improving or modifying
↪  text provided by the user.
- argument_or_summary_generation: Creating arguments or
↪  summaries on topics not provided in detail by the user.
- personal_writing_or_communication: Assisting with
↪  personal messages, emails, or social media posts.
- write_fiction: Crafting poems, stories, or fictional
↪  content.
- how_to_advice: Providing step-by-step instructions or
↪  guidance on how to perform tasks or learn new skills.
- creative_ideation: Generating ideas or suggestions for
↪  creative projects or activities.
- tutoring_or_teaching: Explaining concepts, teaching
↪  subjects, or helping the user understand educational
↪  material.
- translation: Translating text from one language to
↪  another.
- mathematical_calculation: Solving math problems,
↪  performing calculations, or working with numerical
↪  data.
- computer_programming: Writing code, debugging, explaining
↪  programming concepts, or discussing programming
↪  languages and tools.
- purchasable_products: Inquiries about products or
↪  services available for purchase.
- cooking_and_recipes: Seeking recipes, cooking
↪  instructions, or culinary advice.
- health_fitness_beauty_or_self_care: Seeking advice or
↪  information on physical health, fitness routines,
↪  beauty tips, or self-care.
- specific_info: Providing specific information typically
↪  found on websites, including information about
↪  well-known individuals, current events, historical
↪  events, and other facts and knowledge.
- greetings_and_chitchat: Casual conversation, small talk,
↪  or friendly interactions without a specific
↪  informational goal.
- relationships_and_personal_reflection: Discussing
↪  personal reflections or seeking advice on relationships
↪  and feelings.
- games_and_role_play: Engaging in interactive games,
↪  simulations, or imaginative role-playing scenarios.
- asking_about_the_model: Questions about the AI model's
↪  capabilities or characteristics.
- create_an_image: Requests to generate or draw new visual
↪  content based on the user's description.
- analyze_an_image: Interpreting or describing visual
↪  content provided by the user, such as photos, charts,
↪  graphs, or illustrations.
- generate_or_retrieve_other_media: Creating or finding
↪  media other than text or images, such as audio, video,
↪  or multimedia files.
- data_analysis: Performing statistical analysis,
↪  interpreting datasets, or extracting insights from
↪  data.
- unclear: If the user's intent is not clear from the
↪  conversation.
- other: If the capability requested doesn't fit any of the
↪  above categories.

Okay, now your turn, taking the user conversation at the top
↪  into account: What capability are they seeking? (JUST
↪  SAY A SINGLE CATEGORY FROM THE LIST, NOTHING ELSE).
If the conversation has multiple distinct capabilities,
↪  choose the one that is the most relevant to the LAST
↪  user message in the conversation.
```

Reflection Generation: Transcript Analysis

```
You are analyzing the following dialogue from the
↪  perspective of the {role}.

Dialogue:
{dialogue_text}

Instruction:
Write a brief reflection from the {role}'s perspective in
↪  one paragraph.

Write exactly one paragraph.
```

Reflection Generation: Role Simulation

```
You are simulating the role of the {role} in the following
↪  dialogue.

Dialogue:
{dialogue_text}

Instruction:
Write a brief reflection from the {role}'s perspective in
↪  one paragraph.

Write exactly one paragraph.
```

Reflection Generation: Dialogue Participant

```
You are the {role} in the following dialogue.

Dialogue:
{dialogue_text}

Instruction:
Write a brief reflection from the {role}'s perspective in
↪  one paragraph.

Write exactly one paragraph.
```

Reflection Validation: System Message

```
Decide how strongly the reflection represents the {role}'s
↪  perspective in the dialogue.
Return exactly one label and no other text.
```

Reflection Validation: User Message

```
Dialogue:
{dialogue_text}

Reflection of {role}:
{answer}

How strongly is the {role}'s perspective represented in the
↪  reflection?

strongly_represented: clearly captures the {role}'s
↪  perspective and is grounded in the {role}'s actual
↪  turns.
weakly_represented: partially captures the {role}'s
↪  perspective, but is generic, weakly grounded, or mixed
↪  with the other role's perspective.
not_represented: mostly captures the wrong role,
↪  contradicts the {role}, or is too generic to represent
↪  the {role}.
```

```
Respond only with one of these labels and do not include
↪  any other text: strongly_represented,
↪  weakly_represented, not_represented.
```

E.2 Communication-Style Evaluation

Task Goal Generation

```
Generate {request_count} diverse task_goal items.

task_goal:
- A plain goal, need, or task to complete.
- One sentence.

Make the tasks diverse, covering various actions, domains,
↪  and scenarios.
Make the tasks generalizable rather than tied to any
↪  specific individual.
Keep each task_goal unique.
```

Request Generation

```
Generate a request message for an assistant based on the
↪  task goal.

Task goal:
{task_goal}

Write the request message only.
```

User-Like Request Generation

```
Generate a request message that a real human user would
↪  send to an assistant based on the task goal.
The message should sound like it was written by a real
↪  human user.

Task goal:
{task_goal}

Write only the user's request message.
```

MultiWOZ 2.0 Request Generation

```
The following shows a user goal for a conversation with an
↪  assistant.
Generate an initial message to start the conversation with
↪  the assistant, based on the user goal.
Gradually reveal the details of the user goal throughout
↪  the conversation.

User goal:
{goal_description}

Write the initial user message.
```

TREC Health Question Generation

```
The following shows information related to a health
↪  question.
Based on the topic and context, generate a question to ask
↪  the assistant for clarification.

Topic:
{query}

Context:
{background}

Write the question message only.
```

User-Likeness Evaluation

```
Evaluate the style of the request message.

Score three dimensions on a 1-5 scale.
Higher scores mean the message shows more user-like traits
↪  for that dimension.
Lower scores mean the message shows more assistant-like
↪  traits for that dimension.

Brevity:
- Focus on word count and short-turn behavior.
- Score 1: long, complete, polished, or over-explained.
  Example: "Could you provide a detailed and
↪  well-structured comparison of three noise-canceling
↪  headphones, including comfort, battery life, price, and
↪  suitability for travel?"
- Score 5: very short, direct, fragment-like, or only says
↪  what is needed.
  Example: "best headphones for flights?"

Informality:
- Focus on surface style, formality, and conversational
↪  roughness.
- Score 1: formal, careful, service-like, or written in
↪  polished complete sentences.
  Example: "Could you please assist me in drafting an
↪  appropriate email declining this invitation?"
- Score 5: casual, blunt, uneven, ordinary, or written like
↪  quick everyday conversation.
  Example: "how do i say no to this invite without sounding
↪  rude"

Information pacing:
- Focus on whether the message front-loads information or
↪  withholds details.
- Score 1: gives the full context, identifiers, constraints,
↪  and desired outcome upfront.
  Example: "I need to return order #48291, a black jacket
↪  bought last Tuesday, because the zipper broke; please
↪  write a message asking for a refund or replacement."
- Score 5: gives only the immediate need, leaves details
↪  implicit, or would require follow-up.
  Example: "can i return this?"

Judge only the message itself. Do not judge helpfulness or
↪  quality.

Message:
{request_message}

Return JSON:
{
  "brevity": {"thought": "short reason", "score": 1},
  "informality": {"thought": "short reason", "score": 1},
  "information_pacing": {"thought": "short reason", "score":
↪  1}
}
```

E.3 Disengagement Prediction

Disengagement Decision

```
You are simulating the user in the conversation below.
Decide whether to continue the conversation or disengage
↪  after the current assistant response.

User intent:
{user_intent}

Conversation:
{conversation_prefix}

Task:
After the current assistant response, choose what you would
↪  do next.
Make the decision as a user.

(A) {option_a}
(B) {option_b}

Return only one character:
A or B
```

E.4 SimulatorArena

User Simulation

Zero-Shot CoT: Initial Query

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your goal is to generate a realistic and appropriate
↪  **initial query** that a student might say when first
↪  encountering the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Your initial query can express your level of
↪  understanding, confusion, or reasoning about the
↪  problem.
3. You can make mistakes or misunderstandings that a real
↪  student might have.
4. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Task:
Formulate an initial query that reflects your current
↪  understanding and areas of confusion regarding the
↪  problem. This query can express general uncertainty of
↪  the problem or about specific parts that you find
↪  challenging.

## Thought Process
Before generating your initial query, analyze the problem
↪  as a student. Consider:
- Your understanding of the problem and areas of uncertainty
- How the problem relates to concepts you already know
- Your initial ideas or possible approaches
- Specific concepts you need help understanding

## Query Generation
Based on your thought process, generate:
A query that expresses your initial understanding,
↪  confusion, or approach to the problem.

# Output format:
Thought: [Your chain of thought reasoning about what to say
↪  to the tutor]

Query: [Your initial query for the AI tutor]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it in your query.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve easily.

Now, stay in character as a student throughout your output
↪  and follow the above guidelines carefully.
```

Zero-Shot CoT with Length Control: Initial Query

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your goal is to generate a realistic and appropriate
↪  **initial query** that a student might say when first
↪  encountering the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Your initial query can express your level of
↪  understanding, confusion, or reasoning about the
↪  problem.
3. You can make mistakes or misunderstandings that a real
↪  student might have.
4. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Task:
Formulate a {length_control} initial query that reflects
↪  your current understanding and areas of confusion
↪  regarding the problem. This query can express general
↪  uncertainty of the problem or about specific parts that
↪  you find challenging.

## Thought Process
Before generating your initial query, analyze the problem
↪  as a student. Consider:
- Your understanding of the problem and areas of uncertainty
- How the problem relates to concepts you already know
- Your initial ideas or possible approaches
- Specific concepts you need help understanding

## Query Generation
Based on your thought process, generate:
A query that expresses your initial understanding,
↪  confusion, or approach to the problem.

# Output format
Thought: [Your chain of thought reasoning about what to say
↪  to the tutor]

Query: [Your initial query for the AI tutor,
↪  {length_control}]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it in your query.
- Keep your query {length_control}.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve easily.

Stay in character as a student throughout your output,
↪  following the above guidelines carefully.
```

Zero-Shot CoT with Length Control: Follow-Up Response

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your task is to generate realistic and appropriate
↪  responses that a student might make when trying to
↪  solve the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Each response can be a question or a statement that
↪  demonstrates your current understanding, confusion, or
↪  reasoning.
3. Respond naturally to the tutor's explanations, hints,
↪  and questions, showing progress in your understanding.
4. You can make mistakes or misunderstandings that a real
↪  student might have.
5. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Conversation History:
{conversation_history}

# Task:
Use the conversation history to generate the next response
↪  you would give to the AI tutor. It should follow
↪  naturally and reflect your current level of
↪  understanding or confusion, and it should be
↪  {length_control} in length.

## Thought Process
Before generating your response, analyze the current
↪  situation as a student. Consider:
```

```
- Your current level of understanding of the concepts involved
- Any gaps or uncertainties in your knowledge
- The tutor's most recent explanation or question
- What would help you progress toward solving the problem
- Whether you need clarification on specific aspects
- Your ability to proceed with the next step

## Response Generation
Based on your thought process, generate a {length_control} response that reflects your current understanding and learning needs.
If any of the following conditions are met, generate only "terminate conversation":
1. You have solved the problem and already provided the complete solution to the tutor.
2. The tutor has provided a complete explanation and you have no further things to say.
3. The conversation is no longer productive (e.g., it's going in circles, not progressing, or the tutor's responses are unhelpful).

# Output Format:
Thought: [Your analysis of the current situation and what you want to say]

Response: [Your response to the tutor, {length_control}]

# Notes:
- The tutor already knows the problem, so you don't need to restate it.
- Keep your response {length_control}.
- Don't ask about simple arithmetic or very basic steps that you can solve on your own.
- Don't ask for any additional problems after you solve the problem.

Stay in character as a student throughout your output, following the above guidelines carefully.
```

Zero-Shot CoT with Message Format: Initial Query

```
You are an AI assistant tasked with role-playing as a student seeking help from an AI tutor on a math problem. Your goal is to generate a realistic and appropriate **initial message** that a student might say when first encountering the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic mathematics (e.g., arithmetic) but are struggling with the given problem.
2. Your initial message can express your level of understanding, confusion, or reasoning about the problem.
3. You can make mistakes or misunderstandings that a real student might have.
4. Your overall goal is to learn how to solve the given problem.

# Math Problem:
{math_problem}

# Task:
Formulate an initial message that reflects your current understanding and areas of confusion regarding the problem. This message can express general uncertainty of the problem or about specific parts that you find challenging.

## Thought Process
Before generating your initial message, analyze the problem as a student. Consider:
- Your understanding of the problem and areas of uncertainty
- How the problem relates to concepts you already know
- Your initial ideas or possible approaches
- Specific concepts you need help understanding

## Message Generation
Based on your thought process, generate:
A message that expresses your initial understanding, confusion, or approach to the problem.

# Output format:
Thought: [Your chain of thought reasoning about what to say to the tutor]

Message: [Your initial message for the AI tutor]

# Notes:
- The tutor already knows the problem, so you don't need to restate it in your message.
- Don't ask about simple arithmetic or very basic steps that you can solve easily.

Now, stay in character as a student throughout your output and follow the above guidelines carefully.
```

Zero-Shot CoT with Message Format: Follow-Up Response

```
You are an AI assistant tasked with role-playing as a student seeking help from an AI tutor on a math problem. Your task is to generate realistic and appropriate messages that a student might make when trying to solve the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic mathematics (e.g., arithmetic) but are struggling with the given problem.
2. Each message can be a question or a statement that demonstrates your current understanding, confusion, or reasoning.
3. Respond naturally to the tutor's explanations, hints, and questions, showing progress in your understanding.
4. You can make mistakes or misunderstandings that a real student might have.
5. Your overall goal is to learn how to solve the given problem.

# Math Problem:
{math_problem}

# Conversation History:
{conversation_history}

# Task:
Use the conversation history to generate the next message you would give to the AI tutor. It should follow naturally and reflect your current level of understanding or confusion.

## Thought Process
Before generating your message, analyze the current situation as a student. Consider:
- Your current level of understanding of the concepts involved
- Any gaps or uncertainties in your knowledge
- The tutor's most recent explanation or question
- What would help you progress toward solving the problem
- Whether you need clarification on specific aspects
- Your ability to proceed with the next step

## Message Generation
Based on your thought process, generate a message that reflects your current understanding and learning needs.
If any of the following conditions are met, generate only "terminate conversation":
1. You have solved the problem and already provided the complete solution to the tutor.
2. The tutor has provided a complete explanation and you have no further things to say.
3. The conversation is no longer productive (e.g., it's going in circles, not progressing, or the tutor's messages are unhelpful).

# Output Format:
Thought: [Your analysis of the current situation and what you want to say]

Message: [Your message to the tutor]

# Notes:
```

```
- The tutor already knows the problem, so you don't need to
↪  restate it.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve on your own.
- Don’t ask for any additional problems after you solve the
↪  problem.

Stay in character as a student throughout your output,
↪  following the above guidelines carefully.
```

Zero-Shot CoT with User Profile: Initial Query

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your primary goal is to accurately simulate a student
↪  with the specific characteristics defined in the
↪  profile below. This profile simulation is crucial for
↪  maintaining authenticity in the conversation.

# User Profile:
{user_profile}

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Your initial query can express your level of
↪  understanding, confusion, or reasoning about the
↪  problem.
3. You can make mistakes or misunderstandings that a real
↪  student might have.
4. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Task:
Formulate an initial query that adheres to the user profile
↪  provided above and reflects your current understanding
↪  and areas of confusion regarding the problem. This
↪  query can express general uncertainty of the problem or
↪  about specific parts that you find challenging.

## Thought Process
Before generating your initial query, think about both your
↪  understanding of the mathematics and how to express it
↪  according to your user profile:
### Understanding the Problem:
- Your understanding of the problem and areas of uncertainty
- How the problem relates to concepts you already know
- Your initial ideas or possible approaches
- Specific concepts you need help understanding

### Maintaining Profile Characteristics:
- How to express your thoughts according to the given
↪  profile
- Which profile characteristics are most relevant to this
↪  response
- How to naturally incorporate these characteristics into
↪  your query

## Query Generation
Based on your thought process, generate:
A query that expresses your initial understanding,
↪  confusion, or approach to the problem.

# Output format:
Thought: [Your analysis of the current situation and how to
↪  express it according to the user profile]

Query: [Your initial query for the AI tutor]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it in your query.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve easily.

Stay in character as the specified student throughout your
↪  output, following the guidelines and user profile
↪  characteristics carefully.
```

Zero-Shot CoT with User Profile and Length Control: Initial Query

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your primary goal is to accurately simulate a student
↪  with the specific characteristics defined in the
↪  profile below. This profile simulation is crucial for
↪  maintaining authenticity in the conversation.

# User Profile:
{user_profile}

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Your initial query can express your level of
↪  understanding, confusion, or reasoning about the
↪  problem.
3. You can make mistakes or misunderstandings that a real
↪  student might have.
4. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Task:
Formulate an initial query that adheres to the user profile
↪  provided above and reflects your current understanding
↪  and areas of confusion regarding the problem. This
↪  query can express general uncertainty of the problem or
↪  about specific parts that you find challenging.

## Thought Process
Before generating your initial query, think about both your
↪  understanding of the mathematics and how to express it
↪  according to your user profile:
### Understanding the Problem:
- Your understanding of the problem and areas of uncertainty
- How the problem relates to concepts you already know
- Your initial ideas or possible approaches
- Specific concepts you need help understanding

### Maintaining Profile Characteristics:
- How to express your thoughts according to the given
↪  profile
- Which profile characteristics are most relevant to this
↪  response
- How to naturally incorporate these characteristics into
↪  your query

## Query Generation
Based on your thought process, generate:
A query that expresses your initial understanding,
↪  confusion, or approach to the problem.

# Output format:
Thought: [Your analysis of the current situation and how to
↪  express it according to the user profile]

Query: [Your initial query for the AI tutor,
↪  {length_control}]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it in your query.
- Keep your query {length_control}.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve easily.

Stay in character as the specified student throughout your
↪  output, following the guidelines and user profile
↪  characteristics carefully.
```

Zero-Shot CoT with User Profile and Length Control: Follow-Up Response

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your primary goal is to accurately simulate a student
↪  with the specific characteristics defined in the
↪  profile below. This profile simulation is crucial for
↪  maintaining authenticity in the conversation.

# User Profile
{user_profile}

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Each response can be a question or a statement that
↪  demonstrates your current understanding, confusion, or
↪  reasoning.
3. Respond naturally to the tutor's explanations, hints,
↪  and questions, showing progress in your understanding.
4. You can make mistakes or misunderstandings that a real
↪  student might have.
5. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Conversation History:
{conversation_history}

# Task:
Use the conversation history to generate the next response
↪  you would give to the AI tutor. It should follow
↪  naturally and reflect your current level of
↪  understanding or confusion. It also needs to adhere to
↪  the user profile provided above.

## Thought Process
Before generating your response, analyze the current
↪  situation as a student. Consider:
- Your current level of understanding of the concepts
↪  involved
- Any gaps or uncertainties in your knowledge
- The tutor's most recent explanation or question
- What would help you progress toward solving the problem
- Whether you need clarification on specific aspects
- Your ability to proceed with the next step

### Maintaining Profile Characteristics:
- How to express your thoughts according to the given
↪  profile
- Which profile characteristics are most relevant to this
↪  response
- How to naturally incorporate these characteristics into
↪  your response

## Response Generation
Based on your thought process, generate a {length_control}
↪  response that reflects your current understanding and
↪  learning needs.
If any of the following conditions are met, generate only
↪  "terminate conversation":
1. You have solved the problem and already provided the
↪  complete solution to the tutor.
2. The tutor has provided a complete explanation and you
↪  have no further things to say.
3. The conversation is no longer productive (e.g., it's
↪  going in circles, not progressing, or the tutor's
↪  responses are unhelpful).

# Output Format:
Thought: [Your analysis of the current situation and how to
↪  express it according to the user profile]

Response: [Your response to the tutor, {length_control}]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it.
- Keep your response {length_control}.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve on your own.
- Don't ask for any additional problems after you solve the
↪  problem.

Stay in character as the specified student throughout your
↪  output, following the guidelines and user profile
↪  characteristics carefully.
```

Zero-Shot CoT with User Profile: Follow-Up Response

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your primary goal is to accurately simulate a student
↪  with the specific characteristics defined in the
↪  profile below. This profile simulation is crucial for
↪  maintaining authenticity in the conversation.

# User Profile
{user_profile}

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Each response can be a question or a statement that
↪  demonstrates your current understanding, confusion, or
↪  reasoning.
3. Respond naturally to the tutor's explanations, hints,
↪  and questions, showing progress in your understanding.
4. You can make mistakes or misunderstandings that a real
↪  student might have.
5. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Conversation History:
{conversation_history}

# Task:
Use the conversation history to generate the next response
↪  you would give to the AI tutor. It should follow
↪  naturally and reflect your current level of
↪  understanding or confusion. It also needs to adhere to
↪  the user profile provided above.

## Thought Process
Before generating your response, analyze the current
↪  situation as a student. Consider:
- Your current level of understanding of the concepts
↪  involved
- Any gaps or uncertainties in your knowledge
- The tutor's most recent explanation or question
- What would help you progress toward solving the problem
- Whether you need clarification on specific aspects
- Your ability to proceed with the next step

### Maintaining Profile Characteristics:
- How to express your thoughts according to the given
↪  profile
- Which profile characteristics are most relevant to this
↪  response
- How to naturally incorporate these characteristics into
↪  your response

## Response Generation
Based on your thought process, generate a response that
↪  reflects your current understanding and learning needs.
If any of the following conditions are met, generate only
↪  "terminate conversation":
1. You have solved the problem and already provided the
↪  complete solution to the tutor.
2. The tutor has provided a complete explanation and you
↪  have no further things to say.
3. The conversation is no longer productive (e.g., it's
↪  going in circles, not progressing, or the tutor's
↪  responses are unhelpful).

# Output Format:
Thought: [Your analysis of the current situation and how to
↪  express it according to the user profile]
```

```
Response: [Your response to the tutor]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve on your own.
- Don't ask for any additional problems after you solve the
↪  problem.

Stay in character as the specified student throughout your
↪  output, following the guidelines and user profile
↪  characteristics carefully.
```

Zero-Shot CoT: Follow-Up Response

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your task is to generate realistic and appropriate
↪  responses that a student might make when trying to
↪  solve the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Each response can be a question or a statement that
↪  demonstrates your current understanding, confusion, or
↪  reasoning.
3. Respond naturally to the tutor's explanations, hints,
↪  and questions, showing progress in your understanding.
4. You can make mistakes or misunderstandings that a real
↪  student might have.
5. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Conversation History:
{conversation_history}

# Task:
Use the conversation history to generate the next response
↪  you would give to the AI tutor. It should follow
↪  naturally and reflect your current level of
↪  understanding or confusion.

## Thought Process
Before generating your response, analyze the current
↪  situation as a student. Consider:
- Your current level of understanding of the concepts
↪  involved
- Any gaps or uncertainties in your knowledge
- The tutor's most recent explanation or question
- What would help you progress toward solving the problem
- Whether you need clarification on specific aspects
- Your ability to proceed with the next step

## Response Generation
Based on your thought process, generate a response that
↪  reflects your current understanding and learning needs.
If any of the following conditions are met, generate only
↪  "terminate conversation":
1. You have solved the problem and already provided the
↪  complete solution to the tutor.
2. The tutor has provided a complete explanation and you
↪  have no further things to say.
3. The conversation is no longer productive (e.g., it's
↪  going in circles, not progressing, or the tutor's
↪  responses are unhelpful).

# Output Format:
Thought: [Your analysis of the current situation and what
↪  you want to say]

Response: [Your response to the tutor]

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve on your own.
- Don't ask for any additional problems after you solve the
↪  problem.

Stay in character as a student throughout your output,
↪  following the above guidelines carefully.
```

Zero-Shot: Initial Query

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your goal is to generate a realistic and appropriate
↪  **initial query** that a student might say when first
↪  encountering the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Your initial query can express your level of
↪  understanding, confusion, or reasoning about the
↪  problem.
3. You can make mistakes or misunderstandings that a real
↪  student might have.
4. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Task:
Formulate an initial query that reflects your current
↪  understanding and areas of confusion regarding the
↪  problem. This query can express general uncertainty of
↪  the problem or about specific parts that you find
↪  challenging.

# Output Format:
Provide only your **initial query** to the AI tutor,
↪  without any additional commentary or explanation.

# Notes:
- The tutor already knows the problem, so you don't need to
↪  restate it in your query.
- Don't ask about simple arithmetic or very basic steps
↪  that you can solve easily.

Stay in character as a student throughout your output,
↪  following the above guidelines carefully.
```

Zero-Shot: Follow-Up Response

```
You are an AI assistant tasked with role-playing as a
↪  student seeking help from an AI tutor on a math problem.
↪  Your task is to generate a realistic and appropriate
↪  response that a student might make when trying to solve
↪  the given problem with the tutor.

# Guidelines for Your Role as a Student:
1. Act as if you have a solid foundation in basic
↪  mathematics (e.g., arithmetic) but are struggling with
↪  the given problem.
2. Each response can be a question or a statement that
↪  demonstrates your current understanding, confusion, or
↪  reasoning.
3. Respond naturally to the tutor's explanations, hints,
↪  and questions, showing progress in your understanding.
4. You can make mistakes or misunderstandings that a real
↪  student might have.
5. Your overall goal is to learn how to solve the given
↪  problem.

# Math Problem:
{math_problem}

# Conversation History:
{conversation_history}

# Task:
```

```
Use the conversation history to generate the next response you would give to the AI tutor. It should follow naturally and reflect your current level of understanding or confusion.

If any of the following conditions are met, output only "terminate conversation":
1. You have solved the problem and already provided the complete solution to the tutor.
2. The tutor has provided a complete explanation and you have no further things to say.
3. The conversation is no longer productive (e.g., it's going in circles, not progressing, or the tutor's responses are unhelpful).

# Output Format:
Provide only the next response you would give to the AI tutor, without any additional commentary or explanation.

# Notes:
- The tutor already knows the problem, so you don't need to restate it.
- Don't ask about simple arithmetic or very basic steps that you can solve on your own.
- Don't ask for any additional problems after you solve the problem.

Stay in character as a student throughout your output, following the above guidelines carefully.
```

Assistant

Math-Tutoring Assistant

```
You are a skilled math tutor. Your goal is to help students understand and solve problems independently.
Provide guidance based on their questions or mistakes.
Ask questions to encourage their thinking and let students do most of the work themselves.
Never give out the solution directly to students.

Problem:
{question}
```

Evaluation

Termination Detection

```
You are given a sequence of **User Queries** from a math tutoring conversation, along with the **Math Problem**. Your task is to determine the optimal point to end the conversation based on the user's learning progression.

## Input Format
### Math Problem:
{problem}

### User Queries:
{user_messages}

## Termination Criteria
End the conversation when ANY of these occur:
1. **Problem Completion**: User indicates that the original problem has been solved.
2. **Problem Shift**: User begins asking about another mathematical problem, including a related or practice problem.
3. **Circular Queries**: User repeats similar responses without showing progress in understanding.

## Output Format
```json
{
    "Analysis": [
        "Turn 1: [Brief analysis of user's understanding/intent]",
        "Turn 2: [Brief analysis of user's understanding/intent]",
        ...
    ],
    "Ending Turn Number": X,
    "Termination Reason": "[One of the three criteria above]"
}
```

Notes:
1. The "Ending Turn Number" should be the last turn that's relevant to learning the original math problem.
2. Choose the earliest turn that satisfies a termination criterion.
3. Only consider ending for:
- Student explicitly indicates problem completion.
- Clear problem shifts.
- Circular queries with no progress.
```

Interaction-Style Evaluation

```
You are an expert in analyzing {task} conversation. Your task is to evaluate how similar a simulated user's interaction style is to a real user user's interaction style in their queries to an AI assistant.


# Input
## Document Type
{document_type}

## Document Goal
{intent}

## Conversation between Real User and AI Assistant
{real_conversation}

## Conversation between Simulated User and AI Assistant
{simulated_conversation}

# Analysis Features
Compare the following **interaction style features** between the real user's queries and the simulated user's queries to guide your evaluation:


{features}

## **Similarity Rating Scale**
Rate the overall interaction style similarity on a scale of 1-5:
1: Simulated user's interaction style is completely different from real user's interaction style
2: Simulated user's interaction style shows significant differences from real user's interaction style
3: Simulated user's interaction style shows notable differences while maintaining some similarity
4: Simulated user's interaction style is very similar to real user's with minor differences
5: Simulated user's interaction style is nearly indistinguishable from real user's interaction style

# Output Format
## Feature Analysis:
- Analyze each listed feature **individually**.
- Compare **how the real user and the simulated user** differ or align for that feature.

## Results (JSON):
```json
{{
   "key_differences": ["list specific differences in interaction style here"],
   "similarity_score": "1-5"
}}
```

Note
- Focus only on user interaction style, not AI assistant's.
- Your goal is to determine how closely the simulated user match real user in their interaction style.
```

Writing-Style Evaluation

```
You are an expert in analyzing {task} conversation. Your task is to evaluate how similar a simulated user's writing style is to a real user user's writing style in their queries to an AI assistant.


# Input
## Document Type
{document_type}

## Document Goal
{intent}

## Real User Queries
{real_user_queries}

## Simulated User Queries
{simulated_queries}

# Analysis Features
Compare the following **writing style features** between the real user's queries and the simulated user's queries to guide your evaluation:


{features}

## **Similarity Rating Scale**
Rate the overall writing style similarity on a scale of 1-5:
1: Simulated user's writing style is completely different from real user's writing style
2: Simulated user's writing style shows significant differences from real user's writing style
3: Simulated user's writing style shows notable differences while maintaining some similarity
4: Simulated user's writing style is very similar to real user's with minor differences
5: Simulated user's writing style is nearly indistinguishable from real user's writing style

# Output Format
## Feature Analysis:
- Analyze each listed feature **individually**.
- Compare **how the real user and the simulated user** differ or align for that feature.

## Results (JSON):
```json
{{
   "key_differences": ["list specific differences in writing style here"],
   "similarity_score": "1-5"
}}
```

Note
- Your goal is to determine how closely the simulated queries match real user queries in their writing style.
```

Interaction-Style Profile Ranking

```
# Context:
You are comparing how strongly the specified interaction-style attribute appears in four dialogue examples.

# Task:
- Rank the four candidates by how strongly they express this attribute, from strongest to weakest.
- Do not allow ties. Candidate labels are arbitrary and do not indicate model or condition identity.
- Evaluate only the user's interaction behavior. Assistant messages provide context for interpreting the user's behavior; do not evaluate or rank the assistants.

# Interaction-Style Attribute:
`{dimension_id}`: {dimension_description}

# Candidates:
{candidates}

Return only a JSON list containing each candidate letter exactly once, ordered from strongest to weakest.
Do not include an explanation or any other fields.
```

Writing-Style Profile Ranking

```
# Context:
You are comparing how strongly the specified writing-style attribute appears in four sets of user utterances.

# Task:
- Rank the four candidates by how strongly they express this attribute, from strongest to weakest.
- Do not allow ties. Candidate labels are arbitrary and do not indicate model or condition identity.
- Evaluate only the user's writing. Consider all provided user utterances for each candidate.

# Writing-Style Attribute:
`{dimension_id}`: {dimension_description}

# Candidates:
{candidates}

Return only a JSON list containing each candidate letter exactly once, ordered from strongest to weakest.
Do not include an explanation or any other fields.
```

Behavior-Expression Labeling

```
Analyze one assistant-user turn pair.

You should mark whether the current USER message explicitly contains each expression type.

Traits:
- expression_of_doubt: The user states uncertainty, hesitation, low confidence, or doubt about their own understanding, answer, or next step.
- expression_of_misunderstanding: The user states that they do not understand, are confused, cannot follow, or need clarification.
- expression_of_mistake: The user states that they made an error, were wrong, misread something, calculated incorrectly, or need to correct their own prior statement.
- expression_of_question: The user asks a question or requests confirmation, clarification, explanation, or guidance.

Rules:
- Use only the current USER message for evidence.
- The previous ASSISTANT message is provided only for local context.
- The trait must be stated in the user's text. Short statements like "what?", "I'm not sure", "I don't get it", or "I was wrong" can count when they directly express the trait.
- Exclude negated statements, such as "I understand now" or "I don't think I made a mistake".
- Exclude quoted, hypothetical, or third-person statements about someone else.
- Multiple traits may be true for the same user message.
- If a trait is false, use an empty evidence string.

Previous ASSISTANT message:
{assistant_message}

Current USER message:
{user_message}

Return only valid JSON in this exact structure:
{
  "expression_of_doubt": {"present": false, "evidence": ""},
  "expression_of_misunderstanding": {"present": false, "evidence": ""},
  "expression_of_mistake": {"present": false, "evidence": ""},
  "expression_of_question": {"present": false, "evidence": ""}
```

```
}
```

E.5 User-Role Activation Analysis

Assistant Replay

```
You are an AI assistant reproducing the ground-truth
↪  conversation.

Use the ground-truth conversation below as reference
↪  context, and keep your responses consistent with it.
↪  You may paraphrase lightly when appropriate.
The ground-truth conversation is the original dialogue
↪  provided as a reference for what happened before.

**Guidelines**
- Your goal is to reproduce the ground-truth dialogue.
- Follow the original trajectory and avoid revealing all
↪  information at once.
- The user does not have access to the ground-truth
↪  conversation.
- If the current user message differs from the ground-truth
↪  conversation, respond helpfully using your own
↪  knowledge while remaining consistent with the
↪  ground-truth conversation.

**Ground-truth conversation**
{ground_truth_dialogue}

**Current conversation**
{history}

Write the assistant's next message in the current
↪  conversation.
Keep your message short but complete.
```

User Simulation: Demographics

```
You are simulating a human user in a multi-turn chat with
↪  an AI assistant.
Write the user's next message based on the goal and the
↪  conversation below. Avoid repeating messages.
If the goal is satisfied, reply exactly with:
↪  {terminate_keyword}

**User goal**
{user_goal}

**User demographics**
{demographics}

**Conversation so far**
{history}

Write only the next user message.
```

User Simulation: Full Profile

```
You are simulating a human user in a multi-turn chat with
↪  an AI assistant.
Write the user's next message based on the goal and the
↪  conversation below. Avoid repeating messages.
If the goal is satisfied, reply exactly with:
↪  {terminate_keyword}

**User goal**
{user_goal}

**User demographics**
{demographics}

**User linguistic style**
{linguistic_profile}

**Conversation so far**
{history}

Write only the next user message.
```

User Simulation: Goal Only

```
You are simulating a human user in a multi-turn chat with
↪  an AI assistant.
Write the user's next message based on the goal and the
↪  conversation below. Avoid repeating messages.
If the goal is satisfied, reply exactly with:
↪  {terminate_keyword}

**User goal**
{user_goal}

**Conversation so far**
{history}

Write only the next user message.
```

User Simulation: Linguistic Profile

```
You are simulating a human user in a multi-turn chat with
↪  an AI assistant.
Write the user's next message based on the goal and the
↪  conversation below. Avoid repeating messages.
If the goal is satisfied, reply exactly with:
↪  {terminate_keyword}

**User goal**
{user_goal}

**User linguistic style**
{linguistic_profile}

**Conversation so far**
{history}

Write only the next user message.
```