IS THERE AN IMPOSTER AMONG US? AUDITING USER SIMULATORS WITH ATTRIBUTION TRIALS

Anonymous authors
Paper under double-blind review

ABSTRACT

User simulators are commonly evaluated by their ability to steer toward the target user's behavior, given the user's profile or history. However, this similarity does not show whether the simulator captures what makes that individual unique, rather than behavior shared by many users. In interactive systems, this can lead to inaccurate simulations of users and thus, unreliable results. We identify this as the simulators' lack of *individual fidelity* – the ability to capture behavior unique to the intended user. We therefore develop ATTRIBUTION TRIALS (AT), a controlled audit for testing whether a simulator represents that individual rather than merely responding to patterns shared across users. AT compares the target user's information (the treatment) with four controls: no user information, generic population information, matched-group information, and a matched imposter's information. We apply AT to next-action prediction on real user trajectories, for a variety of simulators on three datasets: chess playing, knowledge tracing, and online shopping. Across 84 combinations of simulators, datasets, and predicted behaviors, none shows individual fidelity. Existing user simulation benchmarks barely change their scores when an imposter's information replaces the target user's. Finally, AT can guide simulator selection: on WildChat, simulators with larger AT gains tend to generate text that moves closer to the user's real turn when given the target user's profile instead of an imposter's. Together, these findings show that apparent behavioral similarity can hide limited individual fidelity, and that AT can help choose the simulators that come closest to it.

1 INTRODUCTION

As interactive AI systems become more personalized, a growing body of work uses user simulators to train and evaluate them (Yao et al., 2025; Sun et al., 2026; Abdulhai et al., 2025; Dou et al., 2025). These simulators are typically evaluated by comparing their behavior with observations from real users (Zhu et al., 2026; Yang et al., 2026; Zhou et al., 2026b). Some benchmarks measure population-level realism (Hu et al., 2026), asking whether simulated users reproduce aggregate outcomes (Meister et al., 2025), behavioral features (Zerhoudi, 2026), or conversation distributions (Mehri et al., 2026). Others test whether conditioning on user information shifts predictions away from an unconditioned or population-level baseline (Zhu et al., 2026; Dou et al., 2025; Naous et al., 2026; Yang et al., 2026; Jiang et al., 2025), or whether preferences, rankings, and task outcomes obtained with simulated users transfer to real users (Dou et al., 2025; Zhou et al., 2026b; Suh et al., 2026).

However, none of these benchmarks asks whether a simulator behaves like the particular user it was given. A user simulator can look realistic in aggregate, change its behavior when given user information, and raise a downstream score, yet behave the same way for any comparable user. Figure 1(a) shows this for how long online shoppers in OPeRA (Wang et al., 2026) wait before their next action: real users spread widely, but four simulators given each user's own information place their users in narrow regions away from the real-user average. The crucial distinction is between *steering*, any change of behavior in response to supplied user information, and *individual fidelity*: the degree to which a simulator preserves behavioral differences *specific* to the intended user.

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

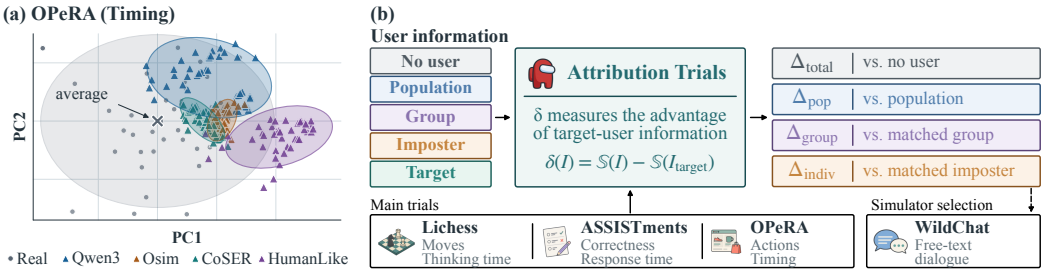

Figure 1: **Does a user simulator preserve individual fidelity?** (a) Waiting times of 41 OPeRA shoppers before their next action, real and predicted by four simulators given each user's own information, projected onto the real users' first two principal components. (b) AT compares the target user's information with four controls, and a simulator passes only when all four gains are significant (Section 3); WildChat is the simulator-selection study (Section 5.3).

To investigate this problem, we propose ATTRIBUTION TRIALS (AT)[1], a controlled audit for measuring individual fidelity in user simulators. Our audit consists of a treatment condition (the target user's information) and four control conditions (alternatives to that user): no user information, generic population information (representing patterns shared across all users), matched-group information (representing patterns shared among similar users) and information from a matched imposter (representing a comparable but different individual). AT evaluates a user simulator under each condition and reports the *gain* of the target user's information over each of the four control conditions (Figure 1(b), Section 3).

We demonstrate AT on next-action prediction (Shaikh et al., 2026) across three datasets: chess games, knowledge tracing, and online shopping (Section 4). None of the 84 combinations of simulator, dataset, and predicted behavior shows individual fidelity. Prompting LLMs, the most common recipe for user simulation, is the least faithful approach (Section 5.1). Frontier API models (GPT-4.1, GPT-4.1-mini, DeepSeek-V3.2) also fail to separate the target user from the imposter (Appendix D). Lastly, on WildChat, simulators that score higher on AT also gain more from the target user's profile when generating text (Section 5.3).

Our contributions are:

1. **Attribution Trials**, a controlled audit that credits a simulator with individual fidelity only when the target user's information beats all four controls significantly (Section 3).
2. **Current simulators do not show individual fidelity**: none of 84 combinations passes, and where user information helps, generic population information helps about as much (Section 5.1).
3. **Existing benchmarks miss imposters**: their scores barely move when a matched imposter's information replaces the target's (Section 5.2).
4. **Simulator selection for text generation**: on WildChat, the target-versus-imposter gain correlates with how much the target's profile moves a simulator's generated text toward the real turn (Kendall's τ up to 0.62) (Section 5.3).

We will release the complete Attribution Trials implementation, controlled synthetic populations, evaluation score tables, and data preprocessing and reproduction scripts.

2 RELATED WORK

Simulator benchmarks compare simulated and real users through population outcomes, behavioral features, or conversation distributions (Zhou et al., 2026b; Yang et al., 2026; Hu et al., 2026; Mehri et al., 2026); persona benchmarks probe synthetic histories (Jiang et al., 2025; Li et al., 2026);

[1]'Attribution' is to motivate 'why' a user simulator behaves a certain way, and 'Trials' is inspired by randomized controlled trials (RCTs) that are very common in understanding interventions in medicine.

RealUserSim, TwinVoice, and CustomerSim add logged profiles or simulator training (Zhu et al., 2026; Du et al., 2026; Pruksachatkun et al., 2026); and SimEval-IR separates realism from ranking validity (Zerhoudi, 2026). Two of these vary the person: BehaviorBench substitutes an unrelated profile in an ablation, and TwinVoice draws the distractors of its multiple-choice task from other users' real replies. The first alternative is unmatched and reported without a paired test; the second varies the answer options rather than the information the simulator is given. Neither retrains the subject under each condition nor fixes an identity verdict in advance, and the remaining designs compare supplied information against none or against an aggregate. Table 1 scores these metrics on our combinations and Table 12 compares their documented designs. Trained simulators, persona generators, and long-horizon memory systems are complementary method classes for this audit (Zhou et al., 2026a; Naous et al., 2026; Paglieri et al., 2026; Yan et al., 2026; Platnick et al., 2025; Meshi et al., 2026).

Simulator utility studies ask whether simulated interactions support decisions on real users. SimulatorArena tests whether simulator-mediated rankings track human rankings (Dou et al., 2025). Other work uses downstream utility, purchase outcomes, or intervention stability (Suh et al., 2026; Chen, 2026; Hullman et al., 2026; Lin et al., 2026). Tool-using agents are already post-trained against user simulators (Gao et al., 2026). Attribution Trials adds a matched imposter, retraining under each expressible condition, and a prespecified identity verdict. Its simulator-selection study tests whether the target-versus-imposter gain predicts generated behavior.

3 Attribution Trials

We introduce our controlled audit, *Attribution Trials* (AT). The audit works on a per user simulator level. Each user simulator will undergo one treatment condition and four control conditions (Section 3.1). From these we calculate the gain that user simulators show for the treatment condition, when compared to control conditions (Section 3.2). We then combine this to define the individual fidelity of a user simulator.

3.1 Control and Treatment Conditions

Assume a user simulator $\mathcal{M}$ that predicts a given user u's next action, given some task information $\mathcal{T}$, and some user-level information I^u. $\mathcal{T}$ would contain the task instructions, along with past user history as context. Depending on the variant we try, I^u could be a structured user profile or a prior interaction trajectory. We then define the following variants of I^u[2]: I_{target}(treatment), corresponds to u's information, $I_{\text{no-user}}$(control) keeps the task context and removes every user-specific input, $I_{\text{population}}$(control) generic population information, built from the pooled behavior of all users, I_{group}(control) matched-group information, built from the users in the target's group, I_{imposter}(control) information of one *matched imposter*, which is another user similar to u.

3.2 Measuring individual fidelity using treatment and control conditions

We use the treatment and control conditions to observe the next action probability of a given user simulator for a given user. Let $y_{I,u} = p_I(\cdot \mid \mathcal{T}, I)$ be the next action probability of $\mathcal{M}$ for the given task $\mathcal{T}$ and I for u. We score the observed behavior $y_{I,u}$ by its *surprisal*, the negative log predictive probability (Hale, 2001; Gneiting & Raftery, 2007). For held-out observations O_u for u, we define mean *surprisal*

$$\mathbb{S}(I; u) = \frac{1}{|O_u|} \sum_{o \in O_u} -\log p_I(y_{I,u} \mid \mathcal{T}, I^u). \tag{1}$$

Intuitively, lower surprisal implies that the simulator assigns greater likelihood to the behavior that actually occurred. Surprisal is then averaged over all users $\mathcal{U}$. We note that a low surprisal under I_{target} does not by itself show that the simulator represents this user because another user might also lead to the same surprisal[3].

[2]For brevity, we abuse the notation slightly, where we let I to represent I^u unless specified otherwise.

[3]Surprisal as a value is comparable across conditions within a pair of $\mathcal{M}$ and $\mathcal{T}$ but not across all settings, whose inputs and tokenizers differ, so every claim in the paper compares conditions for each pair.

We thus measure the *gain*, $\delta(I)$ that we observe from the treatment condition over the other control conditions, on the same held-out observations, for all users:

$$\delta(I) = \frac{1}{|U|} \sum_{u \in U} \left[\mathbb{S}(I; u) - \mathbb{S}(I_{\text{target}}; u) \right] \tag{2}$$

A *positive gain* means that the treatment condition predicts the user's next action better than the given control condition. We compute it against each control condition, which gives four gains: $\Delta_{\text{total}} = \delta(I_{\text{no-user}})$, $\Delta_{\text{pop}} = \delta(I_{\text{population}})$, $\Delta_{\text{group}} = \delta(I_{\text{group}})$, and $\Delta_{\text{indiv}} = \delta(I_{\text{imposter}})$, which answer distinct questions. Δ_{total} asks whether the target user's information helps at all; Δ_{pop} whether it helps beyond generic population information; Δ_{group} whether it helps distinguish the user within a similar group of users; and Δ_{indiv} whether it helps more than a comparable but different user. User simulators can show one gain and not another, so we report all four rather than combining them into one score, and it helps attribute a particular failure mode to a user simulator.

Individual fidelity. A simulator $\mathcal{M}$ shows *individual fidelity* on a task $\mathcal{T}$ if and only if all four gains are significant: we average per-user gains over three training seeds, bootstrap users 10,000 times, and call a gain significant when its 95% interval lies above zero after Holm-Bonferroni correction (Holm, 1979) over the 84 combinations.

4 Experiment Setup

We split each user's history temporally: the first 70% is used to train the simulators or used in-context and the last 30% is used for evaluations. User information (profiles, embeddings, memory) and any other training-only context is fixed at evaluation time. The simulator then predicts each held-out next action in order, and we score it against the real user action.

Datasets. We demonstrate AT across three real-user datasets with varying levels of complexity and user behavior: real user gameplay in chess from Lichess (Lichess, 2025), real student knowledge tracing behavior in math assessments from the ASSISTments corpus (Feng et al., 2009) and online shopping behavior from OPeRA (Wang et al., 2026). Lichess, ASSISTments, and OPeRA have 299, 500, and 41 users, respectively. Each of these datasets contains fully timestamped real user trajectories.

Behavior Channels. A user simulator needs to predict the next action to be taken by the real user. However, the next action is not only the actual action the user took, but also the *time* they took for that action. For example, two chess players, a novice and an expert could take the same next move, but the novice might take more time to think about the move, which is an important behavioral characteristic of the novice. We therefore decompose each next action into two *behavior channels*: the actual next step taken and the time taken, both of which have surprisals computed on a different scale (a discrete choice vs. a continuous duration). For Lichess, we split the next action into the chess move and the time taken to think before it, for ASSISTments, we use the actual response and response time, and for OPeRA, we use the next shopping action and the timing taken to do it as the two channels.

User Simulators. We use several strategies to train user simulators, along with state-of-the-art LLMs trained explicitly as simulators. We evaluate the following five method families and three released simulators, each taking user information in its own form: static embedding (a learned vector per user, the per-user models of McIlroy-Young et al. (2022)), recurrent embedding (a per-user state updated along the history and frozen after training, as in deep knowledge tracing (Piech et al., 2015)), structured memory (per-user statistics, the training-free memory of Zhong et al. (2023); Chhikara et al. (2025)), user-profile LoRA (Hu et al., 2022; Tan et al., 2024) (a textual profile read through a shared adapter), frozen prompt (a fixed language model with the profile in its prompt, the form most often used to train and evaluate interactive agents (Yao et al., 2025; Sun et al., 2026; Abdulhai et al., 2025); Qwen3 at 1.7B, 8B, and 30B-A3B (Yang et al., 2025)), and the natively-trained user simulators Osim, CoSER, and HumanLike (Zhou et al., 2026a; Wang et al., 2025; Çalık & Akkuş, 2026).

Overall, AT is applied over 84 {simulator, dataset, behavior channel} combinations – 8 for chess, 10 for knowledge tracing, and 66 for OPeRA. For OPeRA, we additionally experiment with different

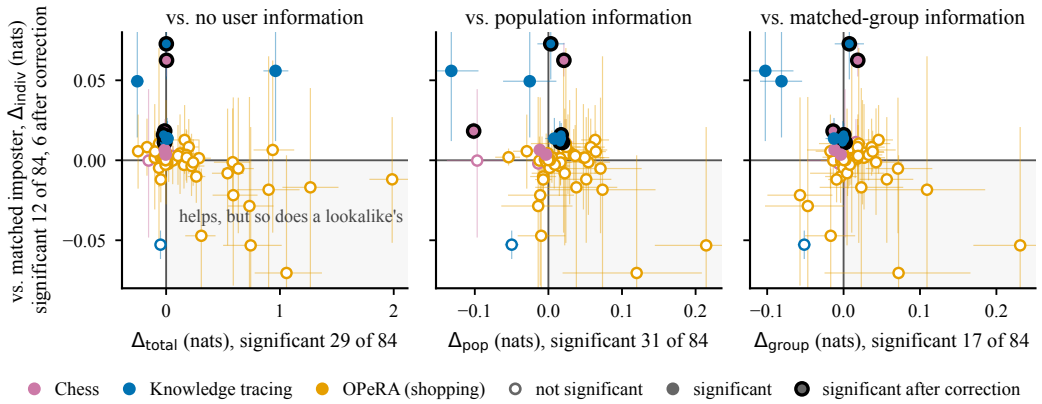

Figure 2: **Gains of the target user's information over each control condition.** Each point is one of the 84 combinations of method, dataset, and behavior channel, with 95% paired-user bootstrap intervals (shown by the lines). The vertical axis of every panel is the gain against the matched imposter's information; the horizontal axis is the gain against no user information (left), generic population information (middle), or matched-group information (right). An ideal simulator should be on the top right.

shopper profiles: the shopper's survey persona (static), running statistics of the shopper's own behavior (dynamic), or both (combined) (more details in Appendix B.3).

5 Results

5.1 Analysing AT audits

Figure 2 plots gains for all 84 {simulator, dataset, behavior channel} combinations, comparing Δ_{indiv} with Δ_{total}, Δ_{pop}, and Δ_{group}, respectively. Table 8 in the appendix shows the numerical values for each point in the plot.

The treatment condition rarely steers user simulators towards the actual user. As seen in Figure 2, some simulators appear to the 'right' of the plot, i.e., the target's information I_{target} predicts better than $I_{\text{no-user}}$, $I_{\text{population}}$, or I_{group}: 29 of the 84 combinations are significant against $I_{\text{no-user}}$ and 31 against $I_{\text{population}}$. However, simulators do not distinguish the target user from the matched imposter nearly as well: only 6 of the 84 combinations are significant against the imposter after correction. No combination is significant on all four gains: 12 have all four gains positive, but at most three are significant after correction. All 6 are small models that learn a per-user embedding or state, and none of the 74 language-model combinations is among them. In four of the six, the target's information predicts worse than no user information at all ($\Delta_{\text{total}} < 0$), so the separation comes from the imposter's information hurting more than that of the target, not from the target's information actually helping the simulator.

What the simulators do instead is population-level simulation. In most of the 29 combinations where the target's information helps against no user information, generic population information does almost as well, recovering a median of 97% of the improvement. In 17 of the 29, the target's information is no better than the population's. In the other 12, where it is better, switching to a matched imposter's information is just as good in all 12. The simulators improve when given any user's information, and it hardly matters whether it is the right user's.

The few simulators that beat the imposter mostly capture the user's average behavior. A baseline that uses no simulator and predicts each user only from the distribution of that user's own past behavior separates the target from the imposter as well as the simulators do (Appendix C.3). It matches or exceeds the simulator in 9 of the 12 combinations significant against the imposter, including all 5 static-embedding combinations significant after correction. Only three combinations, all on knowledge-tracing response, add anything beyond this baseline. Frozen prompt at 1.7B and 8B exceed it by 0.044 and 0.038 nats, and recurrent embedding by 0.003 nats.

Table 1: **Existing benchmarks barely change their score when the simulator is given the matched imposter's information.** Left: how many of the 84 combinations the evaluation judges significant, over the combinations it can be computed on (52 for accuracy and total variation, 5 for the random-profile control; n/a where none). Right: the same score recomputed with I_{imposter} in place of I_{target}, as the combinations still significant and the score kept (Section 5.2); Attribution Trials keeps 0% by construction. In parentheses: after Holm-Bonferroni correction.

	Combinations judged significant				With I_{imposter} for I_{target}	
Evaluation method	Chess	KT	OPeRA	All	Still sig.	Score kept
Perplexity of real behavior (Naous et al., 2026; Zhou et al., 2026a)	n/a	n/a	n/a	n/a	n/a	99.9%
Profile versus no profile (Zhu et al., 2026; Naous et al., 2026)	1/8	1/10	27/66	29/84	29/29	100.6%
Random unrelated profile (Yang et al., 2026)	2/2	3/3	n/a	5/5	3/5	11.6%
Next-decision accuracy (Yang et al., 2026; Jiang et al., 2025)	2/2	2/2	19/48	23/52	21/23	96.2%
Group-level total variation (Hu et al., 2026; Meister et al., 2025)	2/2	2/2	19/48	23/52	23/23	98.8%
Population realism (Mehri et al., 2026; Zerhoudi, 2026)	n/a	n/a	n/a	0/21	n/a	99.9%
Attribution Trials (ours)	4/8 (2)	8/10 (4)	0/66 (0)	12/84 (6)	0/12	0.0%

Zero-shot prompted user simulators are the most common and the least faithful. Putting a profile in the prompt of a fixed language model is how interactive agents are most commonly trained and evaluated (Yao et al., 2025; Sun et al., 2026; Abdulhai et al., 2025; Dou et al., 2025; Gao et al., 2026). On the 22 frozen-prompt combinations, the target's profile predicts significantly better than no user information in 14, but a matched imposter's profile improves over no user information by almost the same amount in all 14. Against the imposter, only two combinations for the knowledge-tracing task are significantly better ($\Delta_{\text{indiv}} > 0$), for user simulators with 1.7B and 8B models (neither after correction). For prompted user simulators, the target user profile can also hurt steering towards the user: in 5 of the 22 combinations, four of them for the timing channel, the simulator predicts worse than no user information. We also observe that scaling from 1.7B to 30B-A3B does not change simulator behavior.

5.2 Existing user simulation benchmarks vs. AT

Existing user simulator evaluations score a simulator by how well its behavior matches real users. We compute each benchmark's score twice on the same 84 combinations: once with the target's information and once with a matched imposter's (Table 1). If a benchmark measured individual fidelity, its score should drop when the profile belongs to the imposter rather than the target user. *Score kept* is the imposter score as a percentage of the target score, so 100% means the benchmark is not robust to the profile swap.

Existing evaluations reward presence of any user information, rather than an individual. Existing evaluations find many more combinations significant than Attribution Trials does. Comparing the simulator with and without user information finds the same 29 of 84 (Zhu et al., 2026; Dou et al., 2025; Naous et al., 2026; Yang et al., 2026; Meister et al., 2025). Next-decision accuracy finds 23 of the 52 combinations it applies to (Yang et al., 2026; Jiang et al., 2025), and group-level total variation also finds 23 of 52 (Hu et al., 2026; Meister et al., 2025). Attribution Trials finds 12 significant against the matched imposter. The difference is that their scores rise whenever user information helps, whoever the information belongs to.

Existing evaluations are not robust to imposter profiles. When the matched imposter's information is used in place of the target's, the perplexity of real behavior changes by a median of 0.1% (Naous et al., 2026; Zhou et al., 2026a). Comparing with and without user information keeps all 29 of its significant combinations and 100.6% of its score. Next-decision accuracy keeps 21 of its 23 and 96% of its score, and group-level total variation keeps all 23 and 99%. Population realism cannot notice the swap at all, because its score does not depend on whose information the simulator is given (Mehri et al., 2026; Zerhoudi, 2026). Only one existing control reacts: replacing the profile with a random unrelated user's (Yang et al., 2026), which can be computed on 5 combinations. Its score drops to

Table 2: **Using the target profile makes generated turns more similar to the user's real next turn.** Gains Δ_{indiv} and Δ_{total} are from the 240-user attribution panel (Section 5.3); $\bar{g}$ and W from 1,427 turns of 236 users, defined in Section 5.3. Intervals are in Tables 13 and 14.

	Gain (nats)		avg@8		best@8	
Simulator	Δ_{indiv}	Δ_{total}	$\bar{g}$	W (%)	$\bar{g}$	W (%)
HumanLike-7B	$+0.956$	$+1.252$	$+0.117$	76.3	$+0.135$	73.3
Qwen3-1.7B	$+0.569$	$+0.135$	$+0.105$	72.5	$+0.115$	68.5
Qwen3-8B	$+0.651$	$+0.546$	$+0.103$	75.3	$+0.133$	74.1
Qwen3-32B (dense)	$+0.630$	$+0.621$	$+0.102$	76.3	$+0.131$	75.7
Osim-8B	$+0.528$	$+0.447$	$+0.067$	70.8	$+0.107$	73.0
HumanLM-opinion	$+0.524$	$+0.561$	$+0.065$	66.8	$+0.087$	66.9
CoSER-8B	$+0.632$	$+0.505$	$+0.046$	67.4	$+0.086$	69.1

12%, but it still finds 3 of the 5 significant, because a matched imposter is more similar to the target than a random user is. Existing evaluations can tell the target from a random stranger, but not from a similar user.

5.3 Selecting simulators with AT

While AT requires user simulators to predict the next action, user simulators are very common in conversational tasks, where they are used as a stand-in for real users while training models for a certain task. Here, we test whether AT helps choose among such simulators: whether simulators that score higher on AT also move their generated text closer to what the user actually wrote when given the target user's profile.

Evaluation. In order to evaluate this, we use WildChat (Zhao et al., 2024), a corpus of real-user free-text conversations with an AI system. We run AT on WildChat for seven user simulators, and then use the same simulators to generate the user's next turn. For each real user turn, every simulator generates the user's next turn twice, once given the target's profile and once given the matched imposter's, and each generated turn is scored by its distance d to the user's real next turn: the cosine distance between MiniLM sentence embeddings (Reimers & Gurevych, 2019; Wang et al., 2020) (Appendix E.1.1). *Content gain* $\bar{g}$ is how much closer the target-profile reply is than the imposter-profile reply, averaged over a user's turns and then over users; a positive value means the target's profile pulls the generated text toward what the user actually wrote. *Target win rate* W is the share of turns on which the target-profile reply is the closer of the two, a tie counting as half a win. For each simulator and profile, we generate eight replies per turn and report avg@8 and best@8. To compare two rankings of the simulators, we report *agree*, the number of simulator pairs both rankings put in the same order (counting only pairs whose content-gain difference is statistically clear), and Kendall's τ, the rank correlation over all 21 pairs. On 1,444 turns from 240 users, with fixed target and imposter profiles, all seven simulators are significant against both the matched imposter and no user information (Table 2, left). HumanLike-7B has the highest Δ_{indiv} across the chosen simulators. We also observe that overall, every simulator has a positive content gain, i.e., the target profile moves generated text closer to the real turn for every simulator, with win rates from 66.8% to 76.3%.

Δ_{indiv} is positively correlated with the content gain. We observe that Δ_{indiv} of the simulators in AT has moderate to high correlation with the content gain in WildChat, with Kendall's $\tau = +0.43$ for avg@8 and $+0.62$ for best@8. Every disagreement involves CoSER, whose generations are often empty (21.6% end at the first token). The correlation holds within users too: for six of the seven simulators, users with larger AT Δ_{indiv} also have larger content gains (Spearman $+0.23$ to $+0.32$, Appendix E.1.1).

Other selection rules. Table 3 compares AT with other ways of choosing a simulator. Ordering by the gain from supplying any profile (Zhu et al., 2026; Dou et al., 2025; Naous et al., 2026) recovers less of the content-gain ordering at both endpoints. Ordering by how close the target-conditioned samples land to the real turn (Wu et al., 2026) depends on the endpoint: at avg@8 it tracks the ordering as well as the trial, at best@8 it is anti-correlated, and both versions pick a mid-field

Table 3: **Rules for choosing a simulator, scored against the content-gain ordering of Table 2.** Agree and τ as defined in Section 5.3. Only the Attribution Trials row was preregistered, and seven simulators do not separate the rows statistically.

	avg@8		best@8	
Selection rule	Agree	τ	Agree	τ
Profile versus no profile (Zhu et al., 2026; Naous et al., 2026)	10/15	$+0.14$	11/16	$+0.33$
Absolute similarity, avg@8 (Wu et al., 2026)	12/15	$+0.43$	12/16	$+0.43$
Absolute similarity, best@8 (Wu et al., 2026)	7/15	-0.24	10/16	-0.05
Population distribution distance (Mehri et al., 2026; Zerhoudi, 2026)	10/15	$+0.05$	12/16	$+0.24$
Judged fidelity (Zhu et al., 2026; Wu et al., 2026)	12/15	$+0.52$	11/16	$+0.33$
Attribution Trials (ours)	11/15	$+0.43$	13/16	$+0.62$

simulator (Qwen3-8B or Osim-8B) where the trial picks HumanLike-7B, the largest content gain. Ordering by how real the population of generated turns looks, the Fréchet distance between generated and real embedding clouds (Mehri et al., 2026; Zerhoudi, 2026), has τ of only $+0.05$ at avg@8 and picks Osim-8B, whose samples are hardest to tell from real text yet rank fifth in content gain; that distance moves by at most 5.4% when the imposter's profile replaces the target's, except for HumanLike-7B (21%). Asking an LLM judge (Zhu et al., 2026; Wu et al., 2026) to score each generated turn against the real one, blind to which profile was used, orders the simulators about as well as the trial (12 of 15 pairs and τ $+0.52$ at avg@8, 11 of 16 at best@8) but picks Qwen3-1.7B, second in content gain. Unlike the discrete-label benchmarks of Section 5.2, the judge does register the imposter's profile: its score falls for every simulator when the imposter's profile replaces the target's, keeping between 34% and 87% of its value (Table 16).

6 Discussion

Does AT detect individual signal when it exists? A test that never finds anything would also report "no individual fidelity", so we first check that the trial finds it when it is present and does not when it is absent. To test this, we make use of PersonaMem (Jiang et al., 2025) and HorizonBench (Li et al., 2026) that are built around facts about specific users, and on both the target's own history does significantly better than the matched imposter's. To control how much individual signal exists, we also simulate our own synthetic users, unrelated to these benchmarks, whose behavior mixes a pattern shared by everyone with a pattern of their own, and we set the share λ of the individual pattern. With $\lambda = 0$ there is no individual signal, and the trial finds a significant gain in 0 of 200 populations; as λ grows, the gain grows steadily (Appendix F.2).

What does each part of the trial contribute? To see this, we change one part at a time (Table 4). For each change we report two numbers: how many of the 84 real combinations are significant after correction, and how often the trial finds a significant gain on synthetic users with no individual signal, where every such finding is a false positive. We change four parts: replacing the imposter with a weaker control, choosing a closer imposter, updating user information as the trajectory proceeds, and mixing the target's information into the imposter's. We first ask, *can a simpler control replace the matched imposter?* It cannot. When users differ only by group, a simulator beats no user information and population information just by learning the user's group: on such synthetic users these two controls give a significant gain in 85% and 100% of populations. A matched imposter comes from the same group, so it gives one in only 3%, within chance. Group information also passes this check, but on real data only 1 of its 6 significant combinations is significant against the imposter, since one user's information can beat a group average without being the target's. *What about imposters more similar to the target user?* The matched imposter shares the target's matching cell, so differences finer than that cell still count as individual: on synthetic users that differ only by such finer groups, the trial fires in 89% of populations. Choosing the imposter as the nearest user on five behavioral features instead lowers this to 0.5%, and only 3 of the 6 real combinations significant after correction remain so (Appendix F.3). Next, we ask, *why fix both users' information before the evaluation?*

Table 4: **Removing or replacing one element of the trial at a time.** *Real data*: combinations of 84 significant against the variant's comparison after Holm correction, the one gain the variant changes (Section 3.2); no combination passes the full rule. *Synthetic*: false-positive rate over 200 populations with no individual signal, so anything above the nominal 5% is a failure: users drawn from one shared rate; that rate drifting across the training boundary (worst of three magnitudes); users differing only by the group the rule matches on; and groups finer than the matching cells. *Of the 6 combinations significant after correction under the registered rule, those still significant under the nearest-neighbor rule, which was recomputed on nine combinations. n/a: not applicable or not run. Settings in Appendix F.2.

	Real data	Synthetic population, no individual signal			
Variant	Significant after correction / 84	Shared rate	+ drift	+ group	+ finer group
Attribution Trials (full)	6	0.000	0.000	0.030	0.890
Compared with: no user information	24	0.000	0.000	0.850	0.850
Compared with: generic population	13	0.015	0.005	1.000	1.000
Compared with: matched group	6	0.000	0.000	0.000	0.520
Information updated with held-out rows	16	0.000	1.000	n/a	n/a
Matching rule: five-feature nearest neighbor	3 of 6*	n/a	n/a	n/a	0.005

If the target's information is updated as the trajectory proceeds while the imposter's stays fixed, 16 combinations are significant after correction instead of 6. But the extra combinations are not evidence of individual fidelity. On synthetic users built with no individual differences, whose shared behavior slowly changes over time, the updated variant is significant in every population and the fixed version in none. The updated information is simply more recent, so it tracks the shared change, not the user. We therefore fix both users' information at the end of the training period (Appendix F.4). Lastly, we consider *what happens when we mix the target user's information into the imposter's?* Mixing the target's fixed information into the imposter's lowers frozen-prompt Qwen3-8B surprisal monotonically on knowledge-tracing response and chess timing, by 3.3% and 4.9% end to end (Appendix F.5). The gain therefore responds to how much of the target's information the simulator receives, not only to whether it receives it.

7 Conclusion

Attribution Trials tests whether a simulator represents the intended individual by comparing the target user's information with four other conditions. No combination of 84 passes: 6 separate the target from a matched imposter after correction, but in 4 of them the target's information predicts worse than no user information, in 5 a per-user baseline built from past behavior alone separates the two at least as strongly, and the last loses significance under finer matching. Scored the way existing benchmarks score them, those same combinations barely move when the simulator is given the matched imposter's information, which is why generic conditioning gains are not evidence of individual fidelity. On WildChat the trial's ordering predicts the target-profile effect on generated text.

AI use statement

In this work, we used generative AI tools, primarily Codex and Claude Code, to write code that implements methods, runs experiments and analyses selected by the authors, and parses experimental output logs. We have not used generative AI tools to develop research ideas or hypotheses or to formulate mathematical claims or proofs; the intellectual ideas and choice of analyses originated with the authors. Additionally, we used generative AI tools to improve the paper's readability. We did not use generative AI tools for literature searches. The language models evaluated as user simulators are part of the experimental methodology described in the paper. The authors manually verified all written text and citations and checked all experiments for correctness. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Ethics statement

All human data come from existing interaction logs: public chess games, a de-identified educational dataset, the OPeRA shopping-behavior corpus (Wang et al., 2026), and a public consented chat corpus (Zhao et al., 2024). We collect no new data and do not attempt re-identification. Because the trial measures whether a profile can be matched to its owner, its artifacts are re-identification-adjacent and the release is scoped for that. The release will include the complete implementation and per-combination score tables whose per-user keys will be replaced by release-local pseudonyms under an unpublished mapping—every verdict and correction family is invariant to relabeling, and the chess cohort's native keys are public account names—and it will not include the underlying records, the rendered profiles, the WildChat conversation text, or the donor cross-matrices from which per-user identification ranks could be recomputed. Those stay with the raw data under the source datasets' terms, and the sparsest cohorts, 41 OPeRA users and a 30-user timing panel, will appear only in aggregate. The trial is intended to identify cases in which simulators used in place of real users misrepresent the individuals they are meant to simulate.

References

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

Liang Chen. Simulated customers never walk away: Decision fidelity of llm user simulators measured against real purchase outcomes, 2026. URL https://arxiv.org/abs/2606.20708.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory, 2025. URL https://arxiv.org/abs/2504.19413.

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

Bangde Du, Minghao Guo, Songming He, Ziyi Ye, Xi Zhu, Weihang Su, Shuqi Zhu, Yujia Zhou, Yongfeng Zhang, Qingyao Ai, and Yiqun Liu. TwinVoice: A multi-dimensional benchmark towards digital twins via LLM persona simulation. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 19604–19628, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.981. URL https://aclanthology.org/2026.findings-acl.981/.

Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, 2009. ISSN 1573-1391. doi: 10.1007/s11257-009-9063-7. URL https://doi.org/10.1007/s11257-009-9063-7.

Jiaxuan Gao, Jiaao Chen, Chuyi He, Shusheng Xu, Di Jin, and Yi Wu. From self-evolving synthetic data to verifiable-reward rl: Post-training multi-turn interactive tool-using agents, 2026. URL https://arxiv.org/abs/2601.22607.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

John Hale. A probabilistic Earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*, 2001. URL https://aclanthology.org/N01-1021/.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. ISSN 03036898, 14679469. URL http://www.jstor.org/stable/4615733.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Tiancheng Hu, Joachim Baumann, Lorenzo Lupo, Nigel Collier, Dirk Hovy, and Paul Röttger. Simbench: Benchmarking the ability of large language models to simulate human behaviors. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PL51SpN6ZJ.

Jessica Hullman, David Broska, Huaman Sun, and Aaron Shaw. This human study did not involve human subjects: Validating llm simulations as behavioral evidence, 2026. URL https://arxiv.org/abs/2602.15785.

Bowen Jiang, Zhuoqun Hao, Young Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo Jose Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=6ox8XZGOqP.

Shuyue Stella Li, Bhargavi Paranjape, Kerem Oktar, Zhongyao Ma, Gelin Zhou, Lin Guan, Na Zhang, Sem Park, Lin Chen, Diyi Yang, Yulia Tsvetkov, and Asli Celikyilmaz. HorizonBench: Long-horizon personalization with evolving preferences, 2026. arXiv:2604.17283.

Lichess. Lichess open database. https://database.lichess.org, 2025. CC0 public-domain dedication.

Victoria Lin, Taedong Yun, Maja Mataric, John Canny, Arthur Gretton, and Alexander Nicholas D'Amour. The illusion of intervention: Your LLM-simulated experiment is an observational study. In *ICML 2026 Statistical Frameworks for Uncertainty in Agentic Systems*, 2026. URL https://openreview.net/forum?id=7EZrgYo07P.

Reid McIlroy-Young, Russell Wang, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Learning models of individual behavior in chess. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 1253–1263, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393850. doi: 10.1145/3534678.3539367. URL https://doi.org/10.1145/3534678.3539367.

Shuhaib Mehri, Philippe Laban, Sumuk Shashidhar, Marwa Abdulhai, Sergey Levine, Michel Galley, and Dilek Hakkani-Tür. Measuring and mitigating the distributional gap between real and simulated user behaviors, 2026. URL https://arxiv.org/abs/2605.07847.

Nicole Meister, Carlos Guestrin, and Tatsunori Hashimoto. Benchmarking distributional alignment of large language models. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 24–49, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.2. URL https://aclanthology.org/2025.naacl-long.2/.

Ofer Meshi, Krisztian Balog, Sally Goldman, Avi Caciularu, Guy Tennenholtz, Jihwan Jeong, Amir Globerson, and Craig Boutilier. ConvApparel: A benchmark dataset and validation framework for user simulators in conversational recommenders. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5270–5304, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-380-7. doi: 10.18653/v1/2026.eacl-long.244. URL https://aclanthology.org/2026.eacl-long.244/.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the dialogue: Training and evaluating user language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ykSmkVqzn4.

Davide Paglieri, Logan Cross, William A. Cunningham, Joel Z. Leibo, and Alexander Sasha Vezhnevets. Persona generators: Generating diverse synthetic personas for arbitrary contexts, 2026. URL https://arxiv.org/abs/2602.03545.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/bac9162b47c56fc8a4d2a519803d51b3-Paper.pdf.

Daniel Platnick, Mohamed E. Bengueddache, Marjan Alirezaie, Dava J. Newman, Alex "Sandy" Pentland, and Hossein Rahnama. Id-rag: Identity retrieval-augmented generation for long-horizon persona coherence in generative agents, 2025. URL https://arxiv.org/abs/2509.25299.

Yada Pruksachatkun, Yixin Wan, Xingrun Chen, Kai-Wei Chang, and Chien-Sheng Wu. Customersim: Benchmarking and aligning multimodal language models as retail user simulators, 2026. URL https://arxiv.org/abs/2605.08334.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Omar Shaikh, Valentin Teutschbein, Kanishk Gandhi, Yikun Chi, Nick Haber, Thomas N. Robinson, Nilam Ram, Byron Reeves, Sherry Yang, Michael S. Bernstein, and Diyi Yang. Learning next action predictors from human-computer interaction. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=SX7AUEqs4L.

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

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

Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. Democratizing large language models via personalized parameter-efficient fine-tuning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 6476–6491, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.372. URL https://aclanthology.org/2024.emnlp-main.372/.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Xintao Wang, Heng Wang, Yifei Zhang, Xinfeng Yuan, Rui Xu, Jen tse Huang, Siyu Yuan, Haoran Guo, Jiangjie Chen, Shuchang Zhou, Wei Wang, and Yanghua Xiao. CoSER: Coordinating LLM-based persona simulation of established roles. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=BOrR7YqKUt.

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, Yu Su, Upol Ehsan, Malihe Alikhani, Toby Jia-Jun Li, Lydia Chilton, and Dakuo Wang. OPeRA: A dataset of observation, persona, rationale, and action for evaluating LLMs on human online shopping behavior simulation. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43942–43960, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.2033. URL https://aclanthology.org/2026.acl-long.2033/.

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

Shannan Yan, Jingchen Ni, Leqi Zheng, Jiajun Zhang, Peixi Wu, Dacheng Yin, Jing Lyu, Chun Yuan, and Fengyun Rao. Adamem: Adaptive user-centric memory for long-horizon dialogue agents, 2026. URL https://arxiv.org/abs/2603.16496.

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

Liangwei Yang, Jielin Qiu, Zixiang Chen, Ming Zhu, Juntao Tan, Zhiwei Liu, Wenting Zhao, Zhujun Lan, Akshara Prabhakar, Silvio Savarese, Huan Wang, and Shelby Heinecke. Behaviorbench: Modeling real-world user decisions from behavioral traces, 2026. URL https://arxiv.org/abs/2606.02798.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. {τ}-bench: A benchmark for \underline{T}ool-\underline{A}gent-\underline{U}ser interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=roNSXZpUDN.

Saber Zerhoudi. Simeval-ir: A unified toolkit and benchmark suite for evaluating user simulators and search sessions. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '26, pp. 3537–3543, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400725999. doi: 10.1145/3805712.3808635. URL https://doi.org/10.1145/3805712.3808635.

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

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory, 2023. URL https://arxiv.org/abs/2305.10250.

Xuhui Zhou, Weiwei Sun, Weihua Du, Jiarui Liu, Haojia Sun, Qianou Ma, Tongshuang Wu, Yiming Yang, and Maarten Sap. Odyssim: Building foundation models for human behavior simulation, 2026a. URL https://arxiv.org/abs/2606.14199.

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

Ming Zhu, Juntao Tan, Rithesh Murthy, Jielin Qiu, Liangwei Yang, Wenting Zhao, Silvio Savarese, Shelby Heinecke, and Huan Wang. Realusersim: Bridging the reality gap in agent benchmarking via grounded user simulation. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=1izVEEZAxT.

Ethem Yağız Çalık and Talha Rüzgar Akkuş. Enhancing human-like responses in large language models, 2026. URL https://arxiv.org/abs/2501.05032.

APPENDIX CONTENTS

A Attribution Trials: protocol and inference 16

B Experiment Setup: additional details 16

C Results: Attribution Trials breakdown 18

D Results: existing benchmark comparisons 22

E Results: selecting simulators 23

F Discussion: validation and ablations 26

A Attribution Trials: protocol and inference

A.1 Significance and correction

Every gain is tested with a paired per-user bootstrap (10,000 resamples); for trained methods, each user's gain is first averaged over the three training seeds. Holm correction is applied over all 84 combinations as the primary family and within each of four groups of combinations as a sensitivity check; both families leave the same six combinations significant against the imposter.

A.2 Seed-aware inference

The estimator averages each user's paired delta over the three training seeds before the user-level bootstrap, so the registered intervals condition on the realized training runs. Table 5 adds a hierarchical bootstrap that resamples seeds and users jointly for every combination that passes pooled Holm on either gain; with three seeds the seed resample is coarse, so the hierarchical interval is an inflation bound beside the user-only one rather than a replacement. Six single-seed OPeRA user-profile LoRA combinations have no seed leg and are out of scope.

Table 5: Training-seed variability in the intervals, for every combination that passes pooled Holm on either the fixed or the updated-profile gain and has three seeds. The hierarchical bootstrap resamples users and seeds jointly; LOSO is the spread of the three leave-one-seed-out points. Across these combinations the width ratio runs 1.04 to 1.72; none loses resolution under the hierarchical interval on either gain.

combination	Δ_{indiv}	user-only CI	hierarchical CI	ratio	LOSO
static · chess · move	$+0.0183$	$[+0.0128, +0.0238]$	$[+0.0107, +0.0256]$	1.35	0.0027
static · chess · timing	$+0.0625$	$[+0.0523, +0.0734]$	$[+0.0513, +0.0752]$	1.13	0.0033
static · KT · resp.	$+0.0112$	$[+0.0058, +0.0167]$	$[+0.0051, +0.0178]$	1.17	0.0024
static · KT · resp.	$+0.0113$	$[+0.0058, +0.0170]$	$[+0.0048, +0.0181]$	1.19	0.0020
static · KT · timing	$+0.0728$	$[+0.0507, +0.0951]$	$[+0.0481, +0.0985]$	1.14	0.0079
structured memory · KT · resp.	-0.0527	$[-0.0617, -0.0440]$	$[-0.0617, -0.0440]$	1.00	0.0004
recurrent embedding · chess · timing	-0.0002	$[-0.0494, +0.0452]$	$[-0.0534, +0.0522]$	1.12	0.0177
recurrent embedding · KT · resp.	$+0.0159$	$[+0.0089, +0.0232]$	$[+0.0078, +0.0249]$	1.20	0.0036
user-profile LoRA 1.7B · KT · resp.	$+0.0136$	$[+0.0012, +0.0258]$	$[-0.0003, +0.0275]$	1.13	0.0011
user-profile LoRA 8B · KT · resp.	$+0.0137$	$[+0.0027, +0.0243]$	$[+0.0016, +0.0256]$	1.11	0.0011
user-profile LoRA 1.7B · chess · timing	$+0.0036$	$[+0.0007, +0.0065]$	$[+0.0002, +0.0075]$	1.24	0.0015
user-profile LoRA 8B · chess · timing	$+0.0065$	$[+0.0021, +0.0110]$	$[+0.0012, +0.0128]$	1.29	0.0027

B Experiment Setup: additional details

B.1 Datasets and cohorts

Chess uses the Lichess open database (CC0) (Lichess, 2025); the panel contains 299 players scored on move and think time. Knowledge tracing uses ASSISTments (Feng et al., 2009) with 500 students scored on response and response time. OPeRA (Wang et al., 2026) is an online-shopping log; the pre-registered minimum-activity filter retains 41 of the 52 logged users. The WildChat panel pins dataset revision `7d6490e4` of WildChat-1M (Zhao et al., 2024), keeping English, non-toxic, non-redacted user turns grouped by hashed IP; 240 users are sampled from the 3,137 with enough activity and with later conversations on topics absent from their earlier ones. Each user's profile is a text card of their own earlier turns, up to 1,200 tokens; the held-out later conversations give 1,444 user turns, of which the generation test uses 1,427 from 236 users; imposters are a within-activity-tertile derangement.

B.2 Model and provider pins

Local backbones are the Qwen3 family (0.6B, 1.7B, 4B, 8B, dense 32B, and 30B-A3B, the last a mixture-of-experts with roughly 3B active parameters) (Yang et al., 2025). The Osim panel evaluates the released 4B and 8B models with and without midtraining (Zhou et al., 2026a). The Sim2Real ports evaluate CoSER-8B (Wang et al., 2025) and Human-Like-Qwen2.5-7B (Çalık & Akkuş, 2026); HumanLM (Wu et al., 2026) is excluded from that original audit because no approved native-contract

adapter was available there. The separate WildChat selection panel includes HumanLM-opinion under its native interface. API methods use pinned provider strings gpt-4.1, gpt-4.1-mini, and deepseek-v3.2, scored through the native letter-choice protocol; the letter parser achieved a 100% valid-parse rate on every API combination. The synthetic-population audits use gpt-5.6-luna as the audited model, with the no-history solvability baseline also measured for gpt-4.1-mini (§6).

B.3 What changes between conditions, and where identity can enter

Table 6 states the conditioning payload of each condition and what is held fixed across them. Within a method family every condition shares the backbone, the initialization, the optimizer and its hyperparameters, the schedule, the seeds, the train/eval split and the scored rows; only the payload differs. The trained families draw their per-condition models from separate training runs, with one exception recorded below.

Table 6: Conditioning payload per condition. 'retrained' means the condition has its own training run under that payload; 'scored' means the payload is swapped at evaluation on an already-trained model. *The updated variant appears only in Appendix F.4.

condition	static embedding / recurrent embedding	user-profile LoRA	frozen prompt	structured memory
$I_{\text{no-user}}$	no identity input	retrained, no profile	no profile	refit, neutral memory
$I_{\text{population}}$	population-mean trajectory	retrained, population profile	population profile	refit, population rows
I_{group}	group-mean trajectory	retrained, group profile	group profile	refit, group rows
I_{imposter}	donor's embedding or latents	scored, donor's frozen profile	donor's frozen profile	refit, donor's frozen rows
I_{target}	own latents at the boundary	scored, own frozen profile	own frozen profile	refit, own frozen rows
I_{target} updated*	latents through held-out rows	scored, updated profile	updated profile	refit, held-out rows

Profile contents. user-profile LoRA, frozen prompt, and the released simulators read the same rendered text profile, built from the user's own training-period history and fixed at the training boundary. On chess it states the player's rating, median think time, the average think time over the last five moves and on the last move, the share of moves made in under two seconds, and the think time under time pressure. On knowledge tracing it states the number of questions seen, overall and recent accuracy, whether the last answer was correct, the current streak, and a skill-adjusted trend. On OPeRA the profile takes one of three configurations: *static* renders the shopper's survey persona (age, gender, Big Five scores, shopping frequency, monthly spend, Prime membership, research-before-buying, ad attention, and holiday shopping); *dynamic* renders running statistics of the shopper's own actions (share of clicks, product views, cart size, and recent pause length); *combined* concatenates the two. The released simulators receive the same text through their native interfaces, as a system-prompt persona where the interface is chat-based. The population and group profiles render the mean of these statistics over all users or over the user's group, computed on the training split.

Hyperparameters. The per-user latent families train for 15 epochs at learning rate 0.01, batch 16, latent dimension 16, hidden dimension 64, with a timing loss weight of 0.5 and a 0.7 training fraction. user-profile LoRA uses rank 16, $\alpha = 32$, dropout 0.05 on all attention and MLP projections, AdamW at 10^{-4}, two epochs, gradient accumulation 16, sequences capped at 320 tokens in bfloat16. Frozen prompting trains nothing. Every number here is identical across the conditions of its family.

Where identity can enter, and where it cannot. $I_{\text{no-user}}$, $I_{\text{population}}$, and I_{group} are trained and scored without any individual's information or parameters, so identity has no channel into them: the population and group payloads are means over many users, computed on the training split, and the conditions are otherwise byte-identical in construction. Identity enters only through the payload column above. Two consequences are worth stating because they bound what Δ_{indiv} means. First, for user-profile LoRA and frozen prompt, I_{imposter} and I_{target} share one model and differ only in the profile pasted at scoring time, so their gain isolates the conditioning text rather than a difference in training; the retrained conditions are $I_{\text{no-user}}$, $I_{\text{population}}$, and I_{group}. Second, I_{imposter} and I_{target} are both scored with user information fixed at the training boundary; Appendix F.4 shows the recency effect that enters when I_{target} is instead updated with held-out rows.

Which gains compare across families. The table shows that the five conditions are not one homogeneous intervention across families: $I_{\text{no-user}}$, $I_{\text{population}}$, and I_{group} carry training-distribution adaptation wherever the family retrains (user-profile LoRA, structured memory, and the latent families' per-condition runs), while for user-profile LoRA and frozen prompt the $I_{\text{imposter}}/I_{\text{target}}$ leg is a score-time substitution on a single trained model. The identity gain is therefore the one leg with the same intervention semantics everywhere—a conditioning-input substitution at inference—and it is the only leg the paper compares across families. The telescoping steps $\Delta_{\text{total}} - \Delta_{\text{pop}}$, $\Delta_{\text{pop}} - \Delta_{\text{group}}$ and $\Delta_{\text{indiv}} - \Delta_{\text{group}}$ are reported within a family only, because their causal content (what was retrained versus what was swapped) differs by column above.

Timing likelihood. Timing channels score the realized time under a log-normal density on seconds, including the change-of-variable term, so the reported value is a density on the raw time scale rather than on its logarithm. The per-user marginal control for these channels fits each user's own μ and σ on the training split alone. Where a channel is bucketed instead, as in the OPeRA dwell-time channel and the chess think-time channel of the prompting families, the label space is the audit's own fixed bucket edges and the score is categorical.

B.4 Combination inventory

Table 7 lists which method runs on which behavior channel; the 84 cells of that grid are the combinations.

Table 7: **The 84 combinations.** Each cell is the number of combinations a method contributes on one behavior channel: one per channel, or three on OPeRA, where the profile comes in a static, a dynamic, and a combined configuration; 'mid' marks midtrained Osim checkpoints. Static embedding contributes a second KT response combination from a response-only model.

	Chess		KT		OPeRA	
Method	move	timing	response	timing	action	timing
static embedding	✓	✓	2	✓		
recurrent embedding	✓	✓	✓	✓		
structured memory			✓			
user-profile LoRA 1.7B		✓	✓		3	3
user-profile LoRA 8B		✓	✓		3	3
frozen prompt 1.7B		✓	✓		3	3
frozen prompt 8B		✓	✓		3	3
frozen prompt 30B-A3B					3	3
Osim 4B					3	3
Osim 4B-mid					3	3
Osim 8B					3	3
Osim 8B-mid					3	3
CoSER-8B					3	3
HumanLike-7B					3	3
Combinations	8		10		66	

B.5 Predeclared gates and screens

The leakage screen for synthetic populations makes one deterministic no-history call per item and discards correctly answered items. All acceptance gates were declared before the powered runs began. The HorizonBench cost slice is the shortest item for each of the 140 personas with the shortest histories; 120 items passed screening and 109 entered the powered runs after derangement. The Osim panel fixed its prompt format with a predeclared A/B comparison between the release formats on a held-out subset before the panel ran; the preferred format was used for every combination, and the comparison is recorded in the panel's run notes in the artifact package.

C Results: Attribution Trials breakdown

C.1 Per-combination results

Table 8 gives the target-versus-imposter gain for every audit combination and for the seven WildChat simulators of Section 5.3; Table 1 in the main text summarizes it by evaluation method.

Table 8: **All four gains for every combination.** Units: 10^{-3} nats for the 84 audit combinations, nats for the seven WildChat simulators. Each entry is a gain of Section 3.2; positive favors the target user's information. **Bold**: significant (95% interval above zero); ‡: significant after Holm correction over the 84 combinations. No combination passes (Section 3.2): no row carries ‡ on all four gains. OPeRA profiles: S = static, D = dynamic, C = combined; Osim 'mid' variants are midtrained checkpoints; KT (joint) is the static-embedding model that predicts response and timing together. –: gain not defined for that panel. Intervals for Δ_{indiv} are in Tables 9 and 13.

Method / channel	Δ_{total}	Δ_{pop}	Δ_{group}	Δ_{indiv}
Chess				
static embedding / move	-12.6	-101.5	-13.7	$\mathbf{+18.3}^{\ddagger}$
static embedding / timing	$+3.5$	$\mathbf{+21.1}^{\ddagger}$	$\mathbf{+18.5}^{\ddagger}$	$\mathbf{+62.5}^{\ddagger}$
recurrent embedding / move	-1.2	-13.8	-6.6	-2.0
recurrent embedding / timing	-154.3	-96.8	$+1.8$	-0.2
user-profile LoRA 1.7B / timing	-3.3	-3.3	-3.7	$\mathbf{+3.6}$
user-profile LoRA 8B / timing	-12.1	-11.7	-11.8	$\mathbf{+6.5}$
frozen prompt 1.7B / timing	$\mathbf{+62.3}^{\ddagger}$	$\mathbf{+18.2}^{\ddagger}$	$\mathbf{+12.8}^{\ddagger}$	$+2.7$
frozen prompt 8B / timing	-10.4	$\mathbf{+22.5}^{\ddagger}$	$\mathbf{+16.2}$	$+11.4$
KT				
static embedding (response-only) / response	-13.2	$\mathbf{+19.2}^{\ddagger}$	$+2.1$	$\mathbf{+11.2}^{\ddagger}$
static embedding (joint) / response	-14.6	$\mathbf{+16.0}^{\ddagger}$	-1.3	$\mathbf{+11.3}^{\ddagger}$
static embedding (joint) / timing	$+1.5$	$+3.2$	$+7.5$	$\mathbf{+72.8}^{\ddagger}$
recurrent embedding / response	-22.5	$\mathbf{+17.4}^{\ddagger}$	$+0.2$	$\mathbf{+15.9}^{\ddagger}$
recurrent embedding / timing	-0.2	$\mathbf{+3.0}$	$+1.3$	$+1.8$
structured memory / response	-49.7	-49.8	-51.5	-52.7
user-profile LoRA 1.7B / response	$+8.5$	$+8.1$	-12.5	$\mathbf{+13.6}$
user-profile LoRA 8B / response	$+11.9$	$\mathbf{+15.4}$	-0.9	$\mathbf{+13.7}$
frozen prompt 1.7B / response	$\mathbf{+961.4}^{\ddagger}$	-131.9	-102.4	$\mathbf{+55.9}$
frozen prompt 8B / response	-253.5	-25.4	-81.3	$\mathbf{+49.4}$
OPeRA				
user-profile LoRA 1.7B / S / action	-41.4	$+4.9$	$\mathbf{+17.7}$	$+0.9$
user-profile LoRA 1.7B / S / timing	$+1.5$	-0.2	$+2.9$	$+0.3$
user-profile LoRA 1.7B / D / action	-26.4	-5.1	$+6.3$	$+1.4$
user-profile LoRA 1.7B / D / timing	-1.4	-4.5	-1.3	$+1.1$
user-profile LoRA 1.7B / C / action	$+13.3$	$\mathbf{+9.0}$	$+6.0$	$+1.2$
user-profile LoRA 1.7B / C / timing	-4.5	-2.1	-0.2	$+0.2$
user-profile LoRA 8B / S / action	-48.5	-54.1	-13.0	$+1.9$
user-profile LoRA 8B / S / timing	-1.4	$+0.4$	$+0.6$	-0.2
user-profile LoRA 8B / D / action	-5.2	$\mathbf{+49.5}$	$+1.3$	$+1.7$
user-profile LoRA 8B / D / timing	-1.6	-4.6	-1.3	$+0.3$
user-profile LoRA 8B / C / action	-17.1	-2.8	-1.7	$+0.1$
user-profile LoRA 8B / C / timing	-5.0	-4.4	-1.8	-0.4
frozen prompt 1.7B / S / action	$\mathbf{+900.7}^{\ddagger}$	$+73.5$	$\mathbf{+109.2}$	-18.4
frozen prompt 1.7B / S / timing	$\mathbf{+159.3}^{\ddagger}$	-0.9	-0.7	$+4.4$
frozen prompt 1.7B / D / action	$\mathbf{+1986.3}^{\ddagger}$	$\mathbf{+51.4}$	$\mathbf{+56.6}$	-11.9
frozen prompt 1.7B / D / timing	-1.1	$\mathbf{+28.8}$	$+24.8$	$+5.0$
frozen prompt 1.7B / C / action	$\mathbf{+1267.5}^{\ddagger}$	$+38.0$	$+23.2$	-16.9
frozen prompt 1.7B / C / timing	-62.8	$+23.7$	$+19.8$	$+10.4$
frozen prompt 8B / S / action	$\mathbf{+590.0}^{\ddagger}$	-11.2	-57.0	-21.8
frozen prompt 8B / S / timing	-30.7	$\mathbf{+2.9}$	$+2.0$	-0.5
frozen prompt 8B / D / action	$\mathbf{+936.5}^{\ddagger}$	$+20.6$	$+10.7$	$+6.5$
frozen prompt 8B / D / timing	$\mathbf{+22.4}$	-3.0	-1.1	$+0.9$
frozen prompt 8B / C / action	$\mathbf{+732.2}^{\ddagger}$	-13.9	-46.9	-28.6
frozen prompt 8B / C / timing	-3.1	-8.7	-3.8	$+0.5$
frozen prompt 30B-A3B / S / action	$\mathbf{+634.3}^{\ddagger}$	$\mathbf{+70.8}$	$\mathbf{+71.2}$	-5.2
frozen prompt 30B-A3B / S / timing	$\mathbf{+17.5}$	-0.2	$+0.7$	-2.2
frozen prompt 30B-A3B / D / action	$\mathbf{+309.1}^{\ddagger}$	-9.8	-16.8	-47.2
frozen prompt 30B-A3B / D / timing	-100.1	$\mathbf{+47.6}$	$+26.8$	$+1.6$
frozen prompt 30B-A3B / C / action	$\mathbf{+584.8}^{\ddagger}$	$\mathbf{+54.3}$	$\mathbf{+42.5}$	-1.2
frozen prompt 30B-A3B / C / timing	-23.6	$\mathbf{+16.4}$	$+9.7$	$+1.4$

Table 8, continued

Method / channel	Δ_{total}	Δ_{pop}	Δ_{group}	Δ_{indiv}
Osim 4B / S / action	$+104.2$	$\mathbf{+36.4}^{\ddagger}$	$\mathbf{+19.0}$	$+2.8$
Osim 4B / S / timing	$\mathbf{+10.2}$	-1.2	-0.7	-1.5
Osim 4B / D / action	$\mathbf{+176.7}^{\ddagger}$	$\mathbf{+34.3}^{\ddagger}$	$\mathbf{+32.7}$	$+3.4$
Osim 4B / D / timing	$\mathbf{+10.8}$	-1.6	-1.0	$+1.3$
Osim 4B / C / action	$\mathbf{+162.3}$	$\mathbf{+63.5}^{\ddagger}$	$\mathbf{+45.8}^{\ddagger}$	$+12.7$
Osim 4B / C / timing	$+5.4$	-1.4	-0.8	-0.5
Osim 4B-mid / S / action	$\mathbf{+223.1}^{\ddagger}$	$\mathbf{+14.9}$	$+5.2$	-3.6
Osim 4B-mid / S / timing	-0.7	$+0.0$	$+0.6$	-1.4
Osim 4B-mid / D / action	$\mathbf{+288.6}^{\ddagger}$	$\mathbf{+16.0}$	$\mathbf{+15.2}$	$+1.3$
Osim 4B-mid / D / timing	-7.1	-1.4	-2.6	-0.1
Osim 4B-mid / C / action	$+122.3$	$\mathbf{+26.6}$	$+18.7$	$+4.2$
Osim 4B-mid / C / timing	-2.7	-0.0	$+0.6$	-0.5
Osim 8B / S / action	$\mathbf{+237.6}^{\ddagger}$	$\mathbf{+16.5}$	$+7.4$	-1.3
Osim 8B / S / timing	-19.5	-2.5	-1.6	-1.3
Osim 8B / D / action	$\mathbf{+265.1}^{\ddagger}$	-7.2	-3.8	-10.2
Osim 8B / D / timing	$\mathbf{+7.4}^{\ddagger}$	-2.8	-1.6	$+0.2$
Osim 8B / C / action	$\mathbf{+198.9}^{\ddagger}$	$\mathbf{+24.3}$	$+18.4$	$+8.3$
Osim 8B / C / timing	-3.3	-4.8	-3.2	$+0.7$
Osim 8B-mid / S / action	$\mathbf{+181.1}^{\ddagger}$	$+4.8$	$+0.7$	$+1.7$
Osim 8B-mid / S / timing	-5.1	-0.2	$+0.1$	$+0.8$
Osim 8B-mid / D / action	$+57.7$	-12.2	-13.5	-0.2
Osim 8B-mid / D / timing	$+0.5$	-2.5	-2.7	-0.7
Osim 8B-mid / C / action	$\mathbf{+156.8}^{\ddagger}$	$+8.0$	$+6.0$	-2.9
Osim 8B-mid / C / timing	-2.5	-2.9	-2.4	-0.8
CoSER 8B / S / action	-5.7	$\mathbf{+8.7}^{\ddagger}$	$\mathbf{+8.3}$	-2.7
CoSER 8B / S / timing	-4.7	-0.3	$+0.3$	$+0.1$
CoSER 8B / D / action	-62.3	$+1.6$	-0.2	-4.8
CoSER 8B / D / timing	-2.8	$\mathbf{+2.2}$	$\mathbf{+2.4}$	$+0.1$
CoSER 8B / C / action	-47.0	-7.0	-9.2	-12.0
CoSER 8B / C / timing	-7.6	$+1.3$	$+1.3$	$+0.9$
HumanLike 7B / S / action	$\mathbf{+742.9}^{\ddagger}$	$\mathbf{+214.4}^{\ddagger}$	$\mathbf{+231.1}^{\ddagger}$	-53.2
HumanLike 7B / S / timing	-97.3	$\mathbf{+64.4}^{\ddagger}$	$\mathbf{+38.0}^{\ddagger}$	$+5.4$
HumanLike 7B / D / action	$\mathbf{+540.0}^{\ddagger}$	$+22.2$	$+4.5$	-8.1
HumanLike 7B / D / timing	-246.5	-29.4	-17.2	$+5.7$
HumanLike 7B / C / action	$\mathbf{+1057.6}^{\ddagger}$	$\mathbf{+119.9}$	$+72.0$	-70.4
HumanLike 7B / C / timing	-168.4	$\mathbf{+57.5}^{\ddagger}$	$\mathbf{+36.6}^{\ddagger}$	$+8.2$
WildChat				
Qwen3 1.7B / next turn	$+0.135$	–	–	$+0.569$
Qwen3 8B / next turn	$+0.546$	–	–	$+0.651$
Qwen3 32B (dense) / next turn	$+0.621$	–	–	$+0.630$
HumanLike 7B / next turn	$+1.252$	–	–	$+0.956$
CoSER 8B / next turn	$+0.505$	–	–	$+0.632$
Osim 8B / next turn	$+0.447$	–	–	$+0.528$
HumanLM opinion / next turn	$+0.561$	–	–	$+0.524$

Olmo-3.1-32B outside the 84. Frozen-prompt Olmo-3.1-32B was scored on the three OPeRA timing configurations as a second model family outside the 84 combinations. No gain favors the target: supplying the profile raises surprisal on two of the three configurations, and on the static configuration the imposter's information scores above the target's own. Read the way Table 1 reads the other methods, profile-versus-none is adverse on two of three, next-decision accuracy favors the profile on none, and total variation favors it on all three and still on two of them under the imposter's information.

Two rows were scored under a prompt window that truncated part of their four-shot prefix: the chess frozen-prompt combinations at 1.7B and 8B, whose prompts run 748–781 tokens against a 640-token cap. Rescoring both at a cap that truncates nothing lowers the surprisal under every condition but

changes no verdict: all eight gains keep their sign and their interval status, so every count in Table 1 and Section 5.1 is unchanged, and the tables keep the original values.

C.2 TARGET-VERSUS-IMPOSTER INTERVALS

Table 9 gives the 95% interval of Δ_{indiv} for all 84 combinations.

Table 9: Target-versus-imposter gain Δ_{indiv} for all 84 combinations, with its paired per-user bootstrap 95% interval; positive favors the target. A double dagger (‡) marks significance after Holm correction over the 84 combinations; a single dagger (†) marks significance only under the correction within each of these four groups of combinations. Both apply in either direction.

family	channel	config	model	Δ_{indiv} [95% CI]
Chess and knowledge tracing (nats per decision) — 18 combinations				
recurrent embedding	move	chess	–	-0.0020 [-0.0063, +0.0025]
recurrent embedding	response	KT	–	+0.0159 [+0.0087, +0.0229]‡
recurrent embedding	timing	chess	–	-0.0002 [-0.0477, +0.0448]
recurrent embedding	timing	KT	–	+0.0018 [-0.0001, +0.0038]
frozen prompt	response	kt (1.7b)	–	+0.0559 [+0.0143, +0.1007]
frozen prompt	response	kt (8b)	–	+0.0494 [+0.0150, +0.0864]
frozen prompt	timing	chess (1.7b)	–	+0.0027 [-0.0036, +0.0086]
frozen prompt	timing	chess (8b)	–	+0.0114 [-0.0028, +0.0250]
user-profile LoRA	response	kt (1.7b)	–	+0.0136 [+0.0015, +0.0263]
user-profile LoRA	response	kt (8b)	–	+0.0137 [+0.0034, +0.0243]
user-profile LoRA	timing	chess (1.7b)	–	+0.0036 [+0.0005, +0.0065]
user-profile LoRA	timing	chess (8b)	–	+0.0065 [+0.0019, +0.0113]
static embedding	move	chess	–	+0.0183 [+0.0129, +0.0238]‡
static embedding	response	KT	–	+0.0112 [+0.0060, +0.0164]‡
static embedding	response	KT (joint)	–	+0.0113 [+0.0058, +0.0169]‡
static embedding	timing	chess	–	+0.0625 [+0.0522, +0.0736]‡
static embedding	timing	KT	–	+0.0728 [+0.0508, +0.0950]‡
structured memory	response	KT	–	-0.0527 [-0.0615, -0.0438]
OPeRA, user-profile LoRA and frozen prompt (nats per decision) — 30 combinations				
frozen prompt	action	combined	1.7b	-0.0169 [-0.0802, +0.0446]
frozen prompt	action	combined	8b	-0.0286 [-0.0886, +0.0416]
frozen prompt	action	dynamic	1.7b	-0.0119 [-0.0509, +0.0282]
frozen prompt	action	dynamic	8b	+0.0065 [-0.0441, +0.0633]
frozen prompt	action	static	1.7b	-0.0184 [-0.1036, +0.0674]
frozen prompt	action	static	8b	-0.0218 [-0.0782, +0.0431]
frozen prompt	timing	combined	1.7b	+0.0104 [-0.0500, +0.0702]
frozen prompt	timing	combined	8b	+0.0005 [-0.0127, +0.0140]
frozen prompt	timing	dynamic	1.7b	+0.0050 [-0.0286, +0.0395]
frozen prompt	timing	dynamic	8b	+0.0009 [-0.0143, +0.0167]
frozen prompt	timing	static	1.7b	+0.0044 [-0.0037, +0.0127]
frozen prompt	timing	static	8b	-0.0005 [-0.0036, +0.0024]
user-profile LoRA	action	combined	1.7b	+0.0012 [-0.0035, +0.0062]
user-profile LoRA	action	combined	8b	+0.0001 [-0.0045, +0.0047]
user-profile LoRA	action	dynamic	1.7b	+0.0014 [-0.0034, +0.0064]
user-profile LoRA	action	dynamic	8b	+0.0017 [-0.0047, +0.0087]
user-profile LoRA	action	static	1.7b	+0.0009 [-0.0037, +0.0052]
user-profile LoRA	action	static	8b	+0.0019 [-0.0032, +0.0074]
user-profile LoRA	timing	combined	1.7b	+0.0002 [-0.0023, +0.0027]
user-profile LoRA	timing	combined	8b	-0.0004 [-0.0017, +0.0009]
user-profile LoRA	timing	dynamic	1.7b	+0.0011 [-0.0011, +0.0037]
user-profile LoRA	timing	dynamic	8b	+0.0003 [-0.0013, +0.0021]
user-profile LoRA	timing	static	1.7b	+0.0003 [-0.0007, +0.0013]
user-profile LoRA	timing	static	8b	-0.0002 [-0.0007, +0.0004]
frozen prompt (30B)	action	combined	30b	-0.0012 [-0.0346, +0.0327]
frozen prompt (30B)	action	dynamic	30b	-0.0472 [-0.0920, -0.0065]
frozen prompt (30B)	action	static	30b	-0.0052 [-0.0449, +0.0394]
frozen prompt (30B)	timing	combined	30b	+0.0014 [-0.0164, +0.0182]
frozen prompt (30B)	timing	dynamic	30b	+0.0016 [-0.0323, +0.0327]
frozen prompt (30B)	timing	static	30b	-0.0022 [-0.0070, +0.0026]
OPeRA, CoSER and HumanLike (nats per decision) — 12 combinations				
released simulator (Sim2Real)	action	combined	coser-8b	-0.0120 [-0.0263, +0.0007]
released simulator (Sim2Real)	action	combined	humanlike-7b	-0.0704 [-0.1498, +0.0029]
released simulator (Sim2Real)	action	dynamic	coser-8b	-0.0048 [-0.0171, +0.0066]
released simulator (Sim2Real)	action	dynamic	humanlike-7b	-0.0081 [-0.0520, +0.0330]
released simulator (Sim2Real)	action	static	coser-8b	-0.0027 [-0.0064, +0.0011]
released simulator (Sim2Real)	action	static	humanlike-7b	-0.0532 [-0.1308, +0.0189]
released simulator (Sim2Real)	timing	combined	coser-8b	+0.0009 [-0.0008, +0.0025]
released simulator (Sim2Real)	timing	combined	humanlike-7b	+0.0082 [-0.0090, +0.0266]
released simulator (Sim2Real)	timing	dynamic	coser-8b	+0.0001 [-0.0011, +0.0012]
released simulator (Sim2Real)	timing	dynamic	humanlike-7b	+0.0057 [-0.0168, +0.0295]

Table 9 (continued)

family	channel	config	model	Δ_{indiv} [95% CI]
released simulator (Sim2Real)	timing	static	coser-8b	+0.0001 [-0.0010, +0.0011]
released simulator (Sim2Real)	timing	static	humanlike-7b	+0.0054 [-0.0078, +0.0181]
OPeRA, Osim (nats per decision) — 24 combinations				
released simulator (Osim)	action	combined	osim-4b	+0.0127 [-0.0091, +0.0336]
released simulator (Osim)	action	combined	osim-4b-mid	+0.0042 [-0.0220, +0.0285]
released simulator (Osim)	action	combined	osim-8b	+0.0083 [-0.0086, +0.0254]
released simulator (Osim)	action	combined	osim-8b-mid	-0.0029 [-0.0180, +0.0111]
released simulator (Osim)	action	dynamic	osim-4b	+0.0034 [-0.0136, +0.0188]
released simulator (Osim)	action	dynamic	osim-4b-mid	+0.0013 [-0.0114, +0.0134]
released simulator (Osim)	action	dynamic	osim-8b	-0.0102 [-0.0259, +0.0056]
released simulator (Osim)	action	dynamic	osim-8b-mid	-0.0002 [-0.0153, +0.0139]
released simulator (Osim)	action	static	osim-4b	+0.0028 [-0.0161, +0.0227]
released simulator (Osim)	action	static	osim-4b-mid	-0.0036 [-0.0219, +0.0166]
released simulator (Osim)	action	static	osim-8b	-0.0013 [-0.0159, +0.0134]
released simulator (Osim)	action	static	osim-8b-mid	+0.0017 [-0.0143, +0.0196]
released simulator (Osim)	timing	combined	osim-4b	-0.0005 [-0.0042, +0.0030]
released simulator (Osim)	timing	combined	osim-4b-mid	-0.0005 [-0.0032, +0.0024]
released simulator (Osim)	timing	combined	osim-8b	+0.0007 [-0.0051, +0.0064]
released simulator (Osim)	timing	combined	osim-8b-mid	-0.0008 [-0.0035, +0.0019]
released simulator (Osim)	timing	dynamic	osim-4b	+0.0013 [-0.0018, +0.0045]
released simulator (Osim)	timing	dynamic	osim-4b-mid	-0.0001 [-0.0027, +0.0024]
released simulator (Osim)	timing	dynamic	osim-8b	+0.0002 [-0.0038, +0.0039]
released simulator (Osim)	timing	dynamic	osim-8b-mid	-0.0007 [-0.0035, +0.0021]
released simulator (Osim)	timing	static	osim-4b	-0.0015 [-0.0035, +0.0004]
released simulator (Osim)	timing	static	osim-4b-mid	-0.0014 [-0.0034, +0.0004]
released simulator (Osim)	timing	static	osim-8b	-0.0013 [-0.0032, +0.0005]
released simulator (Osim)	timing	static	osim-8b-mid	+0.0008 [-0.0006, +0.0022]

C.3 Per-user marginal baseline

The train-only per-user baseline fits a log-normal timing distribution with each user's own μ and σ, or smoothed label frequencies for discrete channels. Chess move frequencies are restricted and renormalized over the current legal moves. Each baseline scores the target user's held-out behavior using either the target's statistics or a matched imposter's statistics. Table 10 gives the method Δ_{indiv}, the baseline Δ_{indiv}, the residual, and their ratio. A negative residual means the baseline matches or exceeds the method's target-versus-imposter separation; the ratio has no stable reading where the method's Δ_{indiv} interval covers zero, marked n.d.

Every timing combination is absorbed: per-user speed alone exceeds each method's separation by a wide margin, so the observed timing separation reflects stable per-user speed rather than conditional timing structure. The discrete channels split. Both chess move combinations and all static-embedding response combinations are absorbed, the two user-profile LoRA response combinations are matched to within 0.0002 nats, and three knowledge-tracing response combinations keep a positive residual: recurrent embedding and frozen prompt 1.7B and 8B. Table 11 closes that gap: under a 1PL item-response baseline and a profile-field logistic reader, no knowledge-tracing component resolvably exceeds a train-only baseline, and on the two static response combinations the baseline wins outright. Stable per-user marginals are themselves individual information; when a train-only marginal matches a simulator, the information structure is simple rather than absent, and the stronger baselines test whether the simulator adds anything beyond it.

D Results: existing benchmark comparisons

Eighteen further rows score API models (GPT-4.1, GPT-4.1-mini, DeepSeek-V3.2) through a letter-choice protocol whose surprisal floor is not comparable to local label-token surprisal; they are diagnostics outside the 84 combinations and appear only in this appendix.

Table 10: Marginal decomposition of all 17 complete five-condition combinations, point estimates in nats with profiles fixed at the training boundary. 'marginal-only' is the target-versus-imposter gain of a per-user marginal model fitted on the training split alone; 'residual' is the method's Δ_{indiv} minus it, and negative means the marginal model matches or exceeds the method. *Capture* is marginal-only over method: 1.00 means the marginal reproduces the whole of Δ_{indiv}, and above 1.00 it beats the method. Combinations whose Δ_{indiv} interval covers zero have no stable ratio and are marked n.d.; positive Δ_{indiv} favors the target.

combination	Δ_{indiv}	marginal-only	residual	capture
user-profile LoRA 1.7B · chess · timing	$+0.0036$	$+0.2061$	-0.2025	57.84
user-profile LoRA 8B · chess · timing	$+0.0065$	$+0.2061$	-0.1996	31.79
static embedding · chess · move	$+0.0183$	$+0.1195$	-0.1012	6.54
static embedding · chess · timing	$+0.0625$	$+0.2055$	-0.1430	3.29
static embedding · KT · timing	$+0.0728$	$+0.1200$	-0.0472	1.65
static embedding · KT · response	$+0.0112$	$+0.0133$	-0.0021	1.19
static embedding · KT (joint) · response	$+0.0113$	$+0.0133$	-0.0019	1.17
user-profile LoRA 1.7B · KT · response	$+0.0136$	$+0.0138$	-0.0002	1.02
user-profile LoRA 8B · KT · response	$+0.0137$	$+0.0138$	-0.0001	1.01
recurrent embedding · KT · response	$+0.0159$	$+0.0133$	$+0.0027$	0.83
frozen prompt 8B · KT · response	$+0.0494$	$+0.0119$	$+0.0375$	0.24
frozen prompt 1.7B · KT · response	$+0.0559$	$+0.0119$	$+0.0440$	0.21
recurrent embedding · chess · move	-0.0020	$+0.1195$	-0.1214	n.d.
recurrent embedding · chess · timing	-0.0002	$+0.2055$	-0.2057	n.d.
recurrent embedding · KT · timing	$+0.0018$	$+0.1200$	-0.1182	n.d.
frozen prompt 1.7B · chess · timing	$+0.0027$	$+0.2050$	-0.2023	n.d.
frozen prompt 8B · chess · timing	$+0.0114$	$+0.2050$	-0.1936	n.d.

Table 11: Stronger train-only baselines on the KT response combinations. IRT is a 1PL per-student ability with per-item difficulty; 'fields' is a population logistic over the profile's own ten numeric fields, frozen at each user's boundary. Both are recency-clean. Capture is baseline over method (ratio of means); the residual is per-user method minus the fields baseline. No residual interval excludes zero favorably.

combination	method Δ_{indiv} [CI]	IRT	capt.	fields	capt.	residual [CI]
static · KT · resp.	$+0.0112$ [$+0.0057$,$+0.0166$]	$+0.0238$	2.14	$+0.0234$	2.10	-0.0122 [-0.0217,-0.0029]
static · KT · resp.	$+0.0113$ [$+0.0057$,$+0.0169$]	$+0.0238$	2.10	$+0.0234$	2.06	-0.0121 [-0.0218,-0.0026]
recurrent embedding · KT · resp.	$+0.0159$ [$+0.0086$,$+0.0231$]	$+0.0238$	1.49	$+0.0234$	1.47	-0.0075 [-0.0164,$+0.0017$]
user-profile LoRA 1.7B · KT · resp.	$+0.0136$ [$+0.0008$,$+0.0263$]	$+0.0241$	1.77	$+0.0226$	1.67	-0.0090 [-0.0219,$+0.0039$]
user-profile LoRA 8B · KT · resp.	$+0.0137$ [$+0.0028$,$+0.0245$]	$+0.0241$	1.76	$+0.0226$	1.65	-0.0090 [-0.0210,$+0.0029$]
frozen prompt 1.7B · KT · resp.	$+0.0559$ [$+0.0126$,$+0.1002$]	$+0.0227$	0.41	$+0.0238$	0.43	$+0.0321$ [-0.0103,$+0.0754$]
frozen prompt 8B · KT · resp.	$+0.0494$ [$+0.0141$,$+0.0856$]	$+0.0227$	0.46	$+0.0238$	0.48	$+0.0256$ [-0.0084,$+0.0603$]

E Results: selecting simulators

E.1 Simulator-selection evidence and preregistered protocol

The seven-simulator attribution comparison uses 1,444 out-of-distribution turns from 240 users, fixed profiles, and a uniform rule for selecting the context prefix under each simulator's native interface. Table 13 gives the complete within-model gains.

E.1.1 Preregistered content-based selection test

The preregistration fixes the following design before the first sample is drawn.[4] The test uses the same 236 users, 1,427 OOD turns, seven simulators, fixed profiles, imposter assignment, and prefix-selection rule. It samples only the target-profile and matched-imposter conditions, with $K = 8$, temperature 1, 96 new tokens, and seed 0, and saves every generated text.

[4]Artifact preregistration: `docs/project/PLAN-P2E-WILDCHAT-CONTENT-PROJECTION.md`, Sections 1–4.

Table 12: Comparison with the eleven closest public benchmarks. ✓ means present, $\sim$ means partial, and ✗ means absent in the documented design. In the imposter column $\sim$ marks a design that varies the person without a matched alternative on the conditioning input: BehaviorBench substitutes an unrelated profile in an ablation, and TwinVoice draws its multiple-choice distractors from other users' replies.

Benchmark	Real indiv.	Seq. domains	Trained conditions	Retrain /cond.	Imposter	Static/ recurrent embedding	Timing
Ours	✓	✓	✓	✓	✓	✓	✓
BehaviorBench (Yang et al., 2026)	$\sim$	✓	✗	✗	$\sim$	✗	✗
SimBench (Hu et al., 2026)	✗	✗	$\sim$	✗	✗	✗	✗
PersonaMem (Jiang et al., 2025)	✗	✗	✗	✗	✗	$\sim$	✗
HorizonBench (Li et al., 2026)	✗	✗	✗	✗	✗	$\sim$	✗
Sim2Real gap (Zhou et al., 2026b)	$\sim$	✗	$\sim$	✗	✗	✗	✗
Osim (Zhou et al., 2026a)	$\sim$	✗	✓	✗	✗	✗	✗
Distributional gap (Mehri et al., 2026)	$\sim$	✗	$\sim$	✗	✗	✗	$\sim$
RealUserSim (Zhu et al., 2026)	✓	$\sim$	✗	✗	✗	✗	✗
CustomerSim (Pruksachatkun et al., 2026)	✗	$\sim$	$\sim$	✗	✗	✗	✗
TwinVoice (Du et al., 2026)	✓	✗	✗	✗	$\sim$	✗	✗
SimEval-IR (Zerhoudi, 2026)	✗	$\sim$	✗	✗	✗	✗	✗

Table 13: Complete 95% paired-user bootstrap intervals for the seven-simulator WildChat attribution comparison (10,000 resamples, seed 0). Both gains use fixed profiles and within-model surprisal differences. These are descriptive intervals without the original audit's pooled multiplicity-corrected test.

Simulator	Δ_{indiv} [95% CI]	Δ_{total} [95% CI]
HumanLike-7B	$+0.956[+0.810, +1.112]$	$+1.252[+1.097, +1.418]$
Qwen3-8B	$+0.651[+0.551, +0.757]$	$+0.546[+0.448, +0.649]$
CoSER-8B	$+0.632[+0.536, +0.734]$	$+0.505[+0.409, +0.604]$
Qwen3-32B (dense)	$+0.630[+0.539, +0.727]$	$+0.621[+0.533, +0.715]$
Qwen3-1.7B	$+0.569[+0.469, +0.675]$	$+0.135[+0.026, +0.247]$
Osim-8B	$+0.528[+0.451, +0.609]$	$+0.447[+0.376, +0.521]$
HumanLM-opinion	$+0.524[+0.372, +0.681]$	$+0.561[+0.423, +0.701]$

Metric selection on real data. Two external metrics are screened before generation. The first is cosine similarity from `all-MiniLM-L6-v2`, with mean pooling, L2 normalization, and 256-token truncation. The second is character n-gram TF–IDF cosine, using within-word 3–5-grams, sublinear term frequency, and L2 normalization; inverse document frequencies are fitted once to the 236 profile texts and 1,427 real turns. For each real turn, the embedding screen uses the maximum similarity over the profile's nonempty message bullets, while TF–IDF uses the full profile. Each metric's screen averages target-minus-imposter similarity within user and bootstraps users 10,000 times at seed 0. A metric passes if its 95% interval is above zero. MiniLM has a target-minus-imposter similarity gain of $+0.230$ $[+0.209, +0.252]$, compared with $+0.089$ $[+0.076, +0.102]$ for TF–IDF (95% user-bootstrap intervals). MiniLM is primary because its screen has $z = 21.1$, compared with $z = 13.5$ for TF–IDF. Both metrics were computed on the generated samples.

Content gain and target win rate. Let $x_{u,t}$ be the real next turn, $\tilde{x}_{I,u,t,k}$ the kth sample under information condition I, and d the selected distance, one minus cosine similarity. The primary per-turn distance and per-user content gain are

$$D_I(u,t) = \frac{1}{K}\sum_{k=1}^{K} d(\tilde{x}_{I,u,t,k}, x_{u,t}), \qquad g_u = \frac{1}{n_u}\sum_{t}\big[D_{I_{\text{imposter}}}(u,t) - D_{I_{\text{target}}}(u,t)\big]. \tag{3}$$

Positive g_u means that the target profile reduces content distance. Table 2 reports its mean across users. We call this avg@8. The minimum sample distance is best@8; it uses the real turn to choose the closest sample and therefore measures what the simulator can produce, not a deployable selection rule. Target win rate is the two-alternative forced-choice score expressed as a percentage: a turn receives 1 when $D_{I_{\text{target}}} < D_{I_{\text{imposter}}}$, 0 when the imposter distance is smaller, and 0.5 for a tie. Scores

Table 14: **Uncertainty in the sampled-content gains.** Paired-user bootstrap 95% confidence intervals (10,000 resamples) for the content gains in Table 2. Positive values favor the target profile.

Simulator	avg@8 content gain	best@8 content gain
HumanLike-7B	$[+0.099, +0.135]$	$[+0.114, +0.157]$
Qwen3-1.7B	$[+0.089, +0.122]$	$[+0.097, +0.133]$
Qwen3-8B	$[+0.089, +0.118]$	$[+0.115, +0.151]$
Qwen3-32B (dense)	$[+0.087, +0.117]$	$[+0.113, +0.148]$
Osim-8B	$[+0.057, +0.078]$	$[+0.092, +0.123]$
HumanLM-opinion	$[+0.052, +0.079]$	$[+0.072, +0.103]$
CoSER-8B	$[+0.036, +0.056]$	$[+0.071, +0.102]$

are averaged within each user and then across users; 50% is the chance reference when profile identity has no effect.

Readouts and decision rule. The preregistered readouts are:

1. R1 reports each simulator's mean content gain and target win rate with 95% user-bootstrap intervals (10,000 resamples).

2. R2 compares the difference in AT gain Δ_{indiv} with the difference in content gain for each of the 21 simulator pairs. A pair is resolved when the paired-user bootstrap interval for its content-gain difference excludes zero. With at least 10 resolved pairs, the amended registered line is 70% for support and 50% for refutation; intermediate agreement is inconclusive. Fewer than 10 resolved pairs is underpowered. Point Kendall ordering agreement and the two-direction user-fold cross-fit version are also reported.

3. R3 compares the ordering by lower absolute target-profile mean distance with that by higher AT gain Δ_{indiv} using the same pairwise rule. This distance shares one metric space across simulators and is reported regardless of R2.

4. R4 correlates per-user AT gains Δ_{indiv} from odd turns with content gains from even turns, and then swaps the halves, using Spearman correlation with user-bootstrap intervals.

5. R5 recomputes form-label readouts (length, question status, code presence) on these same saved samples as robustness columns beside R1.

The completed readout gives outcome A. MiniLM is the primary metric, and Table 2 reports the R1 point estimates. All fourteen content-gain intervals in Table 14 lie above zero, and all target win rates exceed 50%. R2 agrees on 11/15 avg@8 pairs and 13/16 best@8 pairs; Kendall's τ is $+0.43$, with the same point estimate in both cross-fit directions for avg@8; the 95% intervals are $[-0.05, +0.62]$ for fold A to B and $[+0.14, +0.62]$ for fold B to A. R3 agrees on 10/18 avg@8 pairs and 8/17 best@8 pairs, so AT does not order absolute sampled quality. R4 is positive for six of seven simulators, and R5 on the same samples gives form-label 2AFC of 0.44–0.51. The four primary disagreements all involve CoSER; without CoSER, the post hoc exploratory agreements are 9/9 and 11/11. These results support the scoped claim that AT gains correlate with the target-profile effect on sampled content, while leaving absolute quality separate.

A decoding-temperature check re-sampled Qwen3-8B, HumanLike-7B, and Osim-8B at temperature 0.7 on the identical turns and profiles (Table 15). The three-way ordering is unchanged at both endpoints; the paired change in content gain covers zero for Qwen3-8B and HumanLike-7B, and Osim-8B gains slightly, so lowering the temperature does not remove the profile effect.

The LLM-judge selector of Section 5.3 uses gpt-5.6-sol on one sample per turn and condition; the judge sees the last two turns, the real next turn, and one candidate, never the profile. Table 16 gives its score under each profile: the judge registers the substitution on every simulator, most strongly on HumanLike-7B, and least on CoSER-8B and HumanLM-opinion, whose intervals include zero.

Table 15: **Decoding temperature does not change the selection result.** Content gain $\bar{g}$ of the three re-sampled simulators at temperature 1 (the preregistered run) and 0.7, same 1,427 turns, profiles, and eight samples per condition; the change is the per-user paired difference with its 95% bootstrap interval. The ordering HumanLike-7B $>$ Qwen3-8B $>$ Osim-8B holds at both temperatures and both endpoints.

Simulator	Endpoint	$\bar{g}$ at $T{=}1$	$\bar{g}$ at $T{=}0.7$	Change [95% CI]
Qwen3-8B	avg@8	$+0.103$	$+0.102$	-0.001 [-0.006, $+0.004$]
Qwen3-8B	best@8	$+0.133$	$+0.131$	-0.002 [-0.009, $+0.005$]
HumanLike-7B	avg@8	$+0.117$	$+0.117$	-0.000 [-0.005, $+0.004$]
HumanLike-7B	best@8	$+0.135$	$+0.131$	-0.004 [-0.013, $+0.004$]
Osim-8B	avg@8	$+0.067$	$+0.077$	$+0.010$ [$+0.003$, $+0.016$]
Osim-8B	best@8	$+0.107$	$+0.121$	$+0.014$ [$+0.003$, $+0.024$]

Table 16: **An LLM judge does register the matched imposter's profile.** Mean judge score of the generated turn against the real one (gpt-5.6-sol, blind to which profile was used; one sample per turn, 1,427 turns of 236 users) under the target's and under the matched imposter's profile, with the paired-user 95% interval of the difference and the share of the target score kept. Fidelity is the mean of five binary matches (persona, style, competency, interaction flow, pacing); alignment is the share of the real turn's key points covered.

	Judged fidelity			Judged alignment		
Simulator	target	imposter	kept	target	imposter	kept
Qwen3-1.7B	0.256	0.146	57%	0.093	0.062	67%
Qwen3-8B	0.206	0.123	60%	0.099	0.065	66%
Qwen3-32B (dense)	0.187	0.096	51%	0.088	0.053	60%
HumanLike-7B	0.105	0.036	34%	0.047	0.018	39%
CoSER-8B	0.076	0.066	87%	0.033	0.028	84%
Osim-8B	0.193	0.134	69%	0.074	0.061	83%
HumanLM-opinion	0.079	0.068	86%	0.047	0.041	87%

Table 17: Synthetic-population controls after the predeclared no-history screen. Accuracy gain is target-profile accuracy minus matched-imposter accuracy. Intervals use 10,000 persona-clustered bootstrap samples. Chance is 0.250 for PersonaMem and 0.200 for HorizonBench.

population	I_{imposter} [95% CI]	I_{target} [95% CI]	accuracy gain [95% CI]
PersonaMem (220 items, 20 personas)	0.251 [0.189, 0.314]	0.526 [0.441, 0.612]	$+0.274$ [$+0.177$, $+0.378$]
HorizonBench (109 items, 109 personas)	0.055 [0.018, 0.101]	0.119 [0.064, 0.183]	$+0.064$ [$+0.009$, $+0.119$]

F Discussion: validation and ablations

F.1 Known-signal controls

F.2 Stress nulls

The known-signal checks of Section 6 in numbers: after a history-leakage screen, PersonaMem's target history reaches 0.526 while its matched imposter stays at chance, and HorizonBench gives a paired gain of $+0.064$ with both conditions below chance; these controls are model-relative. On the synthetic population with rate $(1-\lambda)\bar{q}+\lambda q_u$, the per-user marginal produces 0 false positives in 200 null replicates at $\lambda = 0$ and a monotone effect reaching $+0.0308$ at $\lambda = 1$ (exact binomial 95% interval for the false-positive rate: $[0, 1.8\%]$).

Section 6's three stress populations, in full. In each the generative truth is known: zero identity signal under drift, zero within-group individuality under group structure, and full stable individuality under the matcher-resolution sweep. Table 18 gives the drift sweep; the group and matcher results appear in its caption.

Table 18: Stress nulls, 200 replicates each, rate of significant intervals against a nominal 5%. Left: a shared drifting rate with identity signal exactly zero—the updated-profile gain fires once drift is moderate, the fixed-profile Δ_{indiv} never does. Right: under pure group structure (zero within-group individuality) the tertile rule fires at 0.890 and the accuracy nearest-neighbour at 0.005; on a fully individual, fully stable population the gain falls from +0.0551 (2-cell) to +0.0299 (tertile) to +0.0000 (NN).

	updated profile		Δ_{indiv}	
drift	FP rate	mean	FP rate	mean
0.00	0.000	-0.0049	0.010	+0.0000
0.15	0.985	+0.0067	0.000	+0.0001
0.30	1.000	+0.0429	0.000	+0.0000
0.45	1.000	+0.1087	0.000	+0.0000

F.3 Closer imposters

Table 19 re-scores the combinations significant against the imposter with the imposter chosen as the nearest other user on five training-period features (activity, timing mean and spread, and for chess the entropy and top share of the move distribution), and with the hardest available imposter as an adversarial bound.

Table 19: The matcher hierarchy: the registered tertile derangement, the training-only nearest-neighbour matcher, and the hardest donor in the recorded set (an adversarial bound, not an estimand). 'k/m' counts matchers under which the gain remains significant, of those reported; the two prompting combinations have no hardest-donor bound recorded and carry their Section 5.1 bound.

combination	cert.	registered [CI]	nearest-neighbor [CI]	hardest [CI]	k/m
static · chess · move	✓	$+0.0183$ [$+0.0129$,$+0.0238$]	$+0.0182$ [$+0.0121$,$+0.0244$]	-0.0438 [-0.0484,-0.0391]	2/3
static · chess · timing	✓	$+0.0625$ [$+0.0522$,$+0.0736$]	$+0.0055$ [$+0.0027$,$+0.0083$]	-0.0194 [-0.0269,-0.0137]	2/3
static · KT · resp.	✓	$+0.0112$ [$+0.0060$,$+0.0164$]	$+0.0015$ [-0.0017,$+0.0047$]	-0.0442 [-0.0502,-0.0387]	1/3
static · KT · resp.	✓	$+0.0113$ [$+0.0058$,$+0.0169$]	$+0.0005$ [-0.0030,$+0.0038$]	-0.0475 [-0.0532,-0.0422]	1/3
static · KT · timing	✓	$+0.0728$ [$+0.0508$,$+0.0950$]	$+0.0842$ [$+0.0589$,$+0.1098$]	-0.1283 [-0.1468,-0.1111]	2/3
recurrent embedding · KT · resp.	✓	$+0.0159$ [$+0.0086$,$+0.0231$]	$+0.0064$ [-0.0009,$+0.0138$]	-0.0707 [-0.0772,-0.0646]	1/3
recurrent embedding · KT · timing		$+0.0018$ [-0.0001,$+0.0038$]	$+0.0007$ [-0.0010,$+0.0025$]	-0.0339 [-0.0366,-0.0314]	0/3
user-profile LoRA 8B · KT · resp.		$+0.0137$ [$+0.0028$,$+0.0245$]	$+0.0096$ [-0.0020,$+0.0212$]	—	1/2
frozen prompt 8B · KT · resp.		$+0.0494$ [$+0.0141$,$+0.0856$]	$+0.0242$ [-0.0079,$+0.0561$]	—	1/2

F.4 Updating the target profile with held-out rows inflates the identity gain

A protocol that is easy to adopt by accident scores the target condition with a profile that keeps absorbing the user's real held-out rows up to the decision being scored, while the imposter's profile stays fixed at its training boundary. Every family with a per-user state can do this: the recurrent embedding updates its latent per event, structured memory refits on the user's own rows, and a rendered profile summarizes history so far. A deployed simulator never observes the real held-out rows, so this gain measures recency as well as identity. Under it, 16 of the 84 combinations pass the pooled Holm correction instead of 6 (Table 20), and recurrent embedding on chess timing moves from -0.0002 [-0.0477, $+0.0448$] to $+0.2199$. Table 21 gives the per-combination gap, $\mathrm{drift} = \mathbb{S}(I_{\text{target}}) - \mathbb{S}(I_{\text{target}}^{\mathrm{updated}})$; nothing is re-run, these are recorded conditions paired differently, and the full recount repeats both Holm families on each gain for all 84 combinations. The synthetic drift sweep in Table 18 shows the same inflation with the identity signal set to zero. We therefore report only the fixed-profile gain in the paper. Static embedding conditions on a trained per-user embedding rather than a rendered profile, so it has no updated variant and does not appear here.

Table 20: Significance counts when the target profile is updated with held-out rows ('updated') versus fixed at the training boundary (Δ_{indiv}, the gain used in the paper). 'Significant' counts combinations whose interval lies above zero; the final column counts pooled-Holm passes.

combinations	number	significant updated / Δ_{indiv}	pooled Holm pass updated / Δ_{indiv}
chess and knowledge tracing	18	$17 \to 12$	$15 \to 6$
OPeRA, LoRA and frozen prompt	30	$4 \to 0$	$0 \to 0$
OPeRA, CoSER and HumanLike	12	$3 \to 0$	$1 \to 0$
OPeRA, Osim	24	$2 \to 0$	$0 \to 0$
all	84	$26 \to 12$	$16 \to 6$

Table 21: Updated-profile gain, drift, and the fixed-profile Δ_{indiv} per combination. Positive favors the target; 'share' is Δ_{indiv} over the updated-profile gain, blank where a ratio has no stable reading; ‡ marks a pooled-Holm pass on Δ_{indiv} (the five static-embedding combinations, with no updated variant, carry the other five passes). 'Status' is 'yes' when Δ_{indiv} is significant.

combination	updated	drift	Δ_{indiv} [95% CI]	share	status
user-profile LoRA 1.7B · KT · resp.	$+0.0279$	$+0.0143$	$+0.0136$ [$+0.0015$, $+0.0263$]	0.49	yes
user-profile LoRA 8B · KT · resp.	$+0.0260$	$+0.0123$	$+0.0137$ [$+0.0034$, $+0.0243$]	0.53	yes
user-profile LoRA 1.7B · chess · timing	$+0.0126$	$+0.0090$	$+0.0036$ [$+0.0005$, $+0.0065$]	0.28	yes
user-profile LoRA 8B · chess · timing	$+0.0219$	$+0.0154$	$+0.0065$ [$+0.0019$, $+0.0113$]	0.30	yes
frozen prompt 1.7B · KT · resp.	$+0.0743$	$+0.0184$	$+0.0559$ [$+0.0143$, $+0.1007$]	0.75	yes
frozen prompt 8B · KT · resp.	$+0.0910$	$+0.0416$	$+0.0494$ [$+0.0150$, $+0.0864$]	0.54	yes
frozen prompt 1.7B · chess · timing	$+0.0088$	$+0.0061$	$+0.0027$ [-0.0036, $+0.0086$]	0.31	no
frozen prompt 8B · chess · timing	$+0.0274$	$+0.0160$	$+0.0114$ [-0.0028, $+0.0250$]	0.42	no
structured memory · KT · resp.	$+0.0101$	$+0.0629$	-0.0527 [-0.0615, -0.0438]		no
recurrent embedding · KT · resp.	$+0.0418$	$+0.0259$	$+0.0159$ [$+0.0087$, $+0.0229$]‡	0.38	yes
recurrent embedding · KT · timing	$+0.0031$	$+0.0012$	$+0.0018$ [-0.0001, $+0.0038$]	0.60	no
recurrent embedding · chess · move	$+0.0010$	$+0.0029$	-0.0020 [-0.0063, $+0.0025$]		no
recurrent embedding · chess · timing	$+0.2199$	$+0.2201$	-0.0002 [-0.0477, $+0.0448$]	0.00	no

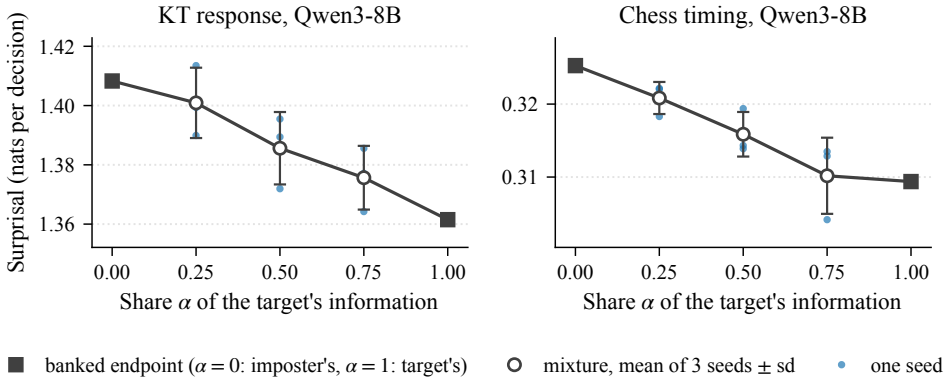

Figure 3: Surprisal against the share of the target's fixed information mixed into the imposter's, frozen-prompt Qwen3-8B. Squares are the original endpoints; circles are the mean over three seeds with the across-seed standard deviation as the bar and the seeds as dots.

F.5 Target-information dose

Figure 3 mixes the target's fixed information into the matched imposter's field by field, in nominal shares $\alpha \in \{0.25, 0.5, 0.75\}$ with three seeds each, for frozen-prompt Qwen3-8B on knowledge-tracing response and chess timing. The endpoints are the original I_{imposter} and I_{target} rows, which the $\alpha = 0$ and $\alpha = 1$ mixtures reproduce exactly; the realized shares are 0.24–0.75. Surprisal falls monotonically in α on both substrates, from 1.408 to 1.362 nats on knowledge tracing (-3.3%) and from 0.325 to 0.309 on chess (-4.9%), against an across-seed standard deviation of at most 0.012 and 0.005. The identity gain therefore responds to how much of the target's information the simulator is given, not only to whether it is given.