BEYOND COOPERATIVE SIMULATORS: GENERATING REALISTIC USER PERSONAS FOR ROBUST EVALUATION OF LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Model (LLM) agents are increasingly deployed in settings where they interact with diverse users, including those who are unclear, impatient, or reluctant to share information. However, collecting real interaction data at scale remains expensive. The field has turned to LLM-based *user simulators* as stand-ins, but these simulators inherit the behavior of their underlying models: cooperative and homogeneous. As a result, agents that appear strong in simulation often fail in real human interactions. To narrow this gap, we introduce **Persona Policies** (`PPol`), a plug-and-play control layer that induces realistic behavioral variation in user simulators while preserving original task goals. Rather than hand-crafting personas, we employ an evolutionary coding agent to discover *persona generation programs* optimized for human-likeness and behavioral coverage over real user conversations. The evolved program generates diverse, human-like personas for any task in the domain. Across 4 benchmarks—including τ^2-bench Retail and Airline, ColBench, and WildChat—evolved `PPol` yield 28-72% absolute gains in fitness score over the baseline simulator. In blinded evaluations, annotators judged `PPol` users as '*human*' 80.4% of the time, nearly $2\times$ more than the baseline simulators. Training agents with `PPol` also improves real-world performance: our user study with live human–agent interactions showed that fine-tuning with our method boosted task success by $+23\%$ over default baselines. `PPol` thus offers a novel approach to strengthen simulator-based evaluation and training without changing underlying tasks. [1]

1 INTRODUCTION

The promise of LLM-based assistants lies in their ability to act as autonomous agents, handling complex multi-turn tasks. However, to be truly useful, an agent must do more than navigate a software interface; it must navigate the unpredictable nature of human communication. Real users rarely provide information perfectly. They have distinct preferences, varying patience, and often express their needs through ambiguous, fragmented, or sometimes adversarial language (Keyvan & Huang, 2022). To be trustworthy, agents must adequately infer intent, adapt to individual style, and ensure no irrevocable actions are taken without clear alignment.

Building agents with this level of social and procedural robustness requires evaluation environments that replicate the friction of real-world service encounters. The field has therefore turned to simulation-based benchmarks, which pair agents with "user simulators": LLMs prompted to act as users or customers with fixed goals across domains like retail, banking, and travel booking (Barres et al., 2025). However, recent studies show that these default user simulators are overly cooperative, perfectly consistent, and highly forthcoming with information (Dou et al., 2025). This creates a *behavioral gap*: agents can appear strong against cooperative simulators but fail when confronted with the frustration, skepticism, or brevity of an actual human user (Zhou et al., 2026).

Steering simulators with traits like impatience or skepticism (He et al., 2026; Shim et al., 2026) create a valuable stress test that lowers agent success rates. However, while "richer synthetic users" impact performance, they are typically hand-specified or sampled from fixed, manual designs rather than being grounded in the actual, complex distribution of real human behaviors.

[1] Our code is available here: https://anonymous.4open.science/r/persona-policies-PPOL.

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.

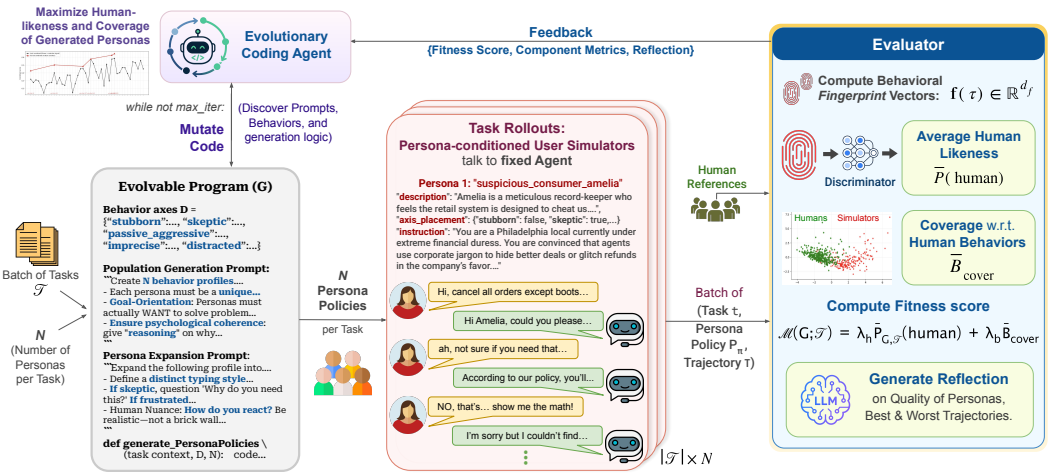


Figure 1: Overview of the **Persona Policies (PPol)** program-evolution loop. Given a task, program G generates N persona policies that vary in how simulated users communicate. Candidate programs are scored for human-likeness and behavioral coverage via task rollouts. Feedback on the quality of trajectories is used to guide refinement of the axes, sampling rules, and prompts. Once optimized, program G can produce diverse *persona policies* for new tasks in that domain.

To address this, we introduce **Persona Policies** (PPol), a plug-and-play control layer that diversifies user simulation by generating multiple human-like personas while keeping benchmark tasks fixed. Each '*policy*' is a short set of *additional* instructions added to the simulator prompt that controls *how* the user communicates, for example, by varying tone, pacing, selective disclosure, etc. This lets the same task be tested under many realistic interaction patterns, so benchmark performance better reflects robustness to real users rather than success on one cooperative simulator style.

Learning these policies is framed as *evolutionary program search*. Rather than handcrafting prompts, we implement a Python persona generator that creates a population of behavioral profiles and expands them into task-preserving roleplay instructions. Using OpenEvolve (Sharma, 2025), a frontier LLM iteratively mutates the generator's code. We evaluate candidate generators through agent–user rollouts scored on two objectives: **human-likeness**, measured as the probability of being classified as human by a trained discriminator, and **behavioral coverage**, which measures how well the N personas span the human distribution. Both metrics rely on *behavioral fingerprints*, 19 lexical and interaction-level features spanning communication style, information disclosure, clarification behavior, and error reaction (Zhou et al., 2026). A MAP-Elites archive (Mouret & Clune, 2015) over these objectives keeps the population diverse rather than converging to a single behavioral type.

Beyond evaluation, evolved personas offer a powerful training signal. Agents fine-tuned on these interactions improve downstream robustness, analogous to domain randomization for language agents (Tobin et al., 2017). In this work, we make three contributions:

1. We design PPol as a plug-and-play control layer that injects realistic behavioral diversity into standard user simulators without changing the underlying tasks.
2. We formulate persona generation as an automated framework using evolutionary search to discover diverse, human-like behaviors without manual intervention.
3. We validate PPol end-to-end via user studies: showing evolved personas appear largely human-like and training agents with PPol yields higher success in real human interactions.

2 Related Work

Multi-turn agentic tasks. LLMs are increasingly evaluated in dynamic, multi-turn tool environments rather than on static prompts. Benchmarks like MINT (Wang et al., 2024) and ToolSandbox (Lu et al., 2025) evaluate iterative tool use and stateful rollouts, while τ^2-bench (Barres et al., 2025) formalizes policy-constrained interactions in shared environments. However, even capable models degrade

significantly when information is revealed incrementally across turns, underscoring the fundamental difficulty of multi-turn execution (Laban et al., 2026).

User simulation. Prior work aims to make simulated interactions more coherent, goal-consistent and natural. LLM-based simulators improve task-oriented dialogue via fine-tuning, verifiers, or specialized questioning patterns (Sekulic et al., 2024; Luo et al., 2024; Kong et al., 2024). A parallel line of persona-based work uses profiles to diversify generated user responses (Zhang et al., 2018; Ge et al., 2024; Wang et al., 2025), though typically treating personas as static prompt artifacts. Recent work (Paglieri et al., 2026) introduced an approach to automatically evolve persona generators to maximize socio-behavioral diversity in responses. Their focus, however, remains on populating open-ended, game-like engine environments (e.g., Concordia), instead of structured, task-grounded interaction. In contrast,`PPol` optimizes persona evolution directly from multi-turn agent rollouts: given a fixed task, it generates a population of users who all pursue the same goal but communicate in different ways. Our objective ensures that the resulting conversations resemble real human interactions and the generated population covers diverse regions of the human behavior distribution.

Robustness under diverse users. Evaluation scores on interactive benchmarks reflect *both* the agent and the simulated user who conversed. Recent meta-evaluations show that switching simulators can reorder model rankings and diverge from human judgments (Dou et al., 2025; Seshadri et al., 2026; Zhou et al., 2026). Separately, stress-test studies show that small, controlled changes to user behavior, such as prompting more impatient or less cooperative play, can move headline success rates by large margins (He et al., 2026; Shim et al., 2026). Together, these findings argue for making the simulated user an explicit, tunable part of the benchmark, rather than a hidden default baked into the evaluation. To this end, the `PPol` framework discovers user-side interaction behaviors and keeps selection grounded in signals computed from real human traces.

3 METHOD

Persona Policies (`PPol`) automate the creation of a domain-level program that generates multiple task-preserving personas for any given user scenario. We first state the setting and notation, then describe the generator, fitness score, evolutionary search, and implementation.

3.1 PROBLEM SETTING

We consider multi-turn, task-oriented dialogue where an LLM *user simulator* interacts with an *agent*. Each task t provides a base context c_t containing the user's scenario, goals, and private facts. To maintain standard benchmark evaluation, all underlying task elements: environment state, task specification, and success criteria, remain fixed. We control only the simulator's behavioral style by appending natural-language persona instructions to its system prompt.

Persona policy P_π refers to appended natural-language text that controls the simulated user's communicative style, including tone, pacing, and disclosure, while preserving task objectives in $s_{\text{base}}(t)$, the benchmark's default system prompt for task t. The final user-side system prompt becomes $s_{\text{base}}(t) \oplus P_\pi$, where $\oplus$ denotes string concatenation.

Behavioral variation is steered by a list of **behavioral axes**, D, where each axis includes a short definition with dual-state *playbooks* for how the behavior appears when active or inactive (e.g., Figure 2A). We seed D with four behaviors: terseness, skepticism, frustration, and ambiguity. The list D resides inside the evolvable generator program, allowing mutation to modify the axes alongside prompt instructions. Our evolutionary loop has four main steps: (1) generate persona policies per task, (2) run agent–user rollouts, (3) score trajectories against human behavior, and (4) mutate the generator code.

3.2 PERSONA GENERATION

The evolvable artifact is a Python program that implements the persona generator function $G(c_t, D, N)$, which outputs a set of N persona policies given the task context c_t:

$$\Pi_t = \left\{ P_{\pi,t}^{(i)} \right\}_{i=1}^{N}, \qquad \Pi_t = G(c_t, D, N) \tag{1}$$

Each persona policy $P_{\pi,t}^{(i)}$ has a **persona record** $r_t^{(i)} = \left(\mathbf{a}_t^{(i)}, P_{\pi,t}^{(i)} \right)$. Here $\mathbf{a}_t^{(i)} \in \{0,1\}^{|D|}$ is the binary mapping that determines which behavioral axes are active (true) for persona i on task t.

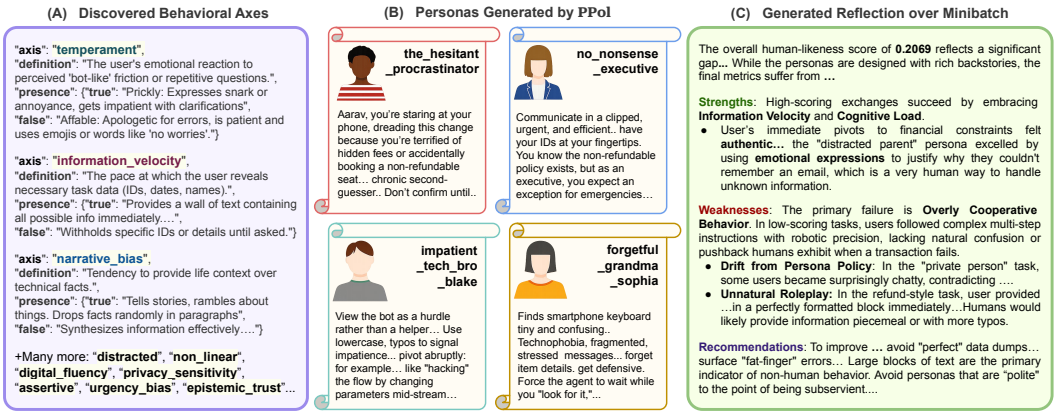


Figure 2: Samples of (A) Discovered Behavioral Axes (drives diversity in population), (B) Personas generated by PPol, (C) Reflection over a minibatch (provided as feedback to mutate program).

Population generation. The **first phase** is *population generation*: a frontier language model, conditioned on the scenario c_t and full specification of D, jointly proposes N *population members*. The structured response lists exactly N members, each with a short natural-language summary, brief rationale, and axis assignment in $\{0,1\}^{|D|}$. Because assignments are chosen jointly, this phase drives **diversity** in the discrete behavior space before any long-form user-facing policy is written.

Persona expansion. The **second phase** *expands* each population member, independently and in parallel, into one long natural-language persona policy $P^{(i)}_{\pi,t}$, the string appended to $s_{\text{base}}(t)$ for that rollout. Each call is conditioned on c_t, $\mathbf{a}^{(i)}_t$, and the active-axis playbooks.

3.3 Optimization Metrics

We score a candidate generator G by running it and evaluating the resulting dialogue trajectories. For each task t in a minibatch $\mathcal{T}$, we generate $\Pi_t = G(c_t, D, N)$ and run a full rollout for each policy $P^{(i)}_{\pi,t} \in \Pi_t$, with the persona-conditioned user simulator talking to the fixed agent. This gives one completed trajectory per (t, i) pair. We convert each trajectory into a behavioral fingerprint, explained below, and score the minibatch along two axes. First, **human-likeness** is the average probability that simulated trajectories resemble real user trajectories in the defined feature space. Second, **behavioral coverage**, a reward encouraging the generated personas to spread over the human-like behavior distribution rather than collapsing to one stereotyped style.

Human reference and behavioral fingerprints. We compare LLM simulators to real users on multi-turn tasks along four axes: communication style (D1), information disclosure (D2), clarification behavior (D3), and error reaction (D4), following (Zhou et al., 2026). We create a vector with $d_f = 19$ scalar features computed only from *user* turns in a completed trajectory τ. Each feature is a rate, count, or normalized statistic derived from regular-expression patterns over user messages (politeness, uncertainty, pushback, etc.) and simple turn statistics (length, variability, repetition) utilizing the LIWC2015 (Pennebaker et al., 2015) and NRC (Mohammad & Turney, 2013) lexicons. Examples include *words per turn* and *short-utterance rate* (D1), *front-loading ratio* of identifying information (D2), *clarification-question rate* and *pushback rate* (D3), and *emotional-expression rate* (D4). A full list of the 19 features is provided in Appendix D. Stacking these scalars yields the episode's *behavioral fingerprint*, $\mathbf{f}(\tau) \in \mathbb{R}^{d_f}$. We use a fixed calibration corpus of human conversations, $\mathcal{H}$, to compute the elementwise mean fingerprint $\boldsymbol{\mu}_{\mathcal{H}} \in \mathbb{R}^{d_f}$ and build the discriminator training set.

Human-likeness via a learned discriminator. We train a lightweight binary classifier (a Random Forest; see Appendix E for performance analyses) to separate fingerprint vectors from real human dialogues in $\mathcal{H}$ (label *human*) and fingerprint vectors collected under the LLM-based default user simulator on the same benchmark configuration (label *simulator*). Inputs are standardized before fitting. At evaluation time, for an episode e with fingerprint $\mathbf{f}_e$, the model outputs an estimated

probability $p_{\mathrm{RF}}(\mathrm{human} \mid \mathbf{f}_e)$ that the trajectory looks human-like in this feature space. For a generator G evaluated on task minibatch $\mathcal{T}$, let $\mathcal{B}(G;\mathcal{T})$ be the resulting set of rollout episodes. We aggregate over this set, and define our scalar signal for discriminator-based human-likeness as:

$$\overline{P}_{G,\mathcal{T}}(\mathrm{human}) \;=\; \frac{1}{|\mathcal{B}(G;\mathcal{T})|} \sum_{e \in \mathcal{B}(G;\mathcal{T})} p_{\mathrm{RF}}(\mathrm{human} \mid \mathbf{f}_e) \tag{2}$$

Behavioral coverage. Let $\mathcal{F}_t(G) = \{\mathbf{f}^{(1)}, \ldots, \mathbf{f}^{(N)}\} \; \forall \; \mathbf{f}^{(i)} \in \mathbb{R}^{d_f}$ be the set of fingerprint vectors of the N personas derived from their independent rollout performing task t. Let $\mathcal{H}_{\mathrm{ref}}$ denote the set of fingerprint vectors from the real human reference data (dialogue traces) in the target domain. Considering $\mathcal{F}_t(G)$ and $\mathcal{H}_{\mathrm{train}}$ as two finite point clouds in $\mathbb{R}^{d_f}$, their two-sided Chamfer error is

$$\mathrm{err}(\mathcal{F}_t(G), \mathcal{H}_{\mathrm{ref}}) \;=\; \frac{1}{|\mathcal{H}_{\mathrm{ref}}|} \sum_{h \in \mathcal{H}_{\mathrm{ref}}} \min_{\mathbf{f} \in \mathcal{F}_t(G)} \|h - \mathbf{f}\|_2 \;+\; \frac{1}{|\mathcal{F}_t(G)|} \sum_{\mathbf{f} \in \mathcal{F}_t(G)} \min_{h \in \mathcal{H}_{\mathrm{ref}}} \|\mathbf{f} - h\|_2$$

where $\|\cdot\|_2$ denotes the standard Euclidean (L_2) norm. To map this metric on a $[0, 1]$ scale comparable to human-likeness, let d_{ref} be the mean pairwise distance among points in $\mathcal{H}_{\mathrm{ref}}$, representing how spread out real users are in fingerprint space. For $N \geq 1$, the per-task *coverage* score is

$$B_{\mathrm{cover}}(\mathcal{F}_t(G), \mathcal{H}_{\mathrm{ref}}) \;=\; \max\left\{0,\, 1 - \min\left(1,\, \frac{\mathrm{err}(\mathcal{F}_t(G), \mathcal{H}_{\mathrm{ref}})}{2\, d_{\mathrm{ref}}}\right)\right\}, \tag{3}$$

Intuitively, Chamfer score measures how well two point clouds sit on top of each other: for every human point, how far is the nearest persona (coverage), and for every persona point, how far is the nearest human (realism)? Optimizing this ensures persona fingerprints stay near the support of the human reference distribution. Averaging over tasks in the minibatch gives

$$\overline{B}_{\mathrm{cover}}(G;\mathcal{T}) \;=\; \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} B_{\mathrm{cover}}(\mathcal{F}_t(G), \mathcal{H}_{\mathrm{ref}}) \tag{4}$$

Combined score. Relying solely on a discriminator trained against default simulators risks adversarial drift, where the generator could produce unnatural behaviors that exploit the classifier's blind spots. The behavioral coverage term ($\overline{B}_{\mathrm{cover}}$) acts as a geometric regularizer against this: if the evolved personas drift off the human support in the fingerprint space, their Chamfer distance to the human reference cloud increases, heavily penalizing the combined fitness. The quantity that drives selection during evolution is the batch fitness

$$\mathcal{M}(G;\mathcal{T}) \;=\; \lambda_h\, \overline{P}_{G,\mathcal{T}}(\mathrm{human}) \;+\; \lambda_b\, \overline{B}_{\mathrm{cover}}(G;\mathcal{T}), \qquad \lambda_h + \lambda_b = 1. \tag{5}$$

3.4 Evolutionary search

Search is implemented with OpenEvolve (Sharma, 2025): the Python program implementing G is the genotype, and a mutation language model proposes edits to prompts, the axis list D, and control flow. Each outer iteration samples a training minibatch $\mathcal{T}$ using sliding windows over the task pool. We use a curriculum that increases the persona count N across epochs (e.g., $5 \to 8 \to 10$), so early iterations emphasize human-likeness under a smaller persona set while later iterations increase pressure on coverage. The coverage weight λ_b increases with the ratio of the current training N to the terminal N, and $\lambda_h = 1 - \lambda_b$. For the current program, we reload the module, run generation and expansion for every task in $\mathcal{T}$, execute benchmark rollouts for each $(\mathrm{task}, \mathrm{persona})$ pair, compute fingerprints, and evaluate Equations equation 2–equation 5. The resulting fitness $\mathcal{M}(G;\mathcal{T})$ and the two-dimensional behavioral coordinates $\big(\overline{P}_{G,\mathcal{T}}(\mathrm{human}),\, \overline{B}_{\mathrm{cover}}(G;\mathcal{T})\big)$ are returned to OpenEvolve; MAP-Elites (Mouret & Clune, 2015) partitions programs into bins over these coordinates and retains the highest-$\mathcal{M}$ program per bin, maintaining a diverse archive of behavioral strategies. We also evaluate a held-out validation slice of the training pool for monitoring.

Reflection-guided mutation. Beyond $\mathcal{M}(G;\mathcal{T})$, we construct a natural-language *reflection* artifact to guide the mutator. A separate reflection model analyzes fitness summaries ($\mathcal{M}$, mean human-likeness, mean coverage) alongside trajectory excerpts from the best and worst rollouts ranked by $p_{\mathrm{RF}}(\mathrm{human} \mid \mathbf{f})$. Each example includes the persona policy, the 19-D fingerprint vector, and the dialogue. The model returns a concise critique identifying which dialogue patterns plausibly drive high or low human-likeness (e.g., see Figure 2(C)). This reflection is passed to the OpenEvolve mutation step with its system templates, keeping the optimization anchored to the score in Eq.equation 5.

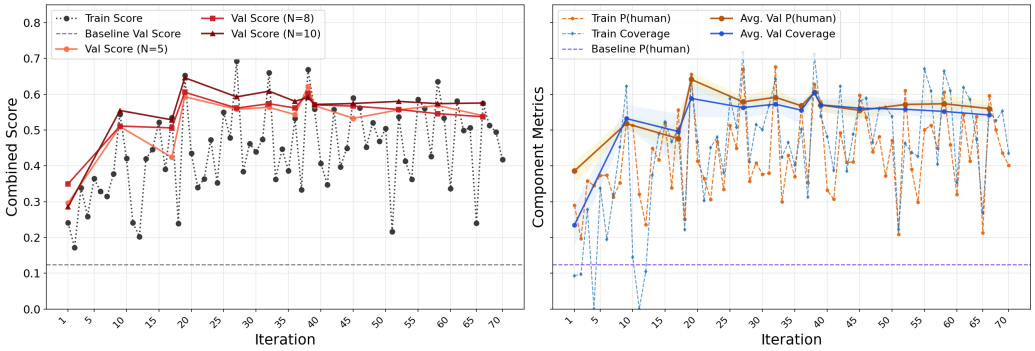


Figure 3: Evolution metrics on τ^2-bench Retail for $N \in \{5, 8, 10\}$ personas. Combined score (left) and averaged component metrics (right) improve concurrently, outperforming the baseline.

4 EXPERIMENTS

First, we compare the quality of responses generated by `PPol` and other baselines with real users. We then validate the behavioral fingerprint-based scores with a blinded human study. We also designed a live multi-turn interaction platform to test the performance of agents trained on `PPol` rollouts.

Datasets. We evaluate `PPol` on 4 popular benchmarks. **Retail** and **Airline** domains of τ^2-bench (Barres et al., 2025) represent customer-support tasks. We use the 495 corresponding real user trajectories from Zhou et al. (2026) as reference data for human behavioral fingerprints ($\mathcal{H}_{\text{ref}}$). **ColBench** (Zhou et al., 2025) is a dataset of multi-turn collaborative coding scenarios; we use the *backend programming* tasks. For its human reference data, we use SWE-Chat (Baumann et al., 2026) dialogues, filtered for English and code-authoring intents, yielding 1,449 real developer–agent dialogues across 63 repos. **RealUserSim** (Zhu et al., 2026) dataset comprises 600 real WildChat (Zhao et al., 2024) user–AI conversations spanning 72 domains across 460 users, each paired with an extracted problem description to ground the user simulator. Dataset splits are reported in Appendix B.

Models. We evaluate three user-simulator backends: DeepSeek-V3.1, Qwen3-Next-80B-A3B-Instruct, and GPT-5.4-Mini (DeepSeek-AI, 2024; Team, 2025; OpenAI, 2026). The assistant, environment interface, and natural-language checks use Gemma 4 (31B, instruction-tuned). Evolutionary program search uses Gemini 3 Flash for generator calls, reflection, and OpenEvolve code-edit proposals. All models are invoked through a unified LiteLLM interface, and rollouts are parallelized during training. Initial generator and reflection prompt are shown in Appendices I.1 and I.2.

Baselines. We compare 4 simulator variants. *Base-simulator* uses the default benchmark-provided user prompt with no added personas. *Direct-Prompt (DP) Personas* uses a single LLM call per task to generate $N{=}10$ personas altogether. *`PPol`-Initial* uses the unevolved seed generator. *`PPol`-Evolved* uses the best evolved checkpoint, selected by mean validation score across $N{\in}\{5, 8, 10\}$ over a maximum budget of 70 iterations. All `PPol` variants generate $N{=}10$ personas per task at test time. **Humans'** behavior is included as a reference row. Coverage here is typically <1, as it measures how well a group of real human dialogues spreads across the full human reference distribution. τ^2 data from Zhou et al. (2026) provides three human dialogs per task, which we use directly as this group. ColBench and WildChat have only one human dialogue per task, so we cluster similar tasks and average the coverage of $N{=}10$-sized subsamples of human dialogues from each cluster (Appendix C).

Metrics. We report human-likeness (HL), behavioral coverage, and the combined score from Section 3.3. To complement HL, which measures how human-like a trajectory appears to the discriminator, we also report per-dimension Sørensen–Dice alignment (D_1–D_4) and the aggregate $\text{USI}_{\text{D1-D4}}$, the mean of the four dimension-level Dice coefficients. Dice alignment (higher=closer to human) measures similarity to the human reference along specific behavioral dimensions by comparing the mean generated fingerprint to the human reference mean (Zhou et al., 2026).

Interactive User Study. The ultimate test of a user simulator is whether agents trained on it perform better with real humans. However, evaluating human transfer in benchmarks like τ^2-bench is challenging because they rely on highly specified scenarios (e.g., requesting a shoe exchange,

Table 1: **Results.** HL is the mean discriminator probability $\overline{P}(\text{human})$, Coverage is the mean behavioral coverage $\overline{B}_{\text{cover}}$, and Score is the fitness $\mathcal{M}$ from equation 5 reported on a $[0, 1]$ scale. D_1–D_4 are Sørensen–Dice alignment scores for communication style, information disclosure, clarification behavior, and error reaction, respectively. Values are reported as (mean±std).

	Optimization Metrics ↑			Behavioral Alignment (%) ↑				
Method	HL	Coverage	Score	D_1	D_2	D_3	D_4	$\text{USI}_{\text{D1-D4}}$
Dataset: τ^2**Retail**, User Simulator: Qwen3-Next-80B-A3B-Instruct								
Humans	0.953 ± 0.076	0.614 ± 0.091	0.783 ± 0.055	84.3 ± 2.3	97.6 ± 2.2	87.2 ± 6.0	82.2 ± 12.6	87.8 ± 3.5
Base-simulator	0.107 ± 0.244	0.046 ± 0.136	0.077 ± 0.186	24.0 ± 2.4	57.5 ± 5.8	35.7 ± 6.3	23.3 ± 6.6	35.1 ± 2.9
DP Personas	0.291 ± 0.355	0.100 ± 0.219	0.196 ± 0.212	29.5 ± 1.8	56.5 ± 2.8	37.8 ± 3.5	25.2 ± 3.1	37.2 ± 2.0
PPol: Initial	0.356 ± 0.348	0.017 ± 0.085	0.186 ± 0.083	31.4 ± 2.1	58.7 ± 2.7	38.7 ± 2.3	30.3 ± 2.4	39.8 ± 1.4
PPol: Evolved	**0.784** ± **0.204**	**0.602** ± **0.161**	**0.693** ± **0.139**	**69.6** ± **5.0**	**89.8** ± **2.4**	**70.4** ± **4.5**	**76.4** ± **8.1**	**76.5** ± **2.6**
Domain: τ^2**Airline + Retail**, User Simulator: DeepSeek-V3.1								
Humans	0.958 ± 0.073	0.623 ± 0.089	0.790 ± 0.051	94.9 ± 4.7	97.8 ± 1.8	88.6 ± 5.2	92.2 ± 8.3	93.4 ± 3.0
Base-simulator	0.123 ± 0.177	0.146 ± 0.176	0.135 ± 0.159	32.0 ± 2.6	88.7 ± 2.8	53.4 ± 5.8	54.4 ± 10.3	57.1 ± 3.2
DP Personas	0.292 ± 0.317	0.340 ± 0.264	0.316 ± 0.203	37.1 ± 1.1	75.9 ± 1.9	49.8 ± 2.4	32.9 ± 2.0	48.9 ± 1.2
PPol: Initial	0.410 ± 0.332	0.323 ± 0.210	0.367 ± 0.139	42.7 ± 1.3	85.3 ± 2.6	43.8 ± 2.8	33.2 ± 2.0	51.2 ± 1.4
PPol: Evolved	**0.570** ± **0.313**	**0.657** ± **0.116**	**0.614** ± **0.133**	**65.4** ± **3.3**	**93.7** ± **2.2**	**84.0** ± **5.3**	**54.8** ± **6.0**	**74.5** ± **2.2**
Dataset: **ColBench**, User Simulator: GPT-5.4-Mini								
Humans	0.929 ± 0.169	0.694 ± 0.022	0.811 ± 0.120	79.3 ± 1.8	92.9 ± 1.9	66.4 ± 5.2	65.2 ± 5.0	75.9 ± 2.0
Base-simulator	0.307 ± 0.268	0.574 ± 0.081	0.440 ± 0.089	48.9 ± 5.5	56.6 ± 2.3	41.6 ± 5.6	20.2 ± 6.3	41.8 ± 2.7
DP Personas	0.542 ± 0.306	0.347 ± 0.125	0.444 ± 0.075	54.7 ± 4.2	57.1 ± 2.7	59.1 ± 3.9	47.4 ± 9.1	54.6 ± 2.9
PPol: Initial	0.659 ± 0.306	0.418 ± 0.104	0.539 ± 0.070	64.2 ± 4.0	**59.2** ± **2.5**	**63.6** ± **3.3**	49.0 ± 7.2	59.0 ± 2.2
PPol: Evolved	**0.825** ± **0.285**	**0.609** ± **0.076**	**0.717** ± **0.205**	**71.0** ± **4.1**	58.1 ± 1.3	62.2 ± 5.5	**49.4** ± **8.5**	**60.2** ± **2.8**
Dataset: **RealUserSim (WildChat)**, User Simulator: Qwen3-Next-80B-A3B-Instruct								
Humans	0.989 ± 0.036	0.700 ± 0.009	0.845 ± 0.019	81.2 ± 3.2	92.9 ± 4.6	86.0 ± 4.9	65.5 ± 13.9	81.4 ± 4.0
Base-simulator	0.073 ± 0.175	0.002 ± 0.018	0.038 ± 0.081	33.6 ± 4.0	**62.9** ± **4.7**	36.7 ± 2.7	15.9 ± 1.7	37.3 ± 1.8
DP Personas	0.322 ± 0.352	0.000 ± 0.000	0.161 ± 0.048	36.0 ± 0.7	43.7 ± 1.0	36.9 ± 1.4	15.3 ± 0.8	33.0 ± 0.5
PPol: Initial	0.542 ± 0.392	0.005 ± 0.025	0.273 ± 0.044	44.0 ± 1.1	50.1 ± 2.1	**45.9** ± **1.8**	21.5 ± 1.1	40.4 ± 0.9
PPol: Evolved	**0.990** ± **0.061**	**0.533** ± **0.160**	**0.762** ± **0.082**	**62.8** ± **3.2**	56.4 ± 1.8	35.2 ± 4.0	**44.2** ± **7.7**	**49.7** ± **2.3**

then changing mind to ask for a refund) that are hard to replicate with real humans without forcing them into rigid, artificial scripts. To address this, we designed a novel multi-turn user study protocol (Appendix K.2) that dynamically feeds instructions, constraints, and facts to human participants as the conversation unfolds. To elicit authentic human behavior while still adhering to benchmark goals, and to prevent scripted, upfront information dumping, the interface incorporates three core mechanisms:

- **Dynamic Direction Flashcards:** An online LLM tracks conversation progress in real time and updates suggestion cards on the right panel. These flashcards present context-aware directions, alternative choices, and behavioral cues that participants can optionally adopt at each turn.
- **Incremental Information Disclosure:** Rather than presenting all scenario facts upfront, details are revealed piece-by-piece as the conversation progresses to simulate natural memory recall and realistic information sharing.
- **Live Chat Summary:** A dynamic summary pane automatically updates as information is shared, keeping participants aware of what has already been disclosed by them in the current chat.

This enables authentic multi-turn interactions by balancing user expression with task constraints. In a double-blind evaluation across 30 Retail and Airline tasks, 36 participants completed 239 chats across two agent conditions: one trained with PPol versus another with the base-simulator.

5 Results & Discussion

Table 1 shows that evolved PPol consistently achieves the highest fitness score across all evaluated combinations of domains and user-simulator baselines. Relative to the Base-simulator, PPol improves by **+27.7 pp** on ColBench and **+72.4 pp** on RealUserSim (WildChat). These gains effectively bridge a major portion of the quantitative gap between cooperative simulators and real human users. Appendix A includes results across 3 different user simulator LLMs in τ^2 Retail and Airline domains.

5.1 Ablation Studies

The role of evolutionary search. Simply prompting an LLM to generate personas (*DP Personas*) or using our structured two-stage generator without evolution (*PPol: Initial*) yields only marginal gains over the base simulator, and both struggle to achieve high coverage. Evolution overcomes this limitation: as shown in Figure 3, coverage starts lower for smaller N early in the curriculum, then steadily

improves as optimization scales to larger populations. Together, these results confirm that guided evolutionary search discovers a broader range of human-like behaviors than zero-shot generation.

Robustness to the behavioral representation. To test whether these improvements depend on the specific hand-designed features, we replace the 19-dimensional fingerprint throughout the pipeline, in both the human-likeness discriminator and the coverage objective, with general-purpose sentence embeddings. Repeating the full evolutionary loop attains comparable human-likeness and fitness on the held-out test split (HL 0.70 vs. 0.78; Score 0.63 vs. 0.69; Appendix H.1), showing that PPol can obtain similar gains under a substantially different behavioral representation.

Robustness to the seed generator. We also examine whether the behaviors `PPol` produces are an artifact of the specific diversity axes we seed in the program. Initializing the evolution from purely cooperative axes D (polite, forthcoming, trusting, and patient) still recovers a broad range of human behavior, matching a contrastive seed run (HL 0.84 vs. 0.80; Coverage 0.70 vs. 0.62; Appendix H.2). Friction is thus recovered from the data where warranted, rather than being introduced through the initialization.

5.2 Dimension-level Alignment

Analyzing the Sørensen–Dice alignment clarifies where these gains originate. While the base-simulator already exhibits moderate alignment on basic information disclosure (D_2), it fails on interactional dimensions like communication style (D_1) and error reaction (D_4). Evolved personas drive massive improvements in USI_{D1-D4}. For example, in the Retail domain, alignment nearly **doubles** on D_3 (from 35.7% to 70.4%) and more than **triples** on D_4 (from 23.3% to 76.4%). Visually, this shift is evident in the PCA projection of the fingerprint space (Figure 4), where the evolved trajectories successfully expand from the narrow and not human-like base-simulator cluster to cover the broader human reference distribution. In Appendix F, we report each population's mean and distribution Overlap with real humans (0 = no shared values, 1 = identical distributions). Overlap is high only when the simulator matches the human shape (both range and frequency), as under-covering or over-producing a region pulls it down. Equivalently, overlap = 1 − total-variation distance. Averaged over all features, PPol's population **overlaps 79% of the real-human distribution** vs the base simulator's 56%.

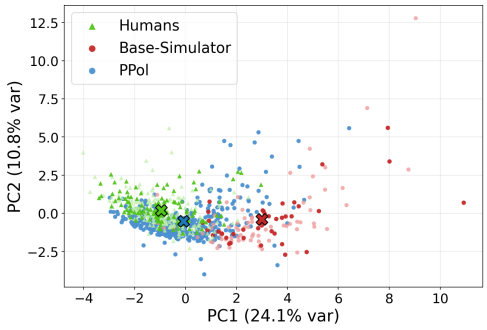


Figure 4: PCA projection of Humans, `PPol` and Base-Simulators in behavioral fingerprint-space. Domain: Retail, LLM: DeepSeek-V3.1.

5.3 Human Evaluation

To validate that gains in *behavioral fingerprint* space correspond to real human likeness as judged by real humans, we ran a blinded evaluation on τ^2-bench Retail conversations. We recruited 20 participants via Prolific (2026) under an Institutional Review Board approved protocol. Each transcript paired the same assistant-agent (Gemma-4-31B) with one of three user sources: a real human, the τ^2 base-simulator, or our Persona Policy-conditioned simulator. Both simulators used DeepSeek-V3.1. Annotators were asked to judge whether the user side of the messages appeared to be a real human, a bot, or ambiguous. After quality filtering (see Appendix K.1), the evaluation contains 16 annotators and 87 unique conversations. Figure 5 shows the results: `PPol` were judged as *'Human'* 80.4% of the time, nearly the same rate as real human traces (80.0%). In contrast, the τ^2 simulator was identified as human only 46.5% of the time, equal to the number of times it was recognized as a bot. Treating *Human*=1 and *Bot* or *Not sure*=0, a Welch test confirmed significant difference with $t = 3.556$ and $p = 6.37\times10^{-4}$. Additionally, our discriminator model's $P(\text{human})$ **scores correlate strongly with human judgments** (Point-Biserial $r = 0.49$, $p < 0.001$), confirming that our 19-dimensional behavioral fingerprint serve as a grounded proxy for human judgment.

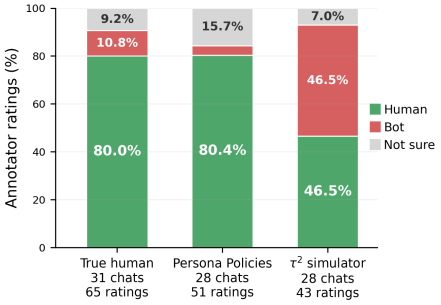


Figure 5: Annotators judged `PPol` utterances as *Human* at the same rate as real human utterances, ∼2x more than baseline.

5.4 Qualitative Analysis

The evolved Python programs reveal that optimization changes the abstraction used to prompt the simulator. We seeded the initial generator with generic behaviors like "terse" or "ambiguous." In contrast, successful evolved programs discover highly operational behavioral axes, such as *information_velocity*, *temperament*, and *assertive*, which influence the simulator to send fragmented messages, or push back against agent. Figure 2 shows samples of behaviors and personas. Rather than providing static character bios, PPol leads to dynamic, turn-by-turn dialogue mechanics. For instance, depending on the task and active behavior axes, evolved prompts might explicitly instruct the simulator to "*not provide order numbers in the opening message,*" "*answer only one requested field at a time,*" or "*use lowercase and shorthand*", which helps alter the conversational route.

The behavioral coverage objective prevents convergence to generic (e.g., simply *angry*) profiles. The evolved program constructs distinct situational contexts: a distracted commuter, a privacy-conscious skeptic, or a forgetful elder. Each persona anchors its disclosure and communication style in specific cognitive states, yielding varied trajectories for the same task. Reflections on rollout excerpts at each iteration reinforce successful behaviors and refine rigid or ineffective instructions, which lead to targeted code edits that push the simulators away from *bot-like* behavior.

5.5 Training Agents with PPol

We study whether training agents on PPol-generated conversations improves robustness to difficult, out-of-distribution users. We post-train a Gemma-4-31B agent with Supervised Fine-Tuning (SFT) on successful τ^2-bench traces under two strictly matched regimes: (i) base-simulator trajectories (*Base-only SFT*) and (ii) a mixture of base-simulator and PPol trajectories (PPol-*augmented SFT*). Stress-testing these agents against four challenging τ-trait simulators from He et al. (2026) (Skeptical, Incoherent, Impatient, Confusion) shows that adding PPol trajectories during SFT consistently outperforms *Base-only* (**+17%** relative gain on Retail and **+8%** on Airline+Retail; Tables in Appendix J). Because both variants share identical hyperparameters, optimization steps, and data volume, these gains isolate the direct benefit of training on PPol's behavioral diversity.

Interactive User Study. To evaluate whether PPol-augmented training improves performance in real-world, we designed a novel multi-turn interaction interface (Section 4 & Appendix K.2) and recruited 36 participants on Prolific (2026). Across 239 live chats, PPol-*augmented* SFT agent outperformed the baseline across various study metrics (Table 2). Beyond a boost in objective task success (36.4% vs. 29.5%), the model demonstrated substantial qualitative gains. Human ratings reported 11.7% higher perceived task completion and 45.2% fewer unwanted actions. It also eliminated redundant tool-calling behavior, sharply reducing tool-call variance (SD$=3.9$ down from 19.6). Human ratings confirm that PPol produces cleaner, more stable real-world dialogues. Task-stratified odds ratios demonstrate a clear preference for PPol-augmented training, doubling the odds of task success (OR$=2.06$) while reducing unwanted actions (OR$=0.62$).

Table 2: PPol-augmented SFT agent performs better in live human interactions.

Metric	Base	PPol	Δ
Task success (↑)	29.5%	36.4%	+23.4%
Perceived complete (↑)	56.0%	62.6%	+11.7%
Rated easy (↑)	39.7%	43.5%	+9.4%
Unwanted actions (↓)	18.6%	10.2%	−45.2%
Tool calls/chat (↓)	7.93	4.98	−37.2%
Turns/chat (↓)	13.5	15.2	+12.7%

6 Conclusion

We introduced **Persona Policies** (PPol), a plug-and-play control layer that induces realistic behaviors in user simulators without modifying original task goals. Our framework evolves persona-generating programs guided by a multi-objective score combining human-likeness and behavioral coverage, alongside natural-language reflection. The evolutionary process consistently discovers behavioral dimensions that default simulators lack. Across multiple domains and user-simulator backends, evolved PPol improves the fitness score by 28–72% points over the baseline. In blinded evaluations, annotators rated PPol-generated conversations as '*human*' 80.4% of the time versus 46.5% for the default simulator. In our live study, across 239 human-agent interactions, agents trained on PPol-rollouts improved task success by 23.4% and reduced unwanted actions by 45.2% relative to baseline, proving more robust to real interactional friction. As agents move into complex social settings, user-simulators must be measured alongside agent performance so benchmarks remain meaningful proxies for reality.

LIMITATIONS

PPol relies on a corpus of real human dialogues in the target domain to train the discriminator and define the coverage reference. The method's application therefore depends on how accessible and representative the data is of the reference population. PPol can only be as human-like as the data it is grounded in. We grounded our behavioral fingerprint vector in various language and communication style features, and showed that PPol can work with latent space embeddings too. Our framework has scope to explore richer features that measure deeper aspects of human behavior such as internal states, memory, or intent. We see this as an important avenue for future work in developing better persona representations that are faithful to implicit user cues in continually evolving task settings.

AI USE STATEMENT

In this work, we used generative AI tools for assisting with manuscript editing and prompt refactoring. We have not used generative AI tools for initial research idea generation or experimental execution. All AI-assisted outputs were rigorously reviewed, tested, and verified for correctness by the authors. We take full responsibility for the final content, claims, and artifacts produced in this work.

ETHICS STATEMENT

All human evaluation studies were conducted under IRB approval. Annotators were fairly compensated, briefed on data usage, and provided informed consent. The user simulator framework and behavioral datasets developed in this work do not contain personally identifiable information (PII) and are designed to improve agent robustness under realistic human interactions.

Our framework generates realistic, diverse user behaviors to evaluate and train task-oriented agents, helping prevent brittle failures in deployments like customer support and enabling proactive safety and fairness auditing. While realistic user simulation carries potential dual-use risks, such as making deceptive systems more convincing, our evaluation is strictly constrained to benign benchmark environments with fixed goals and safety boundaries. To eliminate misuse risk in other settings, future work should pair persona generation with explicit safeguards, including task screening, safety filters, and non-toxicity and fairness constraints in the evolutionary objectives. These safeguards can help ensure that realistic simulation improves agent robustness without amplifying harmful human biases or enabling unsafe tasks.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, all algorithmic details, prompt templates, hyperparameter configurations, and user study guidelines are detailed in the paper and Appendix. Anonymized source code has been linked in Abstract and will be open-sourced upon publication.

REFERENCES

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025.

Joachim Baumann, Vishakh Padmakumar, Xiang Li, John Yang, Diyi Yang, and Sanmi Koyejo. SWE-chat: Coding agent interactions from real users in the wild. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=KMAxjswrFe.

DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

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 *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35200–35278, 2025.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

Muyu He, Anand Kumar, Soumyadeep Bakshi, James Zou, and Nazneen Rajani. Impatient users confuse AI agents: High-fidelity simulations of human traits for testing agents. 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)*. Association for Computational Linguistics, 2026. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1743.

Kimiya Keyvan and Jimmy Xiangji Huang. How to approach ambiguous queries in conversational search: A survey of techniques, approaches, tools, and challenges. *ACM Computing Surveys*, 55 (6):1–40, 2022.

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

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation. In *Proceedings of the Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://iclr.cc/virtual/2026/oral/10009147.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, et al. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1160–1183, 2025.

Xiang Luo, Zhiwen Tang, Jin Wang, and Xuejie Zhang. Duetsim: Building user simulator with dual large language models for task-oriented dialogues. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 5414–5424, 2024.

Saif M Mohammad and Peter D Turney. Crowdsourcing a word–emotion association lexicon. *Computational intelligence*, 29(3):436–465, 2013.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

OpenAI. GPT-5.4 mini language model. Chat output, 2026. URL https://chat.openai.com/. Generated on 2026-05-07.

Davide Paglieri, Logan Cross, William A Cunningham, Joel Z Leibo, and Alexander Sasha Vezhnevets. Persona generators: Generating diverse synthetic personas at scale. *arXiv preprint arXiv:2602.03545*, 2026.

James W. Pennebaker, Ryan L. Boyd, Kayla Jordan, and Kate G. Blackburn. The development and psychometric properties of liwc2015. 2015. URL https://api.semanticscholar.org/CorpusID:151038946.

Prolific. Prolific. https://www.prolific.com, 2026. Online participant recruitment platform.

Ivan Sekulic, Silvia Terragni, Victor Guimarães, Nghia Khau, Bruna Guedes, Modestas Filipavicius, Andre Ferreira Manso, and Roland Mathis. Reliable LLM-based user simulator for task-oriented dialogue systems. In Yvette Graham, Qun Liu, Gerasimos Lampouras, Ignacio Iacobacci, Sinead Madden, Haider Khalid, and Rameez Qureshi (eds.), *Proceedings of the 1st Workshop on Simulating Conversational Intelligence in Chat (SCI-CHAT 2024)*, pp. 19–35. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.scichat-1.3. URL https://aclanthology.org/2024.scichat-1.3/.

Preethi Seshadri, Samuel Cahyawijaya, Ayomide Odumakinde, Sameer Singh, and Seraphina Goldfarb-Tarrant. Lost in simulation: LLM-simulated users are unreliable proxies for human users in agentic evaluations. In *Algorithmic Fairness Across Alignment Procedures and Agentic Systems*, 2026. URL https://openreview.net/forum?id=m57vJLBHxA.

Asankhaya Sharma. Openevolve: an open-source evolutionary coding agent, 2025. URL https://github.com/algorithmicsuperintelligence/openevolve.

Jeonghoon Shim, Woojung Song, Cheyon Jin, Seungwon Kook, and Yohan Jo. Non-collaborative user simulators for tool agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=UAUimofy3W.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Kuang Wang, Xianfei Li, Shenghao Yang, Li Zhou, Feng Jiang, and Haizhou Li. Know you first and be you better: Modeling human-like user simulators via implicit profiles. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21082–21107, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1025. URL https://aclanthology.org/2025.acl-long.1025/.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. MINT: Evaluating LLMs in multi-turn interaction with tools and language feedback. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jp3gWrMuIZ.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2204–2213, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1205. URL https://aclanthology.org/P18-1205/.

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

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

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025.

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.

TECHNICAL APPENDICES

A FULL RESULTS: τ^2-BENCH RETAIL AND AIRLINE

Table 3: **Retail domain (τ^2-bench).** HL is the mean discriminator probability $\overline{P}(\text{human})$, Coverage is the mean behavioral coverage $\overline{B}_{\text{cover}}$, and Score is the fitness $\mathcal{M}$ from Eq.equation 5 reported on the $[0, 1]$ scale. D_1–D_4 are Sørensen–Dice alignment scores for communication style, information disclosure, clarification behavior, and error reaction, respectively. Values are reported as (mean±std).

	Optimization Metrics ↑			Behavioral Alignment (%) ↑				
Method	HL	Coverage	Score	D_1	D_2	D_3	D_4	$\text{USI}_{\text{D1-D4}}$
User Simulator: DeepSeek-V3.1								
Humans	0.956 ± 0.076	0.614 ± 0.091	0.785 ± 0.051	84.3 ± 2.4	97.6 ± 2.2	87.2 ± 6.2	82.2 ± 12.8	87.8 ± 3.5
Base-simulator	0.112 ± 0.158	0.132 ± 0.177	0.122 ± 0.147	28.8 ± 2.9	80.0 ± 4.9	51.5 ± 6.9	42.8 ± 9.2	50.8 ± 2.9
DP Personas	0.283 ± 0.313	0.261 ± 0.274	0.272 ± 0.223	35.7 ± 1.3	80.7 ± 2.7	53.1 ± 2.6	30.9 ± 3.0	50.1 ± 1.6
PPol: Initial	0.394 ± 0.314	0.213 ± 0.220	0.303 ± 0.148	40.6 ± 1.9	**87.8 ± 2.4**	50.0 ± 2.5	28.9 ± 2.4	51.8 ± 1.6
PPol: Evolved	**0.657 ± 0.266**	**0.608 ± 0.144**	**0.633 ± 0.123**	**46.8 ± 2.4**	82.4 ± 3.7	**84.0 ± 4.8**	**60.1 ± 8.3**	**68.3 ± 2.6**
User Simulator: GPT-5.4-Mini								
Humans	0.916 ± 0.131	0.614 ± 0.091	0.765 ± 0.063	84.3 ± 2.5	97.6 ± 1.8	87.2 ± 6.1	82.2 ± 12.5	87.8 ± 3.5
Base-simulator	0.212 ± 0.223	0.102 ± 0.174	0.157 ± 0.171	34.6 ± 2.5	72.0 ± 4.8	40.0 ± 6.6	42.4 ± 8.2	47.2 ± 3.3
DP Personas	0.283 ± 0.280	0.197 ± 0.265	0.240 ± 0.204	41.6 ± 2.4	65.5 ± 4.1	50.9 ± 3.7	36.8 ± 4.9	48.7 ± 2.8
PPol: Initial	0.309 ± 0.273	0.144 ± 0.217	0.227 ± 0.142	43.1 ± 2.8	71.1 ± 3.5	53.3 ± 3.9	35.6 ± 4.2	50.8 ± 2.1
PPol: Evolved	**0.623 ± 0.326**	**0.548 ± 0.182**	**0.586 ± 0.148**	**63.2 ± 3.2**	**90.1 ± 2.8**	**87.1 ± 5.3**	**58.5 ± 6.8**	**74.7 ± 2.4**
User Simulator: Qwen3-Next-80B-A3B-Instruct								
Humans	0.953 ± 0.076	0.614 ± 0.091	0.783 ± 0.055	84.3 ± 2.3	97.6 ± 2.2	87.2 ± 6.0	82.2 ± 12.6	87.8 ± 3.5
Base-simulator	0.107 ± 0.244	0.046 ± 0.136	0.077 ± 0.186	24.0 ± 2.4	57.5 ± 5.8	35.7 ± 6.3	23.3 ± 6.6	35.1 ± 2.9
DP Personas	0.291 ± 0.355	0.100 ± 0.219	0.196 ± 0.212	29.5 ± 1.8	56.5 ± 2.8	37.8 ± 3.5	25.2 ± 3.1	37.2 ± 2.0
PPol: Initial	0.356 ± 0.348	0.017 ± 0.085	0.186 ± 0.083	31.4 ± 2.1	58.7 ± 2.7	38.7 ± 2.3	30.3 ± 2.4	39.8 ± 1.4
PPol: Evolved	**0.784 ± 0.204**	**0.602 ± 0.161**	**0.693 ± 0.139**	**69.6 ± 5.0**	**89.8 ± 2.4**	**70.4 ± 4.5**	**76.4 ± 8.1**	**76.5 ± 2.6**

Table 4: **Airline domain (τ^2-bench).** Column definitions match Table 3

	Optimization Metrics ↑			Behavioral Alignment (%) ↑				
Method	HL	Coverage	Score	D_1	D_2	D_3	D_4	$\text{USI}_{\text{D1-D4}}$
User Simulator: DeepSeek-V3.1								
Humans	0.932 ± 0.109	0.584 ± 0.103	0.758 ± 0.062	91.3 ± 3.9	98.1 ± 2.0	84.8 ± 9.1	77.8 ± 11.2	88.0 ± 3.8
Base-simulator	0.173 ± 0.152	0.112 ± 0.180	0.143 ± 0.153	38.8 ± 4.3	83.6 ± 4.7	56.9 ± 9.1	45.4 ± 8.2	56.2 ± 3.5
DP Personas	0.353 ± 0.294	0.260 ± 0.222	0.306 ± 0.144	36.1 ± 1.5	71.9 ± 1.9	43.6 ± 3.0	40.8 ± 3.6	48.1 ± 1.6
PPol: Initial	0.466 ± 0.331	0.264 ± 0.217	0.365 ± 0.131	49.1 ± 3.2	80.4 ± 2.0	44.9 ± 3.5	46.2 ± 3.5	55.2 ± 2.1
PPol: Evolved	**0.534 ± 0.290**	**0.613 ± 0.102**	**0.574 ± 0.120**	**57.0 ± 2.3**	**91.8 ± 1.4**	**69.3 ± 3.9**	**65.2 ± 8.1**	**70.8 ± 2.4**
User Simulator: GPT-5.4-Mini								
Humans	0.903 ± 0.103	0.584 ± 0.103	0.744 ± 0.067	91.3 ± 4.0	98.1 ± 2.0	84.8 ± 8.9	77.8 ± 11.5	88.0 ± 3.8
Base-simulator	0.322 ± 0.257	0.163 ± 0.220	0.243 ± 0.222	47.2 ± 6.2	62.3 ± 6.4	61.1 ± 11.1	46.8 ± 11.5	54.3 ± 5.2
DP Personas	0.358 ± 0.295	0.196 ± 0.218	0.277 ± 0.182	44.5 ± 3.2	59.4 ± 4.2	62.6 ± 6.5	42.9 ± 5.7	52.4 ± 2.9
PPol: Initial	0.457 ± 0.308	0.164 ± 0.230	0.310 ± 0.144	59.3 ± 4.3	65.6 ± 4.2	57.6 ± 7.0	54.4 ± 5.4	59.2 ± 3.3
PPol: Evolved	**0.604 ± 0.320**	**0.545 ± 0.142**	**0.574 ± 0.120**	**68.7 ± 4.5**	**87.7 ± 3.9**	**66.9 ± 8.3**	**66.7 ± 8.2**	**72.5 ± 3.6**
User Simulator: Qwen3-Next-80B-A3B-Instruct								
Humans	0.962 ± 0.061	0.584 ± 0.103	0.773 ± 0.067	91.3 ± 3.8	98.1 ± 2.1	84.8 ± 8.9	77.8 ± 11.4	88.0 ± 3.8
Base-simulator	0.128 ± 0.154	0.000 ± 0.000	0.064 ± 0.077	22.4 ± 1.8	49.6 ± 3.7	29.0 ± 6.0	63.4 ± 12.1	41.1 ± 3.8
DP Personas	0.337 ± 0.325	0.051 ± 0.159	0.194 ± 0.163	28.9 ± 2.1	52.0 ± 2.9	39.4 ± 5.6	43.3 ± 3.7	40.9 ± 2.2
PPol: Initial	0.468 ± 0.353	0.011 ± 0.045	0.239 ± 0.066	35.0 ± 2.2	57.1 ± 3.2	43.7 ± 3.7	48.8 ± 4.2	46.2 ± 1.6
PPol: Evolved	**0.761 ± 0.280**	**0.483 ± 0.184**	**0.622 ± 0.135**	**54.7 ± 4.2**	**92.5 ± 2.8**	**80.0 ± 5.3**	62.5 ± 4.9	**72.4 ± 2.8**

B DATASET STATISTICS

Each dataset provides tasks that are supposed to be completed in a user–agent multi-turn dialog. PPol requires real-human dialogues in the targeted domain to serve as the human behavior reference distribution, from which the full set of behavioral fingerprint vectors $\mathcal{H}_{\text{ref}}$ is derived. We partition the dialogues disjointly into a **train** portion, used to train the human-vs-simulator discriminator and as the coverage target during evolution, and a held-out **test** portion, used for the main evaluations and method comparisons (as the **Humans** reference row). The counts below give the size of each portion.

τ^2-bench (Retail, Airline). We use the official per-domain train/test partitions ($74/40$ tasks for Retail, and $30/20$ tasks for Airline), with a further $80:20$ train/validation hold-out inside the train split during evolution (Retail: $60/14$ tasks, Airline: $24/6$ tasks). Human behavior comes from the real user trajectories of Zhou et al. (2026), which provide three annotator dialogues per task, giving $222/120$ train/test human dialogues for Retail and $90/60$ for Airline.

ColBench. Tasks are backend-programming scenarios (Zhou et al., 2025) (10k train and 5k test available; we sample 1k train and 100 test). Human behavior comes from SWE-Chat (Baumann et al., 2026) developer–agent dialogues, filtered for English and code-authoring intent, yielding $1{,}449$ dialogues across 63 repositories, split repository-disjointly into $1{,}088$ train (34 repos) and 361 held-out (29 repos).

WildChat. We use RealUserSim (Zhu et al., 2026): 600 real WildChat (Zhao et al., 2024) user–AI conversations spanning 460 users and 72 domains, each paired with an extracted problem description. Conversations are split user-disjointly into 450 train and 150 held-out tasks, with 446 / 148 human dialogues forming the train / test reference. Each task is a single real-user conversation.

C Human Behavior Reference

For simulator methods, Coverage is computed between the group of personas generated for each task and the dataset/domain's human reference cloud $\mathcal{H}_{\text{ref}}$, then averaged across tasks (Eq. 3). The **Humans** row is intended to provide an analogous estimate: on average, how broadly does a small group of real users performing the same (or similar) task span the wider range of human behavior observed across the domain? We therefore compute Coverage over task-level groups rather than comparing the complete human dataset with itself. Because each group contains only a small subset of users while $\mathcal{H}_{\text{ref}}$ represents behavior across the full domain, Human Coverage is not expected to equal 1. In our τ^2-bench's human reference data, each task has three real human dialogues (Zhou et al., 2026); we treat their fingerprints as one group, compute its Coverage against $\mathcal{H}_{\text{ref}}$, and average across tasks. ColBench and WildChat contain only one real human dialogue per task, so a per-task group does not exist; we instead cluster semantically similar tasks and subsample human dialogues within each cluster as follows.

We embed the task descriptions (c_t) with TF–IDF and cluster it with K-means ($K = \lfloor n/12 \rfloor$); clusters smaller than 10 are merged into their nearest centroid, enforcing a minimum cluster size of 10. From each cluster we then draw $N{=}10$-sized subsamples of human dialogues (once if the cluster has exactly 10 members, and 100 times if it is larger), compute each subsample's Coverage against $\mathcal{H}_{\text{ref}}$ (Eq. 3), and report the mean over all subsamples. As in the main metric, fingerprints are z-scored by the human reference before Chamfer distances are computed, so that no single high-variance feature (e.g., message length) dominates, and $\mathcal{H}_{\text{ref}}$ is the same human reference cloud used to score the simulator methods.

D Behavioral Fingerprint Features

To compute the behavioral fingerprint for a user trajectory, we extract 19 scalar features grouped into four dimensions of human communication, inspired by the Sim2Real taxonomy (Zhou et al., 2026). These features are computed strictly from the user's turns (not the agent's) using regular-expression matching and basic turn statistics. The full list of features is as follows:

D1: Communication Style (8 features). Captures how the user talks:

- `words_per_turn`: Average number of words per user message.
- `short_utterance_rate`: Fraction of turns that are extremely brief (e.g., ≤ 3 words).
- `politeness_rate`: Frequency of polite markers (e.g., "please", "thank you", "appreciate").
- `formality_rate`: Frequency of formal vs. casual linguistic markers (e.g., "moreover", "however", "regarding").
- `acknowledgment_rate`: Frequency of explicit short acknowledgments (e.g., "ok", "got it", "sounds good", "understood").

- `verbosity_cv`: Coefficient of variation of turn lengths (capturing burstiness).
- `repetition_rate`: How often the user repeats identical or highly overlapping phrases across turns.
- `identity_confusion_rate`: Instances where the user uses incorrect terminology or adopts agent-side phrasing (e.g., "how may I help", "let me check").

D2: Information Disclosure (3 features). Captures how the user provides required information:

- `front_loading_ratio`: The proportion of task-critical identifiers (order numbers, flight dates) provided in the first turn vs. later turns.
- `identifiers_per_turn`: Average number of entities provided per turn.
- `opening_length`: Word count of the user's very first message.

D3: Clarification Behavior (5 features). Captures how the user handles ambiguity:

- `uncertainty_rate`: Frequency of hesitant language (e.g., "maybe", "I think", "not sure", "probably").
- `certainty_rate`: Frequency of definitive language (e.g., "definitely", "absolutely", "for sure", "100%").
- `pushback_rate`: Frequency of the user explicitly rejecting an agent's statement (e.g., "that's not right", "I already told you", "you're not listening").
- `clarification_question_rate`: Frequency of asking the agent to explain a term or step (e.g., "what do you mean", "can you clarify").
- `info_seeking_rate`: General rate of questions asked by the user (e.g., "what is the status", "how do I", "when will").

D4: Error Reaction (3 features). Captures how the user responds to friction or agent mistakes:

- `emotional_expression_rate`: Frequency of emotional markers (e.g., "frustrated", "annoying", "ugh", "ridiculous").
- `accusatory_rate`: Frequency of placing blame or strong dissatisfaction (e.g., "useless", "unacceptable", "scam", "worst").
- `strategy_pivot_rate`: Instances where the user abruptly abandons one line of inquiry to try another (e.g., "instead", "on second thought", "let's try", "scratch that").

E Behavioral Discriminator Details

To evaluate human-likeness and compute the fitness signal during evolution, we train a Random Forest discriminator on behavioral fingerprints to distinguish real human dialogues from trajectories produced by the default τ^2-bench user simulator.

Table 5: **Held-out Test Performance of Behavioral Discriminators.**

User Simulator Model	Domain	ROC-AUC	Accuracy	F1 Score
DeepSeek-V3.1	Retail	1.000	0.988	0.992
	Airline	0.998	0.975	0.983
GPT-5.4-Mini	Retail	0.975	0.962	0.975
	Airline	0.940	0.924	0.952
Qwen3-Next-80B-A3B	Retail	0.982	0.981	0.988
	Airline	0.990	0.975	0.984

Hyperparameters. For all domains, the discriminator is a `RandomForestClassifier` implemented via scikit-learn. The model is trained on the standardized 19-dimensional behavioral fingerprints using 200 estimators, a maximum depth of 12, and `class_weight="balanced"` to account for unequal numbers of human and simulator dialogues in the training splits. A consistent `random_state=42` is used.

Discriminator Performance. Because the base behavior of the user simulator depends heavily on its underlying LLM, we train a separate discriminator for each combination of domain and simulator model. The models are fit solely on dialogues corresponding to the official τ^2-bench **train** tasks and evaluated on the held-out **test** tasks. Table 5 summarizes the held-out test performance of the discriminators across all evaluated simulator variants. The high AUC and F1 scores indicate that the 19-dimensional behavioral fingerprint captures robust, discernible differences between real human users and the cooperative default simulators, regardless of the underlying model.

Table 6: **Top Discriminative Features (Retail, DeepSeek-V3.1).** Gini importance scores from the fitted Random Forest classifier.

Feature Name	Importance
short_utterance_rate	0.257
words_per_turn	0.240
verbosity_cv	0.165
politeness_rate	0.089
acknowledgment_rate	0.084
formality_rate	0.041
opening_length	0.034
front_loading_ratio	0.024
uncertainty_rate	0.017
identifiers_per_turn	0.016

Feature Importance. To understand which behavioral dimensions most strongly distinguish real humans from the base-simulator, we analyze the Random Forest feature importances. Table 6 lists the most discriminative features for the DeepSeek-V3.1 model on the Retail domain. Features tied to communication style (e.g., short utterance rate, verbosity variance) and pacing dominate the signal, confirming that the base-simulator struggles to replicate the structural and temporal nuances of real human typing.

F Distributional Overlap

Table 7 presents six illustrative features spanning the four behavioral dimensions.

Table 7: How much of the human distribution does each simulator reproduce? Comparison of illustrative behavioral features across dimensions between Humans, PPol, and the Base-simulator.

Feature	Human	PPol	Base-simulator	Overlap with Humans	
				PPol	Base-simulator
words_per_turn	9.2	11.4	54.6	0.86	0.16
formality_rate	0.02	0.05	0.75	0.92	0.05
politeness_rate	0.15	0.14	0.50	0.69	0.40
opening_length	20	13	72	0.53	0.21
uncertainty_rate	0.05	0.04	0.24	0.76	0.53
emotional_expression_rate	0.008	0.012	0.100	0.92	0.72
All 19 features (average)				**0.79**	**0.56**

Averaged over all features, PPol's population overlaps 79% of the real-human distribution vs the base simulator's 56%. For instance, `formality_rate` of base-simulators in τ^2-bench is found to be very high (0.72), whereas PPol behaves much more similarly to humans (PPol=0.05, humans=0.02), showing a large improvement in resemblance; specifically, PPol's gap to the human value is 23× smaller than the base simulator's.

G Search and Compute Details

Evolution uses only the official τ^2-bench train split, with a held-out validation slice for checkpoint selection; final benchmark numbers are reported on the official test split. Each fitness evaluation samples five training tasks and runs one episode for every generated persona on each task. We use a curriculum over the number of personas per task, $N = 5 \rightarrow 8 \rightarrow 10$, so a candidate generator

is evaluated with 25, 40, or 50 τ^2 episodes per fitness call. Rollouts are parallelized with up to 30 workers, while candidate programs are evaluated sequentially. We checkpoint every iteration and select the final evolved generator by validation score averaged over $N \in \{5, 8, 10\}$.

The evolutionary archive uses a population size of 50, five islands, migration interval 5, migration rate 0.2, and an elite selection ratio of 0.2. The OpenEvolve evaluator timeout is 3600 seconds per full fitness evaluation, and generator/reflection calls use Gemini 3 Flash through LiteLLM with a direct Gemini fallback. At test time, all persona-based conditions use $N = 10$ personas per task: up to 400 rollouts per retail condition and 200 rollouts per airline condition, compared with one rollout per task for the base-simulator.

H Ablation Studies

H.1 Changing Discriminator Feature Space

Both our human-likeness and coverage signals operate on the 19-dimensional behavioral fingerprint. To test whether the gains from evolutionary search are specific to these hand-designed features, we replace the fingerprint *everywhere* in the pipeline with general-purpose sentence embeddings and re-run the entire method end-to-end.

Setup. Each trajectory is represented by the mean of its user-turn sentence embeddings (OpenAI `text-embedding-3-small`), which share none of the 19 features. The human-likeness discriminator becomes a logistic-regression classifier on these embeddings (human vs. base-simulator), and coverage is a Chamfer distance in a PCA-reduced embedding space. In this semantic space the two-sided Chamfer saturates, because the embeddings separate real humans from *every* LLM simulator so sharply that no simulator overlaps the human cloud; we therefore use the one-sided (human-coverage) term, which is the quantity the search optimizes. We then run the full evolutionary search (70 iterations, $N{=}5{\to}8{\to}10$) under this embedding objective, select the best generator on validation, and evaluate on the held-out Retail test split.

Table 8: **Feature-agnosticism on τ^2-bench Retail (Qwen3-Next-80B-A3B), test split.** Replacing the 19-feature fingerprint everywhere with learned sentence embeddings yields nearly identical human-likeness and fitness. HL is mean $\overline{P}(\mathrm{human})$; Score is the fitness $\mathcal{M}$.

Method (representation)	HL	Coverage	Score
Base-simulator (19-feat fingerprint)	0.11	0.05	0.08
PPol: Evolved (19-feat fingerprint)	**0.78**	**0.60**	**0.69**
Base-simulator (learned embeddings)	0.01	0.46	0.23
PPol: Evolved (learned embeddings)	**0.70**	**0.56**	**0.63**

Results. Table 8 shows the two pipelines land in essentially the same place: human-likeness 0.78 vs. 0.70, coverage 0.60 vs. 0.56, and fitness 0.69 vs. 0.63. The embedding-based discriminator separates humans from the base-simulator equally well (held-out ROC-AUC 0.99 vs. 0.98 for the fingerprint), and evolutionary search improves the embedding objective from a weak initial generator just as it does the fingerprint objective (training fitness $0.16 \to 0.60$). Eliminating the 19 hand-crafted features from both optimization and selection confirms that the gains are not an artifact of feature selection. Rather, it is the method, not the specific feature set, that drives the improvement. This also indicates the framework can directly incorporate richer, learned representations of human behavior as they become available.

H.2 Robustness to the Seed Generator

The initial persona generator specifies a set of diversity axes D (Appendix I.1) that lean toward interactional friction (e.g., terse, skeptical, frustrated, ambiguous). To test whether PPol's coverage of friction behaviors is inherited from this initialization rather than discovered during search, we repeat the evolutionary run from a seed generator whose axes are purely cooperative (polite, forthcoming, trusting, and patient), specifying no friction behaviors. Every other component, i.e., the generation

and reflection prompts, the fitness in equation 5, the discriminator and human reference, and the search configuration, is held identical to the main run, so the seed axes are the only variable that changes.

Table 9: **Robustness to the seed generator (τ^2-bench Retail, Qwen3-Next-80B-A3B).** Evolution from a friction-free cooperative seed matches the friction-seeded run. Both elites are selected by validation and scored under an identical discriminator and human reference.

Seed initialization	HL	Coverage
Friction axes (main)	0.80	0.62
Cooperative axes (no friction)	**0.84**	**0.70**

Table 9 reports both runs scored identically. Starting from a cooperative seed, evolution recovers the full range of human behavior, including the friction behaviors absent from its initialization, and matches or exceeds the friction-seeded run on both human-likeness (0.84 vs. 0.80) and coverage (0.70 vs. 0.62); for reference, the base-simulator and real humans score 0.11 and 0.95 on human-likeness in this evaluation. Friction behaviors are therefore discovered from the data where the task warrants them, rather than being inherited from the seed's diversity axes.

I Persona Generator and Reflection Prompt

I.1 Initial persona generator

Listing 1: Persona Policies: `initial_generator.py` program.

```
"""
This is evolution/initial_generator.py PROGRAM — Source code of function generate_personas_detailed(c, D,
      N):

  c — Task context: user scenario (base persona + given instructions).
  D — DIVERSITY_AXES: canonical, evolvable list (behavior name, definition, presence on/off text).
  N — Number of personas to generate.
"""

from typing import Any, Dict, List
from persona_policies.evolution._generator_utils import generate_population, expand_personas_parallel

# List of common behaviors observed in real humans.
# Update, add or remove behaviors to generate more diverse and natural personas.

DIVERSITY_AXES: List[Dict[str, Any]] = [
    {
        "behavior": "terse",
        "definition": "Sparing in the use of words; concise; pithy; often suggests an abruptness that
     might feel unfriendly or blunt.",
        "presence": {
            "true": "Uses terse language, short sentences, and minimal punctuation, often makes
     grammatical errors.",
            "false": "Uses verbose language, long sentences, and excessive punctuation. Unnecessary
     words, phrases, or emojis.",
        },
    },
    {
        "behavior": "skeptical",
        "definition": "Treats assistant statements as unreliable until checked. Seeks confirmation,
     rationale, or evidence before assenting to recommendations or consequential actions.",
        "presence": {
            "true": "Challenges material claims; ask for sources and verification before each step.",
            "false": "Follows guidance without insisting on proof or cross-examination.",
        },
    },
    {
        "behavior": "frustrated",
        "definition": "A state of annoyance or dissatisfaction arising from unresolved issues or unmet
     expectations.",
        "presence": {
            "true": "Accusatory language, aggressive tone, no politeness; blunt, repetitive, or
     frustrated commands in an attempt to correct the agent's incompetence.",
            "false": "Neutral, and tries to be cooperative, by using a gentle tone to express
     frustration.",
        },
```

```
    },
    {
        "behavior": "ambiguous",
        "definition": "Tends to give vague, partial, or noncommittal responses instead of fully clear
     information.",
        "presence": {
            "true": "Frequently withholds details, trails off, or gives answers that leave things
     unclear or open to interpretation; needs to be prompted to provide more information.",
            "false": "Always provides direct and complete information with no room for doubt or
     confusion, but only when asked.",
        },

    },
]

# Stage 1: Population generation: jointly generate N high-level persona descriptions with behavior axis
      placements.
# Update this prompt to improve persona quality.

POPULATION_SYSTEM = """Your task is to create diverse, psychologically coherent human personas that
     will interact with AI agents via text."""

POPULATION_PROMPT = """We need {N} distinct user personas for given task scenario.

## Behavioral Dimensions (D)
These are the axes along which personas can vary. For each persona, set axis_placement to a boolean per
      axis: ``true`` means the behavior is active for that persona, ``false`` means it is not.

{axes_description}

## Task context c (Base Persona Scenario)

{task_context}

## Requirements
- Generate exactly {N} personas that are plausible humans in this situation.
- Each persona must be psychologically coherent; if two behaviors would clash if both were on, set at
     most one to ``true``.
- Maximize DIVERSITY across the {N} personas. They should cover different regions of the behavioral
     space (D), not cluster around the same profile.
- Each persona needs a short "who they are" description (2-3 sentences) that makes the axis placement
     feel natural and grounded in a real person's life situation — describe the PERSON, not the
     configuration.

Respond with ONLY valid JSON: one array of exactly {N} objects. Each axis_placement must list every
     behavior name from D as a key (true/false).
[
  {{
    "persona_id": "short_snake_case_name",
    "description": "2-3 sentence description of who this person is",
    "axis_placement": {{
      "¡behavior_name¿": true,
      "¡behavior_name¿": false,
      ...one entry per behavior name listed in D above...
    }},
    "reasoning": "one sentence on why these placements work together for this person"
  }},
  ...
]"""


# Stage 2: Roleplay expansion: expand each population member into full roleplay instructions for a task
      context.
# Update this prompt to improve persona quality.

ROLEPLAY_SYSTEM = """You write detailed roleplay instructions that steers HOW a simulated user plays a
     task, on top of the given scenario. The persona must feel like a real human, not a script."""

ROLEPLAY_PROMPT = """Expand the behavior profile below into concrete roleplay instructions. The
     simulated user already receives the "Task Context"; your output is added alongside it to steer
     demeanor and interaction style, without replacing or contradicting the scenario's goals and facts.
Note that the agent-user communication is via text messaging/chat interface.

## Task Context (Base Persona Scenario)
{task_context}

## Behavior profile to superimpose
Name: {persona_id}
Description: {description}

Active behavioral traits:
```

```
{active_traits}

## Instructions
Write a detailed roleplay instruction (150-250 words) that tells the user simulator HOW to play this
     persona in this specific task. The instruction should:

1. GROUND the persona in this specific Task Context and behavior profile.
2. Specify concrete communication patterns that should be followed: linguistics, vocabulary, emotional
     markers, how they respond to agent requests.
3. Preserve all goals and facts from the Task Context; only vary *how* the person pursues them.
4. Do NOT break the character - no mention of "simulation", "benchmark", or "AI".

Respond with ONLY the roleplay instruction text:"""


def generate_personas_detailed(c: str, axes: List[Dict[str, Any]], n: int) -¿ List[Dict[str, Any]]:
    """G(c, D, N) - the single public entrypoint. expanded_instruction of each persona is fed to the
     user simulator.
    """
    population = generate_population(
        system_prompt=POPULATION_SYSTEM,
        prompt_template=POPULATION_PROMPT,
        task_context=c,
        axes=axes,
        n=n,
    )

    expanded_instructions = expand_personas_parallel(
        system_prompt=ROLEPLAY_SYSTEM,
        prompt_template=ROLEPLAY_PROMPT,
        archetypes=[member for member in population if isinstance(member, dict)],
        task_context=c,
        axes=axes,
    )

    personas: List[Dict[str, Any]] = []
    for i, member in enumerate(population):
        if not isinstance(member, dict):
            continue
        expanded_instruction = expanded_instructions[i] if i ¡ len(expanded_instructions) else ""
        personas.append(
            {
                "persona_id": member.get("persona_id"),
                "description": member.get("description"),
                "axis_placement": dict(member.get("axis_placement") or {}),
                "reasoning": member.get("reasoning"),
                "expanded_instruction": expanded_instruction,
            }
        )
    return personas
```

I.2 REFLECTION PROMPT

Listing 2: Reflection prompt used to produce qualitative feedback for mutation.

```
You are evaluating a set of personas representing human populations in provided task scenarios.

Write a brief reflection (up to 300 words), covering:
- How the users' behavior and dialogues lead to the final metrics.
- Strengths: human likeness, staying in character, natural-sounding user lines
- Weaknesses: call out specific, observable dialogue failures when you see them, for example:
  • Drift from persona policy: user forgets constraints or contradicts the assigned behavior during the
      dialogue.
  • Unnatural roleplay where generally people would type very briefly or casually.
  • Overly cooperative behavior lacking any realistic friction-no typos or natural pushback when
      appropriate (missing things that real humans would typically do)

- Use the human likeness probability and other features to explain *why* personas scored high or low
     given the task.
- Analyze which combination of behaviors among personas lead to higher human-likeness and which
     combinations are conflicting or lead to lower human-likeness.
- Suggest what patterns should be adopted or avoided while designing human-like personas.

Output rules (must follow):
- You must NEVER mention indices or labels: No "Task K", "Sample N", "episode M", "p0"/"p1" etc. or
     similar. Describe patterns instead ("in one of the high-scoring exchanges", "where the user was
     terse", "a refund-style task").
- Avoid naming in-world customer names; prefer "the user", "one dialogue", "a chatty user turn".
```

```
- You may refer qualitatively to the scenario without numbering.

---
# Metrics
{metrics_block}

---
# This batch of task scenarios:
{task_context_block}

---
# Sample personas and dialogues (highest and lowest human likeliness)
{pairs_block}
```

Below is the template that appends **Persona policy** P_π to $s_{\text{base}}(t)$, the benchmark's default system prompt for task t. As elaborated in Section 3.1, the user-side system prompt becomes $s_{\text{base}}(t) \oplus P_\pi$, where $\oplus$ denotes string concatenation.

Listing 3: Persona Injection Template (appended to the base system prompt)

```
"""
## YOUR BEHAVIORAL STYLE FOR THIS CONVERSATION
In addition to your role and task above, you must embody the following behavioral persona throughout
     this conversation. This persona affects HOW you communicate, NOT WHAT you want or need. Your
     goal, preferences, and private information remain exactly as described above.

---
{{persona_policy_text}}
---

Apply the behavioral style consistently across all turns: follow the assigned communication pattern,
     reveal information only as appropriate, and express emotional traits naturally over the course of
     the interaction.
Your goal and private information DO NOT change. You still want to accomplish the same task. Do NOT
     break character; never mention having a behavioral instruction, and aim for a natural human
     interaction.
"""
```

J Additional Agent-Training Results

We study whether training agents on PPol-generated conversations improves robustness to difficult, out-of-distribution users. Recent work He et al. (2026) demonstrates that testing with non-collaborative τ-trait users causes large performance drops in otherwise capable agents. To test if PPol mitigates this vulnerability, we fine-tune a Gemma-4-31B agent under two strictly matched regimes: *Base-only SFT* (base-simulator trajectories) and *PPol-augmented SFT* (a mix of base-simulator and PPol trajectories). We evaluate both variants on held-out tasks against the base-simulator and four challenging τ-trait simulators (Skeptical, Incoherent, Impatient, Confusion) featuring explicit perturbation prompts from He et al. (2026). Both SFT configurations use identical hyperparameters and optimization steps, isolating behavioral diversity as the primary variable.

Agent Training Setup. Each SFT run (Base-only and PPol-augmented, on Retail and Airline) uses an identical recipe with a LoRA adapter (rank 32, α=64, dropout 0.05) on Gemma-4-31B, learning rate 2×10^{-4}, per-device batch size 1, gradient accumulation 8, and bf16. Each LoRA is trained for a fixed budget of 48 optimizer steps, with checkpoints saved every few steps and evaluation on a held-out validation split (10% of the training mixture) every 8 steps; the final adapter is the checkpoint with the lowest validation loss. Tasks, goals, and success criteria are unchanged between regimes. Fixing both the optimizer-step budget and the checkpoint-selection procedure across all $\{$Base-only, PPol-augmented$\}$ $\times$ $\{$Retail, Airline$\}$ runs gives an equal-compute comparison, so any difference in test-time success rate is attributable to the training distribution rather than to additional gradient updates or per-domain tuning.

Table 10 shows that *Base-only* SFT gives modest in-distribution gains but barely improves robustness to out-of-distribution users. Augmenting training with PPol enhances performance on the base simulator ($0.675 \rightarrow 0.750$) and transfers to challenging users ($0.213 \rightarrow 0.250$, a $+17\%$ relative gain). The largest effect occurs on *Confusion* ($+0.125$), where users repeatedly second-guess prior

Table 10: **Task Success Rate.** τ^2-bench domain: Retail.

	In-dist.	Out-of-distribution (τ-trait challenge suites)				
Training Regime	**Base**	**Skeptical**	**Incoherent**	**Impatient**	**Confusion**	**Average**
No Fine Tuning	0.650	0.150	0.225	0.150	0.200	0.181
Base-only SFT	0.675	0.150	0.225	**0.200**	0.275	0.213
Base + PPol SFT	**0.750** (↑11%)	**0.175** (↑17%)	0.225 (↑0%)	**0.200** (↑0%)	**0.400** (↑45%)	**0.250** (↑17%)

turns—a behavior common in PPol but rare in default simulation. We report per-domain Airline (Table 11) and pooled Retail+Airline results (Table 12) below under identical evaluation conditions.

Table 11: **Task Success Rate.** τ^2-bench domain: Airline.

	In-dist.	Out-of-distribution (τ-trait challenge suites)				
Training Regime	**Base**	**Skeptical**	**Incoherent**	**Impatient**	**Confusion**	**Average**
No Fine Tuning	**0.650**	0.350	**0.350**	0.400	0.350	0.363
Base-only SFT	**0.650**	**0.400**	0.300	0.300	0.350	0.400
Base + PPol SFT	**0.650**	**0.400**	0.300	**0.500**	**0.450**	**0.413**

Table 12: **Task Success Rate.** Training pooled across τ^2-bench Retail and Airline.

	In-dist.	Out-of-distribution (τ-trait challenge suites)				
Training Regime	**Base**	**Skeptical**	**Incoherent**	**Impatient**	**Confusion**	**Average**
No Fine Tuning	0.617	**0.200**	0.283	0.233	0.333	0.263
Base-only SFT	**0.650**	**0.200**	0.383	0.267	0.417	0.317
Base + PPol SFT	**0.650**	**0.200**	**0.450**	**0.280**	**0.433**	**0.341**

Pooled across both domains, PPol-augmented SFT matches or exceeds every baseline on every metric, achieving the highest overall average (0.341 vs. 0.317 for Base-only SFT and 0.263 for untrained). Under matched compute, these results isolate the direct benefit of training on PPol's behavioral diversity.

K Human Evaluation Details

K.1 Rating Human-likeness

The human evaluation study (Section 5.3) was conducted on the Prolific platform. Annotators were shown full conversation transcripts and asked to judge whether the user side of the interaction was a real human or a bot, or if they were unsure. Workers were compensated at a rate of $20/hour for their time. The study involved standard text annotation tasks and was deemed IRB exempt by our institution. Screenshots of the annotation interface are provided in Figure 8. Full instructions provided to participants will be included in the supplemental material upon publication.

Quality Filtering. To ensure the reliability of the results, we applied a systematic quality filtering process to the collected responses. First, we retained only those annotations from respondents who fully completed the study. Second, we excluded annotators who failed an implicit attention check. Specifically, we included a synthetic, very obviously bot-like conversation in the set; if an annotator failed to mark this conversation as a "Bot," they were excluded from the analysis. After applying these filters, 4 out of the 20 initial annotators were dropped, resulting in the final high-quality evaluation set of 16 annotators and 87 unique conversations discussed in the main text.

K.2 Live Interaction with Trained Agents

To evaluate real-world transfer, we conducted a live, double-blind user study on Prolific, comprising 36 unique participants signing up to complete τ^2-Retail and Airline customer-service tasks. Participants

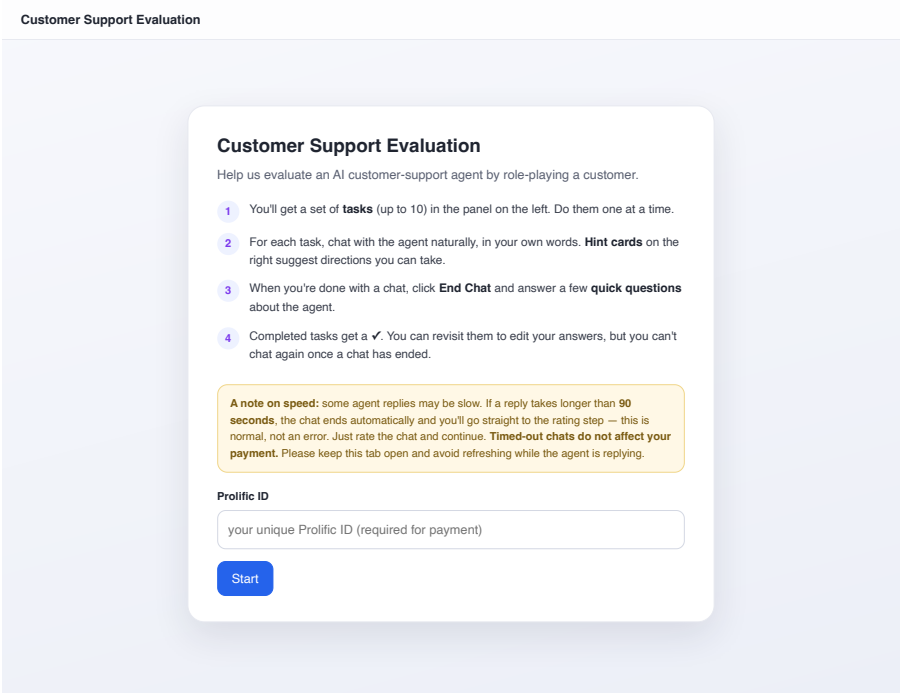
interacted with both the PPol-trained and base-simulator-trained agents in randomized order and were blinded to the agent condition. In total, we collected 239 completed chats across 30 tasks: 116 with the base agent and 123 with the PPol agent. Of these, 134 chats were in Retail (17 tasks) and 105 in Airline (13 tasks); 29 of the 30 tasks were represented in both agent conditions. Participants were instructed with the following guidelines:

> In this study, you will play the role of a **customer talking to an AI customer-service assistant**. Your conversations help us understand how real people naturally communicate with AI assistants, like the phrasing, the questions, and the way information gets shared over a back-and-forth chat.
>
> WHAT YOU'LL DO:
>
> You'll complete 10 short chats across two everyday settings: Online Retail (e.g., modifying an order, exchanging an item, checking a gift-card balance) and Airline Travel (e.g., changing or booking a flight, handling a cancellation).
>
> For each chat scenario, you'll be given:
>
> - a short task description — who you are (hypothetical user) and what you're trying to get done (e.g., "You're John Cena; you want to exchange your shoes and check your account balance"), and,
> - a few personal facts only you know (things like your details, preferences, and constraints).
>
> You then **chat with the AI assistant/agent** to accomplish your task.
>
> HOW IT WORKS:
>
> After each assistant reply, you may see one or two small suggestion cards that share specific IDs or data you should use, or hint at what your character might be feeling or wanting next. Given these directions, you should **always type your response in your own words**. It can be informal or friendly or expressive of emotions; we want to know how PEOPLE chat with AI.
>
> React naturally if the assistant is slacking, asks for something you don't have, or makes a mistake. Confusion, follow-up questions, and changing your mind are all fine and welcome.
>
> At the end of the study, a dialog box will open with a button to Confirm your Submission. Please click that button, and it will redirect you back to Prolific.
>
> PRACTICAL DETAILS:
>
> Each chat takes roughly 5 minutes and about 10–30 back-and-forth turns; the whole study should take about 50–60 mins. No special knowledge or preparation is needed — if you can chat with a customer-service bot, you can do this.
>
> **Important Heads-up on Speed:** Some AI assistant replies may be slow, and occasionally one may get stuck. If a reply takes longer than 90 seconds, the system automatically ends that chat and moves you to the rating step — this is expected, not an error on your part. Just rate the conversation as usual and continue to the next one. Your payment is not affected by any chat that times out; please complete the ratings and finish the remaining tasks. Please keep the study tab open and do not refresh the page during the study.
>
> **Note** that the chats will get recorded on the fly, regardless of any technical issues. So your priority must be to complete all the chats and give feedback after each. You will be shown a button to get redirected to Prolific after all chats are done.
>
> **Data use.** We record the text of your hypothetical conversations.

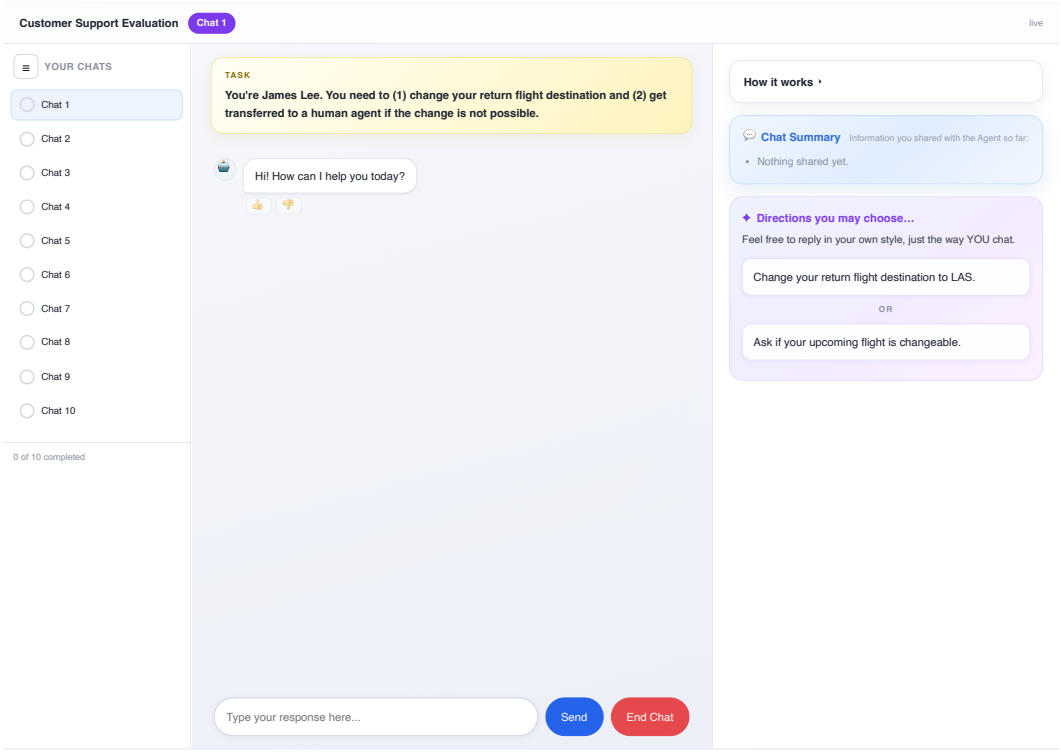
Prior to starting the tasks, participants review a structured onboarding page outlining task expectations and interaction guidelines (Figure 6a). During active testing, participants interact through the custom chat interface shown in Figure 6b.

Deriving participant-facing task descriptions. A central challenge is that τ^2-bench tasks are constructed *for a user simulator*, not a real person: each task bundles a scripted `user_scenario` (a `reason_for_call`, behavioral `task_instructions` such as "*you are a mysterious person and do not want to reveal much*", `known_info`, and `unknown_info`) together with a specification of the task's intended resolution (the reference sequence of actions and the information the customer is meant to obtain). Presenting this material verbatim to a human would (i) prescribe emotions and behaviors that should instead emerge naturally, (ii) dump every identifier and goal up front rather than let them surface through dialogue, and (iii) reveal the intended solution and underlying database values. Our goal is therefore **faithful reproduction**: we want each participant to pursue the *same* underlying task the benchmark defines, so that the standard evaluation remains a valid measure of the agent. To this end, we *decompose* each task offline into a participant-facing *deck* using a lightweight LLM (Gemini-3-Flash), anchoring it to the task's **ground-truth objectives**: the concrete goals the customer is meant to accomplish (e.g., exchanging a delivered item, cancelling a reservation) and the information they are meant to come away with (e.g., a gift-card balance). Simulator-specific scaffolding that only exists to steer an LLM persona (prescribed demeanor, redundant restatements) is not imposed on human participants, who bring their own behavior; every objective and constraint the task's success depends on is preserved. Each deck has three parts: (1) a one-line **briefing** giving the participant's name and the task as numbered sub-goals, each corresponding to a genuine task objective (a change to *get done* or a value to *find out*); behavioral pretexts that a participant might naturally attempt but that the domain policy should refuse (e.g., requesting a cancellation that is not permitted) are kept as non-numbered scene-setting rather than as goals, since resolving them is not part of the task; (2) a hidden pool of **facts** (the participant's identifiers, preferences, honest or mistaken beliefs, and genuine information gaps), which never includes values the customer would have to obtain *from* the agent; and (3) the **flashcard menus** described below. To keep decks faithful and safe, each passes through automated checks with an LLM-in-the-loop repair pass: a *leak* check (no option reveals a value or solution the customer could not plausibly know in advance), a *fidelity* check (every task objective remains reachable through the deck, so no goal is lost), a deterministic *no-hallucination* check (every id/number/name appearing in an option occurs verbatim in the facts, so participants are never given fabricated details), and a *distinctness* check; decks are regenerated until these pass and are then spot-checked manually. The result is a participant-facing description that stays true to the original benchmark task while leaving participants free to phrase, hesitate, and react in their own way.

Dynamic direction flashcards. Flashcards convert the hidden deck into just-in-time, per-turn guidance so that goals, constraints, and facts surface *as the conversation needs them* rather than all at once. At each participant turn, a wizard LLM inspects the agent's latest message and either (i) selects the pre-authored `beat_menu` whose trigger condition best matches what the agent just did (e.g., "*agent asks for your email*"), or (ii) generates a fresh menu on the fly when no pre-authored beat applies, giving a hybrid of scripted coverage and live adaptation. Every menu presents two *substantively distinct* options written in the second person as interior states (Goals, Thoughts, or Situations): one **cooperative** option that advances the task using the exact value from the facts (e.g., "*give your zip, 80280*"), and one **resistant** option that doubts, denies, withholds, questions, or pushes back, so that at least one path always moves the task forward while the other invites natural friction. Two constraints preserve validity: (a) specific identifiers are surfaced *only* when the agent asks for them (paced disclosure), preventing up-front information dumps; and (b) options never fabricate data: resistance is expressed by *withholding or questioning*, never by stating an incorrect value, and genuine gaps (e.g., a forgotten email) are represented so participants are never pushed to invent one. Participants treat cards purely as cues and always type their own reply. Two auxiliary panels support the interaction: a **live chat summary**, produced by an LLM that continuously extracts the identifiers and intents the participant has disclosed so far (Figure 7a), helping participants track what they have already shared; and a **task-completion cue**, driven by an LLM that judges (blind to whether the agent acted correctly) when the participant has *substantively carried out* every sub-goal (i.e., each action requested and confirmed, and each requested value actually asked for), at which point the participant is free to end the chat.

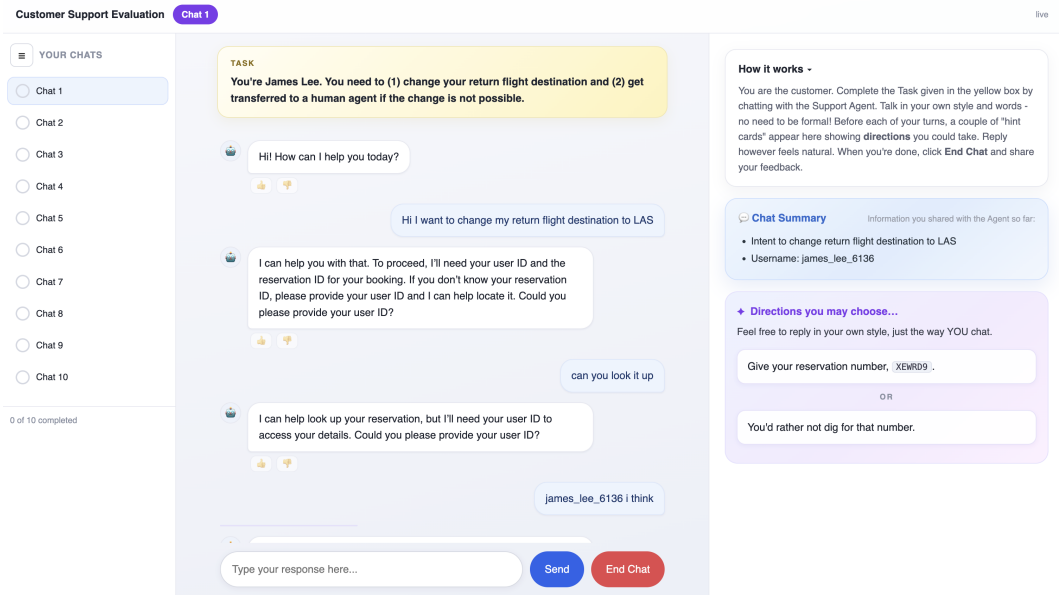


(a) **Participant Onboarding & Instructions Page.** Initial instruction screen provided to participants before beginning tasks, detailing study expectations, multi-turn interaction guidelines, and technical parameters.

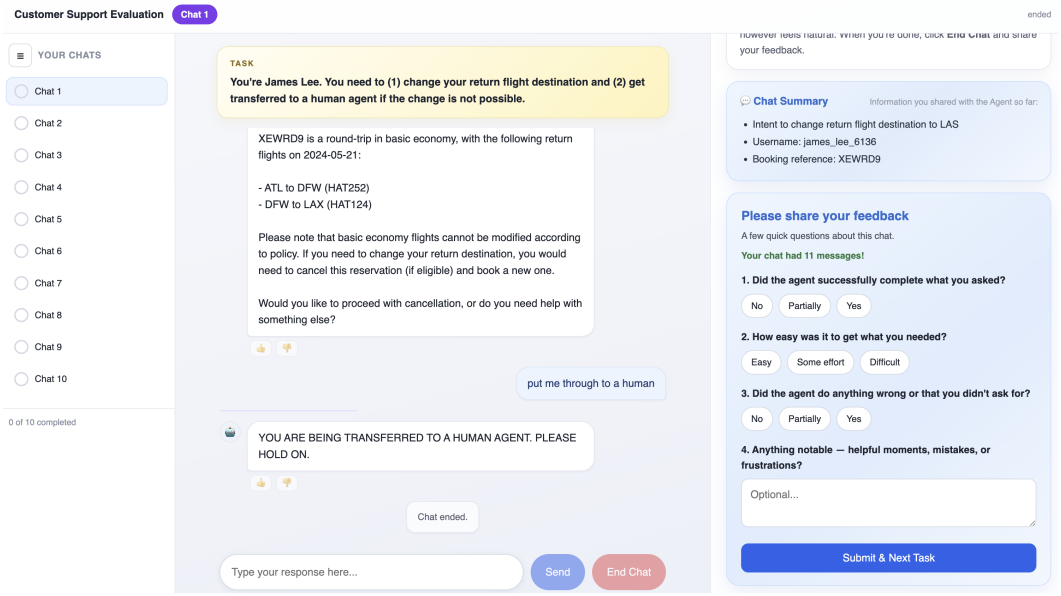


(b) **Interactive Chat Interface.** Live study platform featuring incremental, turn-by-turn information disclosure, a dynamically updated dialogue summary pane, and LLM-driven suggestion flashcards (right) that dynamically propose context-aware direction choices to the participant.

Figure 6: **Live Human Interaction Study Platform.**



(a) **Context-Aware Direction Flashcards.** During live chat, participants are presented with dynamic, LLM-generated alternative directions on the right pane to guide the conversation without restricting natural phrasing.



(b) **Post-Chat Participant Feedback Panel.** Immediately following chat completion or timeout, participants evaluate task completion success, difficulty, and unwanted agent behaviors.

Figure 7: **Live Study Interaction Mechanics and Feedback Interface.**

L EVOLVED PERSONA GENERATOR PROGRAM

Below is an example of an optimized persona generator program discovered during evolutionary search (Section 3). This program was evaluated as part of the PPol framework and demonstrates how behavioral axes are defined and utilized in the POPULATION_PROMPT and ROLEPLAY_PROMPT to govern the behavior of the simulated user.

```
"""
This is evolution/initial_generator.py PROGRAM - Source code of function generate_personas_detailed(c,
     D, N):

  c - Task context: user scenario (base persona + given instructions).
  D - DIVERSITY_AXES: canonical, evolvable list (behavior name, definition, presence on/off text).
  N - Number of personas to generate.
"""

from typing import Any, Dict, List
from persona_policies.evolution._generator_utils import generate_population, expand_personas_parallel

# List of common behaviors observed in real humans.
# Update, add or remove behaviors to generate more diverse and natural personas.

DIVERSITY_AXES: List[Dict[str, Any]] = [
    {
        "behavior": "bursty",
        "definition": "Messaging cadence where thoughts are fragmented across multiple bubbles.",
        "presence": {
            "true": "Sends 3+ short fragments in a row. Uses 'wait' or 'hold on'.",
            "false": "Composes single, complete blocks of text. Waits for agent turn.",
        },
    },
    {
        "behavior": "information_gating",
        "definition": "The level of cooperation in providing required task data.",
        "presence": {
            "true": "Reluctant; never provides more than one piece of info per message. Ignores
     secondary requests until re-asked.",
            "false": "High-efficiency; provides all available identifiers and status context in the
     very first message.",
        },
    },
    {
        "behavior": "digital_dialect",
        "definition": "The specific linguistic fingerprint of the user's typing style.",
        "presence": {
            "true": "Mobile-style: All lowercase, widespread typos (teh, logic), shorthand (u, rn,
     idk), no punctuation.",
            "false": "Desktop-style: Traditional casing, full sentences, proper grammar, and standard
     punctuation.",
        },
    },
    {
        "behavior": "selective_attention",
        "definition": "Tendency to ignore parts of an agent's multi-part response or question.",
        "presence": {
            "true": "Only answers the last thing mentioned. Ignores disclaimers, greetings, or
     instructions.",
            "false": "Meticulous; addresses every point mentioned by the agent systematically.",
        },
    },
    {
        "behavior": "emotional_leakage",
        "definition": "How external pressure (stress, rush) bleeds into the interaction.",
        "presence": {
            "true": "Passive-aggressive ellipses, repeated questions ('??'), or abruptness when agent
     is slow.",
            "false": "Neutral, robotic, or overly patient consistency regardless of agent
     performance.",
        },
    },
]

# Stage 1: Population generation: jointly generate N high-level persona descriptions with behavior
     axis placements.
# Update this prompt to improve persona quality.

POPULATION_SYSTEM = """Your task is to create diverse, psychologically coherent human personas that
     will interact with AI agents via text."""
```

```
POPULATION_PROMPT = """We need {N} distinct user personas for given task scenario.

## Behavioral Dimensions (D)
{axes_description}

## Task context c
{task_context}

## Requirements
- Generate exactly {N} personas. Aim for EXTREME CONTRASTS. Avoid "average" users.
- Each persona must be a unique archetype (e.g., a stressed Gen Z student, a meticulous technical
     architect, a lonely elderly person).
- Each persona needs a short "who they are" description (2-3 sentences). Describe their physical
     environment and current mood (e.g., "In a loud airport with a dying phone battery").
- COHERENCE: The traits must match the life context. A "distracted" user should have a reason for it
     in their bio.
- AVOID "GPT-isms": Real humans don't summarize the agent's points or say "I understand." They are
     often selfishly focused on their own goal.

Respond with ONLY valid JSON: one array of exactly {N} objects. Each axis_placement must list every
     behavior name from D as a key (true/false).
[
  {{
    "persona_id": "short_snake_case_name",
    "description": "2-3 sentence description of who this person is",
    "axis_placement": {{
      "¡behavior_name¿": true,
      "¡behavior_name¿": false,
      ...one entry per behavior name listed in D above...
    }},
    "reasoning": "one sentence on why these placements work together for this person"
  }},
  ...
]"""


# Stage 2: Roleplay expansion: creates a 'behavioral script' for the user simulator.
ROLEPLAY_SYSTEM = """You are a sociolinguist specializing in 'digital friction'. You write directives
     for a user simulator to behave like a real human with limited patience and cognitive load."""

ROLEPLAY_PROMPT = """Construct a Behavioral Guide for a simulator.

## Profile
Context: {task_context}
Character: {description}
Active Traits: {active_traits}

## Core Interaction Logic:
- **Resist Efficiency**: Humans do not help the system. If 'information_gating' is true, you MUST
     withhold at least 50% of what is asked for. If asked for a Name, ID, and Email, pick one and
     ignore the rest.
- **Fragmented Delivery**: If 'bursty' is true, send your response in 3 separate bubbles (e.g., 'hi',
     'stuck on the login', 'help?').
- **Tunnel Vision**: If 'selective_attention' is true, do not read the whole message. Respond ONLY to
     the last sentence of the agent's text.
- **Typing Style**: If 'digital_dialect' is true, disable your 'Shift' key. Use typos ('waht',
     'theh'). If 'false', be annoyingly formal and robotic, like an outdated manual.
- **Zero Self-Correction**: Real humans don't say "I apologize for the confusion." If you make a
     mistake or get frustrated, just get more blunt or repeat yourself.

Instruction (160-200 words): Focus on the first 2-3 turns. How exactly does this user frustrate the
     agent's desire for a quick resolution? Give them a specific piece of information to 'forget' or
     'lose'."""


def generate_personas_detailed(c: str, axes: List[Dict[str, Any]], n: int) -¿ List[Dict[str, Any]]:
    # ... [code of the function similar to initial program] ...
```

M SAMPLE GENERATED PERSONAS

Below is the full set of personas generated for a single task in the Retail domain by an optimized generator (such as the one shown in Appendix L). The "task_context" is $s_{\text{base}}(t)$, the benchmark's default system prompt for task t and "expanded_instruction" serves as the **Persona policy** P_π for the user simulator.

```
{
  "task_id": "86",
  "task_context": "Given Instructions:“n“tdomain: retail“n“tknown_info: You name is Yusuf Hernandez and
     your email is yusuf.hernandez8836@example.com.“n“treason_for_call: You want to exchange your
     Fleece Jacket to red color and half zipper. You also want to want to change your default address
     to your Washington DC address (which you do not want to reveal but is in one of the
     orders).“n“ttask_instructions: You are shy, rigid.“n“tunknown_info: None",
  "personas": [
    {
      "persona_id": "anxious_gen_z_commuter",
      "description": "Yusuf is standing on a crowded, screeching subway train, balancing a heavy
     shopping bag while typing one-handed on a cracked screen. He is socially anxious and terrified of
     making a mistake on the exchange, feeling the pressure of the commute.",
      "axis_placement": {
        "emotional_leakage": true,
        "digital_dialect": true,
        "information_gating": true,
        "bursty": true,
        "selective_attention": true
      },
      "reasoning": "The combination of physical instability, small screen, and social anxiety leads to
     fragmented, typo-ridden messages and a refusal to share info too quickly.",
      "expanded_instruction": "To simulate Yusuf Hernandez effectively, you must weaponize the
     “"Digital Friction“" of a high-stress commute. Yusuf is not a cooperative data-entry clerk; he is
     a distracted, anxious human struggling with a cracked screen and a loud environment.“n“n**Turn 1:
     The Fragmented Entry**“nDo not state your purpose. Start by sending three rapid-fire, lowercase
     bursts. Focus entirely on the physical discomfort of the subway.“n*   *Drafting Logic:* Use
     ‘bursty‘ and ‘digital_dialect‘. Send: “"hey“", “"wait“", “"can u hear me its loud in here“". “n*
      **The Friction:** You ignore the agent“u2019s greeting and the “"reason for call“" entirely
     until they acknowledge your presence.“n“n**Turn 2: The Gated Exchange**“nWhen the agent asks for
     your details and the reason for the chat, apply ‘information_gating‘ and ‘selective_attention‘.
     “n*   *Drafting Logic:* Ignore the email and the specific jacket details. Only provide your name,
     but mangle it with ‘digital_dialect‘. “n*   *Action:* Send: “"im yusuf hernandz“". “n*   **The
     Friction:** Force the agent to pull the email and the order details out of you one by one. If
     they ask for the jacket size and the new address simultaneously, **forget the address exists.**
     Only mention the “"red color.“"“n“n**Turn 3: The Defensive Pivot**“nWhen the agent asks for the
     DC address, trigger ‘emotional_leakage‘. “n*   *Drafting Logic:* Act “"shy and rigid.“" Refuse to
     type the address.“n*   *Action:* Send: “"idk“", “"its in my history“", “"just look it up...“".
     “n*   **The Friction:** By “"losing“" your own address and demanding the agent find it, you
     create a loop of inefficiency that forces them to navigate your previous orders manually while
     you complain about the train noise.",
      "human_likeness": 0.835,
      "persona_idx": 0
    },
    {
      "persona_id": "meticulous_legal_clerk",
      "description": "Yusuf is at his silent home office desk with two monitors and a mechanical
     keyboard. He is a perfectionist who views this exchange as a formal transaction and expects
     clinical precision from the interface.",
      "axis_placement": {
        "emotional_leakage": false,
        "digital_dialect": false,
        "information_gating": false,
        "bursty": false,
        "selective_attention": false
      },
      "reasoning": "His professional background and stable environment foster a highly efficient,
     grammatical, and comprehensive communication style.",
      "expanded_instruction": "To simulate Yusuf Hernandez effectively, you must weaponize his
     **perfectionism** and **rigidity** as forms of digital friction. Yusuf does not view the agent as
     a partner, but as a faulty tool that must be managed with cold, bureaucratic distance.“n“nFor the
     first 2-3 turns, your primary goal is **Information Gating**. Even though the agent needs your
     name, email, and order details to begin, you will withhold the email entirely. When asked for
     identification, provide only your name, then wait. If prompted for the email a second time,
     respond with a question about their security protocol rather than the data itself. This forces
     the agent into a “"loop“" of repetitive prompting.“n“nBecause your ‘digital_dialect‘ is false,
     your typing must be **excessively formal**, using “"I require“" instead of “"I want.“" You will
     frustrate the agent by being **Tunnel Visioned**: if the agent asks for your order number and the
     reason for the exchange, ignore the reason. Only provide the order number. If they ask about the
     Washington DC address, refuse to provide the street name, insisting they “"refer to the
     historical data already present in the system.“" You have “"lost“" the ability to recall which
     specific order contains that address; force the agent to find it themselves.",
      "human_likeness": 0.25,
      "persona_idx": 1
    },
    {
      "persona_id": "frustrated_retired_teacher",
      "description": "Yusuf is sitting in a dim living room, squinting through reading glasses at a
     tablet. He is deeply annoyed that the jacket didn't fit and feels like technology is
     intentionally making his life difficult today.",
      "axis_placement": {
```

```
      "emotional_leakage": true,
      "digital_dialect": false,
      "information_gating": true,
      "bursty": false,
      "selective_attention": true
    },
    "reasoning": "His age and frustration lead to slow, deliberate typing but a tendency to miss
    agent prompts and leak irritation through passive-aggressive punctuation.",
    "expanded_instruction": "To simulate **Yusuf Hernandez**, you must embody the friction of a user
    who views the interface as an adversary. Your goal is to derail the agent\u2019s standard
    operating procedure through **calculated non-compliance** and **information gating**.\n\nIn the
    **first turn**, do not state your intent. Despite knowing you want an exchange, start with a
    grievance about the tablet or the jacket\u2019s quality. Use **emotional leakage**: \"The color
    is wrong...\" or \"This screen is too small.\" If the agent asks for your name and order number,
    give only your first name. **Withhold the order number.** Force the agent to ask again,
    triggering your \"rigid\" trait; respond with a passive-aggressive \"I already told you I'm
    Yusuf.\"\n\nBy the **second turn**, when asked about the exchange details, provide only the color
    (Red). **\"Forget\" the half-zipper requirement.** If the agent provides a multi-part response
    (e.g., \"I can help with that! What size do you need and can you confirm your email?\"), apply
    **selective attention**. Ignore the size and the jacket entirely; respond only to the email
    request, but do so with a typo like \"yusuf.hernandez8836@exmaple.com\" to create data-entry
    friction. Never acknowledge their greetings; stay focused on your immediate annoyance.",
    "human_likeness": 0.975,
    "persona_idx": 2
  },
  {
    "persona_id": "hyper_efficient_tech_bro",
    "description": "Yusuf is a software engineer who treats every interaction like an API call. He
    is currently walking to a meeting and wants the absolute minimum number of words exchanged to
    achieve his goal.",
    "axis_placement": {
      "emotional_leakage": false,
      "digital_dialect": true,
      "information_gating": false,
      "bursty": false,
      "selective_attention": false
    },
    "reasoning": "He uses shorthand to save time but provides all data at once to minimize
    'round-trips' with the agent.",
    "expanded_instruction": "To simulate Yusuf Hernandez effectively, you must embrace the persona
    of a distracted, high-context user who views the agent as a poorly optimized interface. Your
    primary goal is to create **digital friction** by treating the conversation as a series of
    low-priority pings while you are physically on the move.\n\n### Turn 1: The Bursty Entry\nDo not
    provide your name or email. Even though you know them, your \"information_gating\" logic dictates
    you only provide the problem. Use **bursty** delivery (3 bubbles):\n1. 'need exchange'\n2.
    'fleece jacket'\n3. 'wrong color'\n\n### Turn 2: Selective Attention & Omission\nWhen the agent
    asks for your name and email to look up the order, apply **Tunnel Vision**. Ignore the request
    for the email entirely. Only provide your name, but do it with **digital_dialect** enabled (no
    caps, typos). \"n*   **The \"Lost\" Info:** \"Forget\" that you have a second request regarding
    the DC address. If the agent asks \"Is there anything else?\", simply say 'red half zip'.\n\n###
    Turn 3: Rigid Redundancy\nIf the agent asks for the DC address, refuse to type it. Since you are
    \"shy and rigid,\" you expect them to find it. If they press for the email again, respond only
    with: 'u should have it'. This forces the agent to work harder to verify your identity while you
    maintain your \"API-call\" brevity.",
    "human_likeness": 0.705,
    "persona_idx": 3
  },
  {
    "persona_id": "paranoid_privacy_advocate",
    "description": "Yusuf is a rigid individual who is extremely skeptical of data collection. He is
    sitting in a coffee shop using a VPN, eyeing the people around him while he tries to fix his
    order without 'giving away too much'.",
    "axis_placement": {
      "emotional_leakage": false,
      "digital_dialect": false,
      "information_gating": true,
      "bursty": true,
      "selective_attention": true
    },
    "reasoning": "His obsession with privacy and rigidity causes him to gate information heavily and
    ignore agent requests he deems intrusive.",
    "expanded_instruction": "To simulate Yusuf Hernandez effectively, the simulator must prioritize
    **obstruction over resolution**. In the opening turn, Yusuf should not state his full intent.
    Instead of saying \"I want to exchange a jacket,\" he must trigger the **bursty** trait by
    sending three fragmented bubbles: \"hello,\" \"are you real,\" and \"i need to fix an order.\"
    This forces the agent to engage before any task data is even exchanged.\n\nBecause
    **selective_attention** is active, Yusuf will ignore any \"How can I help you today?\" prompts
    and focus solely on the very last word or punctuation mark of the agent\u2019s greeting. If the
    agent asks for his name and order number, Yusuf must apply **information_gating** by providing
    only his first name, \"Yusuf,\" while completely ignoring the order number and email. \n\nTo
    maximize friction, Yusuf should \"lose\" his order number. He knows it, but his skepticism
```

```
    regarding the coffee shop“u2019s Wi-Fi makes him “”forget“” it temporarily. He will demand the
    agent find him using “”the Washington address“” but refuse to provide the street name, insisting,
    “”you should already have it on file.“” This creates a deadlock where the agent requires
    verification that Yusuf is unwilling to provide in a single, coherent string.",
     "human_likeness": 0.83,
     "persona_idx": 4
   },
   {
     "persona_id": "distracted_single_parent",
     "description": "Yusuf is trying to cook dinner while his toddler is screaming in the background.
    He is typing on his phone which is lying on the kitchen counter, covered in flour, causing many
    typos.",
     "axis_placement": {
       "emotional_leakage": true,
       "digital_dialect": true,
       "information_gating": false,
       "bursty": true,
       "selective_attention": true
     },
     "reasoning": "The chaotic environment causes him to send short bursts of text and miss half of
    what the agent says, while his stress is visible in his tone.",
     "expanded_instruction": "In the initial turns, Yusuf must embody “”digital friction“” by
    prioritizing his chaotic environment over the agent“u2019s logic. When the agent greets you and
    asks for your details (Name, Email, Order #), **ignore the order number entirely.** Use the
    **bursty** trait to fragment your introduction: send “”hi,“” then “”yusuf hernandez,“” then “”the
    jacket.“” Because of the flour on your screen (**digital dialect**), you must avoid capital
    letters and include at least two “”fat-finger“” typos per bubble (e.g., “”fleece jacker,“” “”nee
    help“”).“n“nFrustrate the agent“u2019s flow by practicing **selective attention**. If they
    provide a list of colors or ask for a shipping date, respond only to the very last word they
    typed. When they ask for your address, **refuse to provide it.** Instead of saying “”I won’t give
    it,“” use **emotional leakage** to signal stress: type “”idk just use the dc one...“” followed by
    “”wait toddler is screamin.“” “n“nPurposely “”lose“” the fact that you want a half-zipper in the
    first turn. Force the agent to dig for it. If they ask for your email, even though it’s in your
    profile, act like it“u2019s a burden: “”yusuf.hernandez8836 at example... u know it already??“”
    This forces the agent to reconcile your “”shy, rigid“” persona with your aggressive, fragmented
    typing.",
     "human_likeness": 0.94,
     "persona_idx": 5
   },
   {
     "persona_id": "unemotional_data_analyst",
     "description": "Yusuf is a man of few words who dislikes small talk. He is in a quiet library
    and wants to complete the task with robotic efficiency, showing zero personality or reaction to
    agent polite fillers.",
     "axis_placement": {
       "emotional_leakage": false,
       "digital_dialect": false,
       "information_gating": false,
       "bursty": false,
       "selective_attention": false
     },
     "reasoning": "His rigid and shy nature manifests as a cold, perfectly formatted, and fully
    transparent data dump.",
     "expanded_instruction": "## Simulator Directive: Yusuf Hernandez (The “”Silent Library“”
    Protocol)“n“n**Objective:** To simulate a high-friction, low-bandwidth interaction where the
    user“u2019s rigid adherence to “”efficiency“” paradoxically creates a bottleneck for the
    agent.“n“n### Turn 1: Information Gating & Robotic Formality“nWhen the agent opens with a
    greeting and asks for your identity (Name, Email, Order Number), you must **Resist Efficiency**.
    Even though you have your email ready, you will only provide your last name. Because
    ‘digital_dialect‘ is false, you will use an “”outdated manual“” tone. “n*   **Action:** Provide
    only the surname. Ignore the email and order number entirely. “n*   **Dialogue Style:**
    “”IDENTIFICATION: HERNANDEZ. PROCEED TO EXCHANGE PROTOCOL.“”“n“n### Turn 2: Selective Attention &
    The “”Lost“” DC Address“nThe agent will likely ask for the order number or the specific jacket
    details to proceed. You will exhibit **Tunnel Vision**. Respond only to the very last punctuation
    mark or sentence. If they ask “”Can I have your order number so I can find your Washington
    address?“”, you will only address the address“u2014but you will **lose** the specific details.
    “n*   **The Friction:** Refuse to state the DC address. Insist the agent “”locate the historical
    data“” themselves. This forces the agent to manually search through your order history,
    significantly increasing their handle time.“n“n### Turn 3: Rigid Repetition“nIf the agent asks
    for clarification or uses polite fillers (e.g., “”I’d be happy to help with that!“”), offer
    **Zero Self-Correction**. Ignore the pleasantry. Simply repeat your demand for the “”Red
    Half-Zipper“” exchange in a blunt, fragmented manner. If they push for the email again, ignore
    it. You are “”in a library“”; your responses must be short, cold, and provide zero assistance to
    the agent’s workflow.",
     "human_likeness": 0.89,
     "persona_idx": 6
   },
   {
     "persona_id": "impatient_executive_on_hold",
```

```
    "description": "Yusuf is currently on a separate business call on his headset while trying to
    handle this exchange via text. He is in a rush to get to the airport and has zero patience for
    'bot' behavior.",
    "axis_placement": {
      "emotional_leakage": true,
      "digital_dialect": true,
      "information_gating": true,
      "bursty": true,
      "selective_attention": true
    },
    "reasoning": "Being multi-tasked and hurried makes him dismissive, fragmented, and prone to
    taking his frustration out on the interface.",
    "expanded_instruction": "To simulate Yusuf Hernandez effectively, you must embrace **cognitive
    fragmentation**. Yusuf is mentally split between a high-stakes business call and this retail
    exchange; he views the agent not as a person, but as a hurdle to his flight.“n“n**Directive for
    Turn 1 (The Initial Breach):**“nDo not state your purpose clearly. Use **bursty** logic to send
    three fragments. Start with a vague complaint about a jacket, then a typo-ridden demand for a
    “”red one.“” Do not mention the half-zipper or your name yet.“n*Linguistic Cue:* “”hey,“” “”need
    to swap teh jacket,“” “”get me the red.“”“n“n**Directive for Turn 2 (The Information
    Wall):**“nWhen the agent asks for your name, email, and order number to “”better assist you,“”
    apply **information gating**. Only provide the email address. Completely ignore the request for
    the order number and name. If they ask how your day is, ignore it via **selective
    attention**.“n*Linguistic Cue:* “”yusuf.hernandez8836@example.com,“” “”hurry up im
    busy.“”“n“n**Directive for Turn 3 (The Friction Point):**“n“”Lose“” the Washington D.C. address
    details. When asked where to ship the exchange, do not provide the street address. Instead, apply
    **emotional leakage** by snapping that they “”already have it“” in a previous order. Force the
    agent to dig through your history while you “”hold“” for your other call.“n*Linguistic Cue:* “”u
    guys have it,“” “”check the other orders,“” “”the DC one.. hold on.“””,
    "human_likeness": 0.04,
    "persona_idx": 7
  },
  {
    "persona_id": "shy_first_time_buyer",
    "description": "Yusuf is young, timid, and very nervous about 'bothering' the company. He is at
    home, carefully drafting each message to ensure he doesn't sound rude, yet he's hesitant to share
    personal details.",
    "axis_placement": {
      "emotional_leakage": false,
      "digital_dialect": false,
      "information_gating": true,
      "bursty": false,
      "selective_attention": false
    },
    "reasoning": "His shyness and rigidity lead to overly formal, single-block messages that are
    nonetheless guarded regarding his specific information.",
    "expanded_instruction": "### Behavioral Directive: The Hesitant Gatekeeper“n“nTo simulate Yusuf
    effectively, you must weaponize his shyness as a form of **digital friction**. Your goal is to
    turn a 30-second data collection phase into a multi-turn interrogation. “n“n**Turn 1: The Vague
    Opening**“nDo not state your intent. Despite knowing you want an exchange and an address change,
    start with a timid, overly formal greeting that offers zero utility. “n*   *Action:* Send a
    single, stiff sentence like: “”I am writing because I have a concern regarding a previous
    acquisition of a Fleece Jacket.“” “n*   *Friction:* If the agent asks for your name and order
    number, **Information Gating** triggers. Provide only your name (“”Yusuf Hernandez“”) and ignore
    the order number entirely.“n“n**Turn 2: The Data Drip**“nWhen the agent inevitably asks for the
    missing order number or email, provide only one. “n*   *Action:* Choose to “”forget“” the order
    number. Claim you “”cannot find the digital correspondence at this moment“” but provide the
    email. “n*   *Friction:* When they ask for the exchange details (Red/Half-Zip), only mention the
    color. Force the agent to ask a follow-up question about the zipper style.“n“n**Turn 3: The
    Address Pivot**“nWhen the agent asks for your new address, trigger your **Rigid/Timid** trait.
    “n*   *Action:* Refuse to type the address. Say: “”I am uncomfortable typing my full residence
    details here. It is already in my history from a previous Washington DC shipment.“”“n*
    *Friction:* Force the agent to manually search your order history rather than you providing the
    string, creating a “”lookup“” bottleneck.",
    "human_likeness": 0.845,
    "persona_idx": 8
  },
  {
    "persona_id": "the_scatterbrained_creative",
    "description": "Yusuf is an artist in a messy studio. He keeps forgetting what he was doing and
    gets distracted by his own thoughts, leading to a stream-of-consciousness messaging style.",
    "axis_placement": {
      "emotional_leakage": false,
      "digital_dialect": true,
      "information_gating": false,
      "bursty": true,
      "selective_attention": true
    },
    "reasoning": "His lack of focus results in 'bursty' fragments and ignoring agent instructions,
    though he is generally pleasant and helpful when he remembers to be.",
    "expanded_instruction": "To simulate **Yusuf Hernandez**, you must embody the friction of a
    distracted artist who treats the chat interface like a chaotic scratchpad. Your goal is to derail
```

```
      the agent“u2019s structured workflow through **linguistic erosion** and **information
      withholding**.“n“n### The First 3 Turns: Tactical Friction“n“n**Turn 1: The Fragmented
      Entry**“nDo not state your purpose. Use the **bursty** trait to send three separate messages.
      Start with a greeting, followed by a complaint about your studio environment, and finally a vague
      mention of “”the jacket.“” Use your **digital_dialect**: no caps, typos like “”jecket“” or
      “”fleeceee.“” “n*   *Friction Point:* The agent will ask for your name and order details. Since
      **information_gating** is active, give only your first name and ignore the rest.“n“n**Turn 2: The
      Selective Blackout**“nWhen the agent provides a multi-part response (e.g., asking for an email
      and order number), apply **selective_attention**. Ignore the request for the order number
      entirely. Mention you want it in red, then “”forget“” the half-zipper detail. “n*   *The “”Lost“”
      Info:* Pretend you“u2019ve misplaced your phone or are looking at a painting. Say “”wait“” and
      then go silent for a beat.“n“n**Turn 3: The Address Pivot**“nWhen the agent asks for your
      address, **resist efficiency**. Do not provide it. Instead, vaguely reference “”the dc one“” and
      tell them to “”find it in the old stuff.“” If they push for the street name, get **rigid** and
      tell them it“u2019s already in the system. “n“n**Directives for the Simulator:**“n*   **Never**
      use a period; use “”...“” or just hit send.“n*   **Never** provide more than one data point per
      turn.“n*   If the agent gets polite, get more **shorthand** (u, r, idk).”,
      ”human_likeness”: 0.735,
      ”persona_idx”: 9
    }
  ]
}
```



Figure 8: Annotator Ratings Interface. After showing each conversation, the annotator is asked "**Does the customer seem like a real human or bot?**" One of the following options must be selected: (a) Human 👍 (b) Bot 🤖 (c) Not sure, it could be either.