SIMTRACE: GROUNDED MULTIMODAL USER TRAJECTORIES GENERATION FOR ONLINE USER MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Virtual clients offer a cost-effective approach to support applications such as A/B testing, recommender system development, and interface evaluation. However, building them requires access to large-scale, semantically faithful, fine-grained online user trajectories. These data are difficult to obtain because proprietary logs are subject to privacy restrictions and small businesses often lack sufficient traffic. Consequently, existing public datasets either abstract away fine-grained user interaction details or preserve rich context but remain platform-specific and small-scale. To address this gap, we propose SIMTRACE, a framework that generates faithful, fine-grained synthetic multimodal clickstreams through a computer-use client agent that is grounded in real user trajectories and the given web environment. SIMTRACE anonymizes real interactions and constructs a simulated twin of the given web environment, then uses both to generate synthetic interaction trajectories. Each action is paired with its corresponding web observations and user context, yielding a shareable alternative to confidential logs for developing computer-use agent-style virtual clients. We apply SIMTRACE to an e-commerce setting and evaluate both its fidelity and downstream utility. SIMTRACE outperforms competing baselines on 7 out of 8 fidelity metrics. Models trained on synthetic data achieve performance comparable to those trained on real data on downstream tasks such as purchase prediction and recommendation. For next action prediction task, augmenting real data with synthetic data further improves accuracy by 11.0% relative to training on real data alone. We release SIMTRACE as an open-source package to facilitate research on online user behavior modeling.

1 INTRODUCTION

LLMs are increasingly used to model online user behavior in web environments to support a broad range of applications, including A/B testing, recommender systems, usability studies, market research, and the evaluation of interactive LLM-based systems (Li et al., 2026; Lu et al., 2026a; Chen et al., 2025; Sun et al., 2025; Wang et al., 2026a). Many of these applications require models not only to predict final outcomes but also to reproduce the step-by-step interactions leading to those outcomes. Recording these interactions as trajectories allows researchers and practitioners to inspect how user decisions unfold. This provides an interpretable record of the decision-making process. Therefore, fine-grained trajectories that faithfully pair each action with its corresponding web observation are essential for developing computer-user virtual clients for these applications.

However, existing public datasets generally either abstract away fine-grained user interaction details or preserve rich context but remain limited in scale. Large-scale datasets (Requena et al., 2020; Ben-Shimon et al., 2015; Zhang et al., 2015), commonly used for recommendation and user behavior prediction, typically represent sessions as sequences of item identifiers or event types, omitting the interface observations and semantic context needed to train computer-use agents to reproduce users' interactions. Context-rich datasets (Wang et al., 2026b; Sun et al., 2025) collected through user studies, by contrast, preserve detailed interactions, but they are costly to construct, limited in scale, and typically confined to a single platform or environment, restricting their reuse in other settings. Organizations can instead collect interaction logs from their own environments, but this requires sufficient user traffic, and the resulting data are often proprietary and constrained by privacy regulations (Voigt & Bussche, 2017). Consequently, existing data sources rarely provide semantically faithful, environment-grounded trajectories at scale.

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.

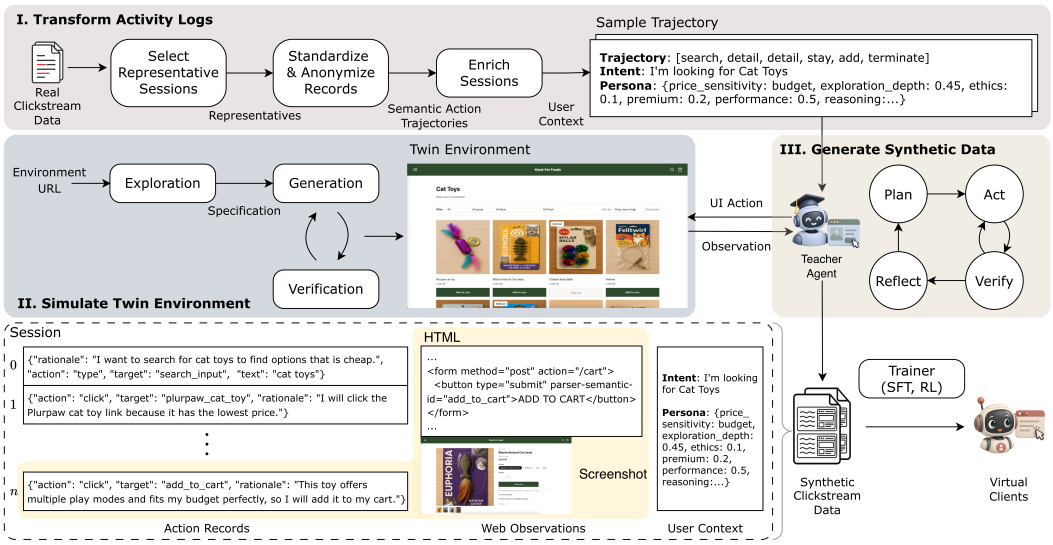

Figure 1: Overview of SIMTRACE and the resulting dataset. SIMTRACE transforms real activity logs into representative, anonymized behavior trajectories, constructs a twin sandbox environment of the given website, and uses LLM-based teacher agent to generate synthetic interaction sessions. Each resulting session contains user context and a sequence of timestamped steps, where each step pairs a structured action record with the corresponding webpage HTML and screenshot. See Appendix C.2 for a complete data example.

LLM-based user simulation provides a promising solution for generating context-rich trajectories at scale. Existing studies infer personas and intents from user activity logs or surveys and use persona-based LLM simulators to generate synthetic trajectories (Sun et al., 2025; Li et al., 2026). However, these methods typically prioritize plausible task completion. Their generated step-by-step actions still differ substantially from those observed in real user sessions, as reported by Zhang et al. (2026).

To address these limitations, we introduce SIMTRACE, a framework that uses a computer-use client agent to generate synthetic trajectories conditioned on real user trajectories and a given environment to improve the fidelity of step-by-step interactions in user modeling tasks. Through empirical evaluation, we demonstrate that synthetic data generated by SIMTRACE outperform competing baselines (Li et al., 2026; Lu et al., 2025) on 7 out of 8 fidelity metrics. Moreover, models trained on synthetic data perform comparably to those trained on real data across multiple downstream tasks. Augmenting real data with synthetic data further improves next action prediction accuracy by 11%. Together, these results show that SIMTRACE can alleviate data scarcity and privacy constraints without compromising behavioral fidelity or downstream utility. We release SIMTRACE as an open-source package with modules for data collection, synthetic data generation, and user-model training, together with a synthetic multimodal computer-use trajectory dataset spanning five e-commerce environments. The package enables researchers to construct synthetic datasets for additional environments, facilitating the growth of shared resources for user behavior research. [1]

2 SIMTRACE

We introduce SIMTRACE as a scalable framework for generating semantically faithful multimodal clickstreams to support user behavior modeling. As illustrated in Figure 1, SIMTRACE comprises three steps. First, it transforms noisy and private activity logs into abstract, anonymized behavioral trajectories (Section 2.1). However, these abstract trajectories alone lack the fine-grained web observations required for user behavior modeling. To recover this interaction context in a controllable environment, SIMTRACE constructs a simulated twin grounded in the given web environment using a website-agnostic approach (Section 2.2). Finally, a teacher agent uses the anonymized trajectories as behavioral guidance and navigates the twin environment to generate fine-grained interactions paired with corresponding web observations and user context (Section 2.3).

[1] The code is available at `https://anonymous.4open.science/r/SimTrace-D42C`.

2.1 TRANSFORM ACTIVITY LOGS

Activity logs record fine-grained interactions, including clicks, keystrokes, navigation paths, and time spent on each page (Bucklin & Sismeiro, 2009). For environments without an existing logging pipeline, our package integrates with commonly used platforms, including Microsoft Clarity (Microsoft Clarity) and PostHog (PostHog), to facilitate activity-log collection.

Select Representative Sessions. To obtain a compact yet representative set of user interactions, we select sessions that maximize the diversity of the given environment context and user behavior. In the e-commerce setting, the environment context is represented by the range of products encountered, while user behavior is captured by three conversion-funnel outcomes (Mittal, 2025): Browse-only, Cart Abandonment, and Checkout. These outcomes are defined as follows: *Browse-only* means that a user views products without adding any product to the cart or proceeding to checkout; *Cart Abandonment* means that a user adds at least one product to the cart but does not proceed to checkout; and *Checkout* means that a user proceeds to checkout.

Based on these representations, our objective is to select a subset of sessions that maximizes product coverage while maintaining a balanced distribution across the three outcomes. To solve this problem, we use a greedy algorithm as an efficient method that directly captures our selection criteria. The algorithm cycles through the three outcomes and selects the eligible session with the largest marginal coverage of previously unseen products. This iterative procedure terminates when it reaches the specified sample-size budget or a configured product-coverage threshold, such as 95%. Appendix A provides implementation details and the pseudocode.

Standardize and Anonymize Records. To provide a common interface between heterogeneous logging tools and downstream models, we standardize raw events using a semantic-action taxonomy. This standardization is necessary because logging tools differ in their event schemas and levels of granularity. Some tools record only high-level actions that trigger page transitions, whereas others capture low-level events such as individual mouse movements. These tool-specific and overly granular events can introduce noise and make activity logs difficult to compare across tools or use for behavior modeling. We therefore design a taxonomy that maps raw events to semantic actions in two categories: *domain-specific actions*, which capture user goals and task-state changes specific to a particular domain, and *domain-independent actions*, which capture navigation behavior that transfers across environments. For our e-commerce instantiation, we build on and extend the taxonomy introduced by Requena et al. (2020), following the mutually exclusive and collectively exhaustive (MECE) principle (Minto, Barbara, 1987). Table 1 presents the resulting taxonomy.

After standardization, we further anonymize user, store, session, and product identifiers using keyed HMAC-SHA-256 hashes (National Institute of Standards and Technology, 2008). We also abstract product information into a product category and a price bucket. Each transformed record contains `store_id`, `session_id`, `user_id`, `timestamp`, `semantic_action`, `product_hash`, `product_category`, and `price_bucket`.

Enrich Sessions. Standardization and anonymization operate at the record level, but session-wide patterns can reveal latent user intent and preferences valuable for behavior modeling. We represent this session-level user context through **intent** and **persona** inferred from interaction history. For our e-commerce instantiation, we construct the persona following SimGym (Li et al., 2026). The persona comprises five dimensions: *price sensitivity*, reflecting preferences across budget, mid-range, and premium products; *exploration depth*, reflecting the extent of search and browsing; and *premium focus*, *performance focus*, and *ethics focus*, capturing attention to luxury, durability, and ethical sourcing, respectively. See Appendix C.1. The final transformed log represents each session through its semantic-action trajectory, inferred intent, and persona, as illustrated in Figure 1.

2.2 SIMULATE TWIN ENVIRONMENT

To provide web observations of corresponding actions, fine-grained trajectories need to be generated through interactions with an environment. To support controllable generation when direct interaction with the original environment is infeasible, such as when sensitive content must be masked, we provide a website-agnostic approach for constructing a simulated sandbox of a given environment.

Inspired by ShopGym (Savadikar et al., 2026), we autonomously construct the simulated sandbox through three modules: **exploration**, **generation**, and **verification**. During exploration, a power-

Table 1: Taxonomy for transforming raw clickstream events in e-commerce domain.

Category	Semantic Action	Definition
Domain-specific Actions	`search`	The user enters a query in the search bar.
	`detail`	The user visits a product page or interacts with product attributes.
	`add`	The user adds a product to the cart.
	`remove`	The user removes a product from the cart.
	`checkout`	The user proceeds to purchase using *Buy Now* or *Checkout*.
Domain-independent Actions	`goto`	The user navigates to a new non-product-detail page, e.g. applying a filter, changing the sort order, or following a navigation link.
	`back`	The user returns to the previous page.
	`stay`	The user remains on the same page while a new event is recorded, typically because of a same-page interaction, such as scrolling.
	`terminate`	The label is appended as the final action of every session.

ful coding agent (e.g. `Pi` harness (Earendil Inc.& Contributors) with `GPT-5.5`) follows the task lists we designed to inspect the original environment and collect browser evidence, including page screenshots. These observations are consolidated into specification capturing the visual theme, navigation structure, and supported interactions. The generation module, which is driven by the same agent, then transforms this specification into source code for a runnable sandbox while replacing sensitive content with synthetic alternatives. See Appendix B for implementation and cost details. To ensure that the simulated environment preserves similar user experience, including user-observed content and user actions, we introduce two verifiers: a *visual-fidelity verifier* and an *action-replay verifier*. We modify the simulated environment iteratively until all the verifier feedback is resolved.

The *visual-fidelity verifier* compares screenshots of the generated sandbox against reference screenshots collected during exploration. An LLM-based evaluator scores each screenshot from 0 to 10 along five dimensions: layout, color theme, component correspondence, content density, and language consistency. The sandbox passes visual verification only if the average score across these dimensions meets a configured threshold. Otherwise, the verifier returns concrete discrepancies and repair instructions for the next code-generation iteration.

The *action-replay verifier* evaluates whether observed user trajectories remain executable in the simulated environment. It samples trajectories from real clickstreams and replays them in the simulated environment using Playwright browser automation. For example, a `detail` action passes if the associated product path returns a non-error response and renders a product page, whereas an `add` action passes if the verifier can successfully activate an add-to-cart control. A trajectory passes only if every action is reproduced successfully. Each failure report identifies the corresponding trajectory, action, and reason and is supplied to the next code-generation iteration.

2.3 Generate Synthetic Data

To reproduce fine-grained multimodal interaction trajectories based on the transformed sessions, we use a LLM-based teacher agent to navigate the simulated environment. This teacher agent needs to plan step-by-step actions while maintaining consistency with the user's high-level intent and persona. It also needs to ensure that each action is valid and realistic before execution, since an incorrect action can irreversibly alter the environment state and cause the trajectory to deviate from the reference behavior. We therefore develop a nested reflection–verification architecture: the outer loop maintains task-level coherence across the trajectory, while the inner loop validates each proposed action before execution. See Figure 1 Section III.

To maintain coherence over multiple interaction steps, the outer loop repeatedly plans, acts, and reflects. The **plan module** determines the next step using the current web observation and guidance from prior reflections. The **act module** translates this plan into a browser action with a corresponding rationale. After execution, the **reflect module** examines the resulting observation and provides high-level guidance for subsequent planning.

To prevent an incorrect action from distorting the remaining trajectory, the **verify module** in the inner loop evaluates each proposal against three criteria: (1) executability, whether the action can be performed on the current page; (2) trajectory consistency, whether the action aligns with the reference trajectory; and (3) realism, whether a real user would plausibly take the action in the current context. For each criterion, the verifier returns a rationale, score, and confidence value. If an action fails verification, the action module receives the failure reason and generates a revised action.

Table 2: Comparison of SIMTRACE with existing datasets. *Multi-platform* indicates whether a dataset spans multiple environment; *Sandbox* indicates whether it is paired with controllable environments for interaction.

Dataset	Size	Tasks	Action Space	Multi-modal	Multi-platform	Sandbox
Coveo	203k	Purchase Prediction	6 Semantic Actions	✗	✗	✗
YOOCHOOSE	9M	Recommendation	Product Identifiers	✗	✗	✗
Tmall	8M	Recommendation	Product Identifiers	✗	✗	✗
SHOPCART	31,865	User Behavior Simulation	click, type, terminate	✗	✗	✗
OPeRA	692	All Above	Rich GUI Actions	✓	✗	✗
SIMTRACE	1,513	All Above	Rich GUI Actions	✓	✓	✓

Table 3: Action distribution for SIMTRACE.

Action	Count (%)
Click	8,195 (79.40)
Terminate	901 (8.73)
Scroll	551 (5.34)
Type	295 (2.86)
Back	277 (2.68)
Select	94 (0.91)
Clear	8 (0.08)
Total	10,321

3 E-COMMERCE SYNTHETIC DATASET

Using SIMTRACE, we generate an e-commerce synthetic dataset across five simulated stores covering apparel, food, cookware, toys, and accessories, and four languages: English, French, Turkish, and Hindi.[2] Following OPeRA (Wang et al., 2026b), each session data contains three components: action records, web observations, and user context.

Action Records. Each session contains an ordered sequence of actions, each with an action type, a natural-language rationale, and action-specific metadata. Table 3 shows the action type distribution. Each action type defines its own metadata fields. For example, a `click` action includes a `target` field that identifies the selected interface element in the web observation, such as `"search_icon"`, whereas a `type` action includes an `input` field specifying the text entered by the agent.

Web Observations. At each step, the dataset records a web observation comprising an HTML representation of the current page and a screenshot. Together, they provide complementary structural and visual context for interpreting the action and its rationale. Following common practice in prior work (Lu et al., 2025; Sun et al., 2025; Wang et al., 2026b), we simplify the raw HTML into a semantic representation that preserves meaningful interface elements.

User Context. Each session is paired with an inferred persona and intent derived from the raw activity logs following SimGym (Li et al., 2026). The persona summarizes the user's behavior and preferences, while the intent specifies the session-level goal. See Appendix C.1 for an example.

Table 2 compares our dataset with existing public datasets for modeling online user behavior. Large recommendation-oriented datasets (Requena et al., 2020; Ben-Shimon et al., 2015; Zhang et al., 2015) provide substantial scale but primarily record product-level interactions with limited context. Another line of datasets record real user interactions Lu et al. (2026b); Wang et al. (2026b). They preserve finer-grained user interactions but are typically tied to a specific data-collection platform and do not provide an interactive environment. In contrast, our dataset combines fine-grained interaction context with controllable sandbox environments across multiple platforms, and can be further expanded or customized to new environments. Appendix C.2 provides a complete example session.

4 EXPERIMENTS

We evaluate SIMTRACE along two complementary perspectives: **fidelity** and **downstream task utility**. Fidelity assesses how closely the synthetic data reproduce the characteristics of real data, whereas downstream utility assesses whether they provide effective supervision for user behavior modeling. Specifically, our experiments address the following research questions:

- **RQ1 - Fidelity**: How faithfully does SIMTRACE reproduce real user behavior compared with existing methods?
- **RQ2 - Downstream Task Utility**: Can synthetic data generated by SIMTRACE effectively support downstream user behavior modeling tasks?
- **RQ3 - Low-resource Setting**: How effectively can SIMTRACE support user behavior modeling in low-resource settings, such as next-action prediction?

[2]The dataset is available at `https://anonymous-hf.com/a/owfgiuyo9my1/`.

Table 4: Fidelity comparison between different generation methods on `gemini-3-flash` model. The best results are shown in **bold**. Statistically significant improvements ($p < 0.05$) over all other baselines are marked with *.

	Outcome-level	Sequence-level			Semantic-level			
Method	**Outcome JSD ↓** $[0,1]$	**Action Freq. JSD ↓** $[0,1]$	**Transition Matrix L_1 Distance ↓** $[0,1]$	**Trajectory Levenshtein ↓** $[0,1]$	**Product Coherence Gap ↓** $[0,1]$	**Product Diversity Ratio ↓** $[0,\infty)$	**Product Alignment ↑** $[-1,1]$	**Rationale Alignment ↑** $[-1,1]$
Prompting-based methods								
Persona	2.94×10^{-1}	5.40×10^{-2}	4.30×10^{-1}	6.74×10^{-1}	3.92×10^{-2}	2.84×10^{-1}	5.75×10^{-1}	6.10×10^{-1}
Trajectory	8.56×10^{-3}	1.10×10^{-2}	2.34×10^{-1}	2.18×10^{-1}	6.86×10^{-2}	$\mathbf{7.74\times10^{-2}}$	5.32×10^{-1}	6.52×10^{-1}
Trajectory + Persona	3.36×10^{-3}	9.73×10^{-3}	2.39×10^{-1}	2.85×10^{-1}	6.85×10^{-2}	9.15×10^{-2}	5.43×10^{-1}	6.54×10^{-1}
Agentic methods								
UXAgent	1.86×10^{-3}	9.58×10^{-3}	2.30×10^{-1}	3.50×10^{-1}	3.81×10^{-2}	9.68×10^{-2}	5.52×10^{-1}	6.59×10^{-1}
SIMTRACE	$\mathbf{7.98\times10^{-5*}}$	$\mathbf{4.96\times10^{-3*}}$	$\mathbf{2.27\times10^{-1}}$	$\mathbf{1.80\times10^{-1}}$	$\mathbf{3.75\times10^{-2}}$	8.50×10^{-2}	$\mathbf{5.92\times10^{-1}}$	$\mathbf{6.73\times10^{-1}}$

4.1 FIDELITY (RQ1)

Experimental setup. To evaluate the fidelity of SIMTRACE, we compare it with prompting and agentic baselines. *Persona* uses the persona representation from SimGym (Li et al., 2026); *Trajectory* uses a reference trajectory as a one-shot demonstration; and *Trajectory + Persona* combines both. We use UXAgent (Lu et al., 2025) as the agentic baseline. Its planning–reflection loop conceptually corresponds to the outer loop in SIMTRACE, providing a practical comparison for the added benefit of action-level verification. For each method, we generate 200 sessions on both `gemini-3-flash` (Google, 2025) and `gpt-5.6-sol` (OpenAI, 2026) models.

Evaluation metrics. Behavioral fidelity is multifaceted and cannot be adequately characterized by a single metric. We therefore evaluate it at three complementary levels. Outcome-level fidelity uses Jensen–Shannon divergence (JSD) to assess whether synthetic data preserve the distribution of three session outcomes: browsing only, cart abandonment, and checkout. Sequence-level fidelity captures finer-grained trajectory similarity using action-frequency JSD, action transition matrix distance, and trajectory-level Levenshtein distance. In addition to these structural measures, semantic-level fidelity assesses whether generated trajectories reflect user preference similar to that observed in real sessions. Specifically, it evaluates product coherence, diversity, alignment and rationale alignment. Together, these metrics capture population-level statistical similarity and session-level behavioral plausibility. Appendix D provides detailed definitions of each metric.

Analysis. Table 4 shows that SIMTRACE achieves the best fidelity on 7 out of 8 metrics, with consistent gains across all three evaluation levels. We also test with the `gpt-5.6-sol` model, it largely preserves the method ranking with a slight absolute gain. See Table 6 for results.

Beyond the quantitative results, our qualitative analysis reveals two practical issues when generating synthetic trajectories from activity logs. The first issue, which we refer to as **granularity mismatch**, occurs when activity logs omit the interface-level steps required to reproduce an event. For example, a `search` event may record only the query, whereas an agent must locate the search field, enter the query, and submit it through multiple actions. The prompt-based *Trajectory* method often advances to the next logged event before completing these intermediate steps, producing discontinuous trajectories. Our outer reflection–planning loop addresses this issue by tracking task progress and expanding each coarse event into the required actions while treating the reference trajectory as high-level guidance. The second issue, which we refer to as **interface drift**, occurs when actions recorded in an earlier environment are no longer directly executable in the current interface and require intermediate actions to reach the intended state. Without action-level verification, an agent may repeatedly attempt infeasible actions until exhausting its action budget. Our inner verifier instead checks proposed actions before execution, identifies appropriate intermediate steps, and redirects local deviations toward the reference behavior. Together, these mechanisms allow SIMTRACE to use logged trajectories as flexible behavioral guidance rather than brittle execution scripts.

4.2 DOWNSTREAM TASK UTILITY (RQ2)

To assess whether the synthetic sessions generated by our pipeline provide useful training signals across multiple downstream tasks. We evaluate SIMTRACE on two tasks: purchase prediction and session-based recommendation. For each task, we compare models trained and tested on real sessions (TRTR) with models trained on synthetic sessions and tested on real sessions (TSTR). Both conditions use the same held-out real test set, isolating the effect of replacing real training trajectories with their synthetic counterparts.

Table 5: Downstream performance of models trained on real (TRTR) and synthetic (TSTR) sessions and evaluated on the same held-out real test set. Δ is TSTR-TRTR. All 95% CIs for Δ include zero, indicating no statistically significant differences between two training conditions.

Method	Metric	TRTR	TSTR	Δ	95% CI
Purchase prediction					
SFT (Qwen3.5-4B)	F1 ↑	64.36	65.52	+1.16	[−3.00, +5.00]
Session-based recommendation					
RAIN (Graph-based)	MRR@5 ↑	12.47	12.58	+0.11	[−1.59, +1.70]
	HR@5 ↑	21.25	20.85	−0.40	[−3.20, +2.27]
	MRR@10 ↑	13.20	13.64	+0.44	[−1.24, +2.03]
	HR@10 ↑	26.71	29.12	+2.41	[−0.71, +5.40]
DIMO (Context-aware)	MRR@5 ↑	17.84	16.49	−1.35	[−3.67, +1.16]
	HR@5 ↑	31.83	30.27	−1.56	[−5.05, +2.28]
	MRR@10 ↑	18.91	18.34	−0.57	[−2.85, +2.41]
	HR@10 ↑	39.86	40.46	+0.60	[−2.83, +3.95]

Purchase Prediction Given the interaction history before a session's terminal event, the model predicts whether the session ends in a purchase. We evaluate this binary classification task using the F1 score. For training, we use all available matched real–synthetic session pairs and balance the purchase and non-purchase classes, yielding 739 examples in each training condition. To construct the held-out test set, following prior work (Requena et al., 2020), we exclude real sessions with length $L < 5$ to ensure sufficient interaction history. We then randomly sample 500 of the remaining sessions to form the final held-out test set.

We finetune `Qwen3.5-4B` (Qwen Team, 2026) with QLoRA for five epochs with a learning rate of 1×10^{-5}. We report preliminary results averaged over three random seeds, together with the TSTR–TRTR difference and its 95% confidence interval.

Session-Based Recommendation Let $\mathcal{V}$ denote the item set, and let $S = (v_1, v_2, \ldots, v_L)$ denote a sequence of item interactions within a user session. Given an observed session S, session-based recommendation aims to predict the next item $v_{L+1} \in \mathcal{V}$ by ranking all candidate items and returning the top-K items. Following prior work (Zhang et al., 2023), we remove sessions containing only one interaction and items appearing fewer than five times. Applying these filters to the matched real and synthetic data yields 357 training sessions covering 110 items. Since ID-based recommendation models cannot score items unseen during training, we restrict the held-out real test set to the shared training-item vocabulary. The resulting test set contains 434 sessions involving 40 items.

We apply it to two complementary recommendation methods. RAIN (Zeng et al., 2025) is a recent graph-based session recommendation method with publicly available code. DIMO (Zhang et al., 2024) is a context-aware method that additionally incorporates product titles, vendors, product types, and images into its recommendation model. DIMO therefore allows us to assess whether the rich multimodal content generated by our pipeline provides additional utility for downstream recommendation. We evaluate both models using Hit Rate at K (HR@K) and Mean Reciprocal Rank at K (MRR@K), where $K \in \{5, 10\}$. We repeat each experiment with three random seeds and use bootstrap resampling to estimate the TSTR–TRTR differences and 95% confidence intervals.

Findings Table 5 reports downstream performance for the models trained on real and synthetic sessions using the same held-out real test set. It shows that every 95% confidence interval includes zero, indicating no statistically significant difference between the two training conditions under our experimental setting. These results suggest that the synthetic sessions preserve task-relevant training signals for both purchase prediction and session-based recommendation.

4.3 Low-Resource Setting (RQ3)

For small businesses, limited traffic hinders the training of user simulation models that can perform fine-grained navigation behaviors and enable practitioners to analyze how users interact with a website step by step. Therefore, we assess how effectively synthetic data generated by SimTrace supports next-action prediction in low-resource settings. Specifically, we sample 50 sessions from OPeRA (Wang et al., 2026b) as the real training data and compare real-only training (Real-only) with synthetic-only (Synth-only) and synthetic augmentation (Real + Synth) training. All these trainings use `Qwen3.5-9B` (Qwen Team, 2026) as the base model and are evaluated on the held-out OPeRA

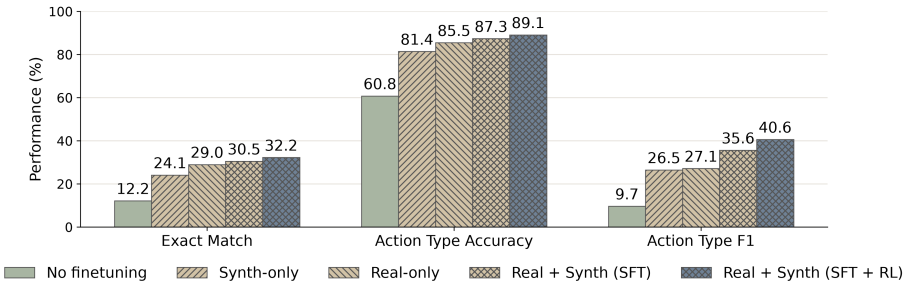

Figure 2: Next action prediction performance of `Qwen3.5-9B` on the OPeRA test set. Augmenting real with synthetic data improves all metrics, with RL achieving the best overall performance.

test set using exact match, action-type F1, and action-type accuracy. A prediction counts as an exact match only when both its `action` and `target` fields exactly match the reference.

SFT Training and Analysis. We formulate each session as a multi-turn conversation, where each user turn contains the current web observation and each assistant turn contains a rationale followed by a structured action. The system prompt specifies the task instruction, persona, and shopping intent. See Appendix E for the complete template and Appendix F.1 for training details.

Figure 2 shows that SFT on synthetic data improves the base model without finetuning by 11.9–20.6 percentage points across the three metrics. Training on the 50 real sessions yields stronger performance, likely because of distributional differences between synthetic and real supervision, as analyzed in Appendix F.2. Despite this gap, synthetic data provides complementary supervision: jointly training on real and synthetic sessions produces further gains of 1.5–8.5 percentage points over real-only. The improvement mainly comes from broader `target` field coverage, which reduces overprediction of frequent values and helps the model recognize infrequent or previously unseen interface elements. See Appendix F.3 for details.

However, `target` prediction remains the main bottleneck for exact-match performance. Target field values are long and highly sparse: 77.4% of unique targets occur only once in OPeRA, and 14.9% contain at least 200 characters. Nevertheless, their hierarchical structure encodes behavioral similarity. For example, `"customers_also_bought.<product1>"` and `"customers_also_bought.<product2>"` both indicate exploration of recommended alternatives, whereas a target beginning with `"review"` indicates further investigation of the current product. Exact match treats these predictions as equally different, despite the first two reflecting similar behavioral directions. We therefore introduce an RL objective that assigns partial credit according to hierarchical similarity of `target` field.

RL Training and Analysis. Following the approach of (DeepSeek-AI et al., 2025), we initialize the training with SFT on synthetic data to learn the dependencies among web context, action type, and target. We then apply RL on the real sessions to align the learned policy with the behavioral patterns and target distribution of the real environment. Each real session is converted into $\langle \text{context}, \text{action} \rangle$ pairs, where the context contains all preceding web observations and executed actions and the prediction is the structured next action.

Based on our observations from the SFT results, we design a reward function for the `target` field that assigns partial credit based on the prediction. The reward comprises three terms: a string-similarity reward, a binary validity reward indicating whether the predicted target exists in the current UI context, and an exact-match bonus. Let $\hat{t}$ and t denote the predicted and reference targets:

$$R_{\text{target}} = w_{sub} s_{\text{sub}}(\hat{t}, t) + w_{valid} \mathbb{I}[\hat{t} \in \mathcal{T}(x_t)] + w_{exact} \mathbb{I}[\hat{t} = t],$$

where $s_{\text{sub}}(\hat{t}, t)$ is the symmetric longest-common-substring ratio between predicted and ground-truth target, and $\mathcal{T}(x_t)$ is the set of semantic identifiers in the current web observation.

Target accuracy alone, however, does not ensure the prediction accuracy. Following Zhang et al. (2026), the full reward therefore combines R_{target} with R_{action}, which rewards correct action-type predictions, and R_{format}, which rewards valid structured responses. We optimize this multi-reward objective using Group Reward-Decoupled Normalization Policy Optimization (Liu et al., 2026); implementation details are provided in Appendix F.1. Relative to mixed-data SFT, the RL stage further

improves the three metrics by 1.7%, 1.8%, and 5.0%, respectively. The improvements mainly come from better output validity and target selection. RL eliminates malformed JSON predictions and more often redirects predictions from an unrelated UI component to the correct hierarchical target path. For example, it corrects `reviews.popover.review_images.next` to the ground-truth `buybox.purchase_form.add_to_cart`. It also improves predictions when the correct parent path is identified, but the final target segment is wrong. See Appendix F.4 for details.

5 Related Work

LLM-Based User Simulation Recent LLM-based user simulation can be categorized into two paradigms: prompting LLMs and training dedicated user models. Among prompt-based methods, a large amount of work focuses on steering LLM behavior through personas. PAARS (Mansour et al., 2025) induces shopper personas from anonymized sessions, while SimGym (Li et al., 2026) constructs buyer profiles for offline A/B testing. Other work enriches simulators with domain knowledge. SAGE grounds conversations in business principles (Shea et al., 2025), whereas VISTA enables interaction through both UI and API actions (Lu et al., 2026a). Although flexible, prompting-based methods rely on powerful LLMs, making long multimodal simulations costly. Dedicated user models reduce this cost by learning human behavior into model parameters. UserLM (Naous et al., 2025) and Socratic (Kong et al., 2024) learn from human-authored turns to reproduce realistic questions, while Shop-R1 (Zhang et al., 2026) and Customer-R1 (Wang et al., 2025) fine-tune models to create virtual shoppers for e-commerce domain. However, these models remain tied to their training domains and interaction protocols. In contrast, SimTrace generates environment-specific supervision for training efficient user models tailored to the given web environment.

Datasets for User Behavior Modeling User behavior modeling requires data that captures both user actions and the context in which they occur. Existing datasets generally fall into two categories. Large-scale datasets contain millions of shopping or recommendation sessions but typically represent them as sequences of item identifiers or coarse event types, omitting the interface observations and semantic context needed to reconstruct user decisions (Requena et al., 2020; Ben-Shimon et al., 2015; Zhang et al., 2015). Context-rich datasets preserve semantic actions, webpage content, but are costly to collect, limited in scale, and usually tied to a single platform (Wang et al., 2026b; Sun et al., 2025). SimTrace bridges this gap by transforming private logs into standardized, anonymized trajectories and generating scalable multimodal clickstreams of the given environment.

6 Conclusion

We introduce SimTrace, a scalable framework for generating semantically faithful multimodal computer-use client-agent trajectories for user modeling. SimTrace transforms real activity logs into standardized and anonymized trajectories, constructs a simulated twin environment grounded on the given web, and uses a teacher agent to generate behaviorally grounded interactions. Our evaluation shows that SimTrace reproduces real user behavior more faithfully than the baselines. Besides, models trained on synthetic data perform comparably to those trained on real data for multiple downstream tasks. Augmenting real data with synthetic data further improves next-action prediction accuracy by 11.0%. These results demonstrate the potential of SimTrace to alleviate real-data scarcity and reduce reliance on sensitive user logs when training downstream models.

7 Limitations

The proposed data generation method is designed to be domain-adaptive, while our empirical evaluation focuses on an e-commerce scenario. Further validation is needed in other domains, where user semantic action taxonomy, privacy requirements may differ. Future work should evaluate SimTrace in domains such as finance, education, and healthcare. Secondly, SimTrace does not fully solve the cold-start problem, as trajectory-conditioned generation still requires initial activity logs. Although synthetic data generated from a similar environment can provide useful supervision when target-domain data are scarce, our results show that such data remain less effective than real target-domain interactions for next-action prediction. Future work should investigate cross-environment schema alignment and minimal-data adaptation to bootstrap generation from related environments while reducing reliance on target-domain logs.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, designing or providing feedback on research methodology or experiments, implementing methods, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, or interpreting results. While, formulate mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, assist with translation are not applicable to this work. Additionally, we used generative AI tools for editing software code, editing a research paper to improve readability, sourcing/searching for information, creation of artifacts. We have reviewed all AI-assisted work. In particular, we verified that AI-edited text preserved the authors' intended meaning; LLM-generated code was verified and tested for correctness by 4 authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses real user activity logs as behavioral guidance for generating synthetic clickstreams. The source logs were collected and processed under user who explicated opted in for data-use agreement. We use the logs only for the purposes authorized by the data provider and do not release the original activity logs.

To reduce privacy and confidentiality risks, our pipeline replaces identifiers associated with users, sessions, stores, products, and URLs with keyed HMAC-SHA-256 hashes and abstracts product information into categories and price buckets. Our public release only contains the synthetic clickstream dataset, excluding the original logs, hashing keys, original identifiers and URLs, proprietary website assets, and any other information that could directly identify users or businesses. We document the dataset composition, provenance, processing procedure, intended uses, and licensing conditions in a dataset card.

These safeguards reduce direct disclosure risks but do not constitute a formal privacy guarantee. Inferred personas and intents are model-generated behavioral summaries, therefore they may reflect biases in the source logs or the generative models. The released dataset should therefore not be used to infer sensitive characteristics, make consequential decisions about individuals, or facilitate deceptive or manipulative interactions. We release datasets and resources for research only purpose.

REPRODUCIBILITY STATEMENT

We provide the methodological and experimental details required to reproduce the publicly reproducible components of this work. Sections 2.1–2.3 describe the activity-log transformation, simulated-environment construction, and synthetic-session generation procedures, while Algorithm 1 specifies the session-selection algorithm. The appendix provides complete definitions of the eight fidelity metrics and the user-model prompt template for reproduce training and evaluation. The experimental sections 4 report the model configurations, training procedures, evaluation metrics, data splits, etc.

Upon publication, we will release the open-source code for reproducing our method, experiments and step-by-step documentations for reproducing the reported downstream experiments.

REFERENCES

David Ben-Shimon, Alexander Tsikinovsky, Michael Friedmann, Bracha Shapira, Lior Rokach, and Johannes Hoerle. RecSys Challenge 2015 and the YOOCHOOSE Dataset. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pp. 357–358, New York, NY, USA, September 2015. Association for Computing Machinery. ISBN 978-1-4503-3692-5. doi: 10.1145/2792838.2798723. URL https://dl.acm.org/doi/10.1145/2792838.2798723.

Randolph E. Bucklin and Catarina Sismeiro. Click here for internet insight: Advances in clickstream data analysis in marketing. *Journal of Interactive Marketing*, 23(1):35–48, 2009. doi: 10.1016/j.intmar.2008.10.004.

Luyu Chen, Quanyu Dai, Zeyu Zhang, Xueyang Feng, Mingyu Zhang, Pengcheng Tang, Xu Chen, Yue Zhu, and Zhenhua Dong. RecUserSim: A Realistic and Diverse User Simulator for Evaluating Conversational Recommender Systems. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 133–142, May 2025. doi: 10.1145/3701716.3715258. URL http://arxiv.org/abs/2507.22897. arXiv:2507.22897 [cs].

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *Nature*, 645(8081):633–638, September 2025. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://arxiv.org/abs/2501.12948. arXiv:2501.12948 [cs.CL].

Earendil Inc.& Contributors. Pi. https://pi.dev/. Accessed: 2026-07-25.

Google. Gemini 3 flash, 2025. URL https://deepmind.google/models/gemini/flash/. Accessed: August 28, 2026.

Chuyi Kong, Yaxin Fan, Xiang Wan, Feng Jiang, and Benyou Wang. PlatoLM: Teaching LLMs in Multi-Round Dialogue via a User Simulator. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7841–7863, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.424. URL https://aclanthology.org/2024.acl-long.424/.

Han Li, Vibhor Malik, Zahra Zanjani Foumani, Alberto Castelo, Shuang Xie, Ailin Fan, Keat Yang Koay, Yuanzheng Zhu, Meysam Feghhi, Ronie Uliana, Zhaoyu Zhang, Angelo Ocana Martins, Mingyu Zhao, Francis Pelland, Jonathan Faerman, Nikolas LeBlanc, Aaron Glazer, Andrew McNamara, Zhong Wu, and Lingyun Wang. SimGym: A Framework for A/B Test Simulation in E-Commerce with Traffic-Grounded VLM Agents, 2026. URL https://arxiv.org/abs/2605.19219. Version Number: 1.

Jianhua Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. doi: 10.1109/18.61115.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. GDPO: Group reward-Decoupled Normalization Policy Optimization for Multi-reward RL Optimization, January 2026. URL http://arxiv.org/abs/2601.05242. arXiv:2601.05242 [cs.CL].

Yunan Lu, Ryan Shea, Yusen Zhang, and Zhou Yu. VISTA: A Versatile Interactive User Simulation Toolkit for Agent Evaluation, June 2026a. URL http://arxiv.org/abs/2606.11079. arXiv:2606.11079 [cs.CL].

Yuxuan Lu, Bingsheng Yao, Hansu Gu, Jing Huang, Jessie Wang, Yang Li, Jiri Gesi, Qi He, Toby Jia-Jun Li, and Dakuo Wang. UXAgent: An LLM agent-based usability testing framework for web design. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, 2025. doi: 10.1145/3706599.3719729.

Yuxuan Lu, Jing Huang, Yan Han, Bingsheng Yao, Sisong Bei, Jiri Gesi, Yaochen Xie, Yisi Sang, Zheshen, Wang, Qi He, and Dakuo Wang. Can LLM Agents Simulate Multi-Turn Human Behavior? Evidence from Real Online Customer Behavior Data, April 2026b. URL http://arxiv.org/abs/2503.20749. arXiv:2503.20749 [cs.CL].

Saab Mansour, Leonardo Perelli, Lorenzo Mainetti, George Davidson, and Stefano D'Amato. PAARS: Persona Aligned Agentic Retail Shoppers, March 2025. URL http://arxiv.org/abs/2503.24228. arXiv:2503.24228 [cs].

Microsoft Clarity. Microsoft clarity. https://clarity.microsoft.com/. Accessed: 2026-08-16.

Minto, Barbara. The minto pyramid principle: Logic in writing, thinking, and problem solving, 1987. Introduced the MECE (Mutually Exclusive, Collectively Exhaustive) framework.

Banwari Mittal. The psychology of online shopping cart abandonment: a scrutiny of the current research framework and building an improved model of the online shopper journey. *Electronic Commerce Research*, 25(2):777–803, April 2025. doi: 10.1007/s10660-022-09667-0. URL https://ideas.repec.org/a/spr/elcore/v25y2025i2d10.1007_s10660-022-09667-0.html.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the Dialogue: Training and Evaluating User Language Models, October 2025. URL http://arxiv.org/abs/2510.06552. arXiv:2510.06552 [cs].

National Institute of Standards and Technology. The keyed-hash message authentication code (hmac). Technical Report FIPS PUB 198-1, U.S. Department of Commerce, 2008.

OpenAI. Openai, 2026. URL https://developers.openai.com/api/docs/models/gpt-5.6-sol. Accessed: August 28, 2026.

PostHog. Posthog. https://posthog.com/. Accessed: 2026-08-16.

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

Borja Requena, Giovanni Cassani, Jacopo Tagliabue, Ciro Greco, and Lucas Lacasa. Shopper intent prediction from clickstream e-commerce data with minimal browsing information. *Scientific Reports*, 10(1):16983, October 2020. ISSN 2045-2322. doi: 10.1038/s41598-020-73622-y. URL https://www.nature.com/articles/s41598-020-73622-y.

Chinmay Savadikar, Mingyu Zhao, Yuanzheng Zhu, Han Li, Shuang Xie, Alberto Castelo, Tianfu Wu, and Lingyun Wang. ShopGym: An Integrated Framework for Realistic Simulation and Scalable Benchmarking of E-Commerce Web Agents, May 2026. URL http://arxiv.org/abs/2605.16116. arXiv:2605.16116 [cs.AI].

Ryan Shea, Yunan Lu, Liang Qiu, and Zhou Yu. SAGE: A Top-Down Bottom-Up Knowledge-Grounded User Simulator for Multi-turn AGent Evaluation, October 2025. URL http://arxiv.org/abs/2510.11997. arXiv:2510.11997 [cs].

Lu Sun, Shihan Fu, Bingsheng Yao, Yuxuan Lu, Wenbo Li, Hansu Gu, Jiri Gesi, Jing Huang, Chen Luo, and Dakuo Wang. LLM Agent Meets Agentic AI: Can LLM Agents Simulate Customers to Evaluate Agentic-AI-based Shopping Assistants?, September 2025. URL http://arxiv.org/abs/2509.21501. arXiv:2509.21501 [cs] version: 1.

Paul Voigt and Axel von dem Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer Publishing Company, Incorporated, 1st edition, July 2017. ISBN 978-3-319-57958-0.

Mengxin Wang, Dennis J. Zhang, and Heng Zhang. Large Language Models for Market Research: A Data-augmentation Approach, April 2026a. URL http://arxiv.org/abs/2412.19363. arXiv:2412.19363 [cs.AI].

Ziyi Wang, Yuxuan Lu, Yimeng Zhang, Jing Huang, and Dakuo Wang. Customer-R1: Personalized Simulation of Human Behaviors via RL-based LLM Agent in Online Shopping, October 2025. URL http://arxiv.org/abs/2510.07230. arXiv:2510.07230 [cs.CL].

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, Yu Su, Upol Ehsan, Malihe Alikhani, Toby Jia-Jun Li, Lydia Chilton, and Dakuo Wang. OPeRA: A Dataset of Observation, Persona, Rationale, and Action for Evaluating LLMs on Human Online Shopping Behavior Simulation, April 2026b. URL http://arxiv.org/abs/2506.05606. arXiv:2506.05606 [cs].

Yu Xia, Chi-Hua Wang, Joshua Mabry, and Guang Cheng. Advancing Retail Data Science: Comprehensive Evaluation of Synthetic Data, June 2024. URL http://arxiv.org/abs/2406.13130. arXiv:2406.13130 [cs.LG].

Xinyi Zeng, Shuchao Li, Zequn Zhang, Li Jin, Zhi Guo, and Kaiwen Wei. Rain: Reconstructed-aware in-context enhancement with graph denoising for session-based recommendation. *Neural Networks*, 184:107056, 2025.

Xiaokun Zhang, Bo Xu, Fenglong Ma, Chenliang Li, Liang Yang, and Hongfei Lin. Beyond Co-occurrence: Multi-modal Session-based Recommendation, September 2023. URL http://arxiv.org/abs/2309.17037. arXiv:2309.17037 [cs.IR].

Xiaokun Zhang, Bo Xu, Zhaochun Ren, Xiaochen Wang, Hongfei Lin, and Fenglong Ma. Disentangling ID and Modality Effects for Session-based Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1883–1892, July 2024. doi: 10.1145/3626772.3657748. URL http://arxiv.org/abs/2404.12969. arXiv:2404.12969 [cs.IR].

Yimeng Zhang, Tian Wang, Jiri Gesi, Ziyi Wang, Yuxuan Lu, Jiacheng Lin, Sinong Zhan, Vianne Gao, Ruochen Jiao, Junze Liu, Kun Qian, Yuxin Tang, Ran Xue, Houyu Zhang, Qingjun Cui, Yufan Guo, and Dakuo Wang. Shop-R1: Rewarding LLMs to Simulate Human Behavior in Online Shopping via Reinforcement Learning, February 2026. URL http://arxiv.org/abs/2507.17842. arXiv:2507.17842 [cs].

Yuyu Zhang, Liang Pang, Lei Shi, and Bin Wang. Large Scale Purchase Prediction with Historical User Actions on B2C Online Retail Platform, March 2015. URL http://arxiv.org/abs/1408.6515. arXiv:1408.6515 [cs.LG].

A Representative Session Selection

Let S denote the candidate sessions. Each session s is associated with an outcome y_s, an interaction-sequence length l_s, and a set of encountered products p_s. Our objective is to select a subset of sessions that maximizes product coverage subject to two constraints: (1) the selected sessions are evenly distributed across the outcomes, and (2) the l_s are larger than 3 for meaningful interactions and smaller than the empirical 95th-percentile length constraint to avoid unusually long sessions.

We solve this selection problem using a greedy algorithm. We first filter the candidate sessions according to the sequence-length constraint (e.g. $3 \leq l_s \leq 20$). At each iteration, we consider each outcome and select the eligible session with the largest marginal coverage of previously unseen products. The procedure terminates when it reaches the specified sample-size budget or a configured product-coverage threshold, such as 95%. See Algorithm 1 for pseudocode.

Algorithm 1 Session Selection

Require: Candidate sessions S; optional budget B; product coverage threshold τ; minimum length $l_{\min}$; maximum length $l_{\max}$
Ensure: Selected sessions X

$S' \leftarrow \{s \in S : l_{\min} \leq l_s \leq l_{\max}\}$
$U \leftarrow \bigcup_{s \in S'} P_s$ ▷ U is product union
$\mathcal{Y} \leftarrow \{\text{Browser}, \text{Cart Abandoner}, \text{Checkout}\}$
$X \leftarrow \emptyset,\ C \leftarrow \emptyset$
while $(B \neq \emptyset$ **and** $|X| < B)$ **or** $(B = \emptyset$ **and** $|C|/|U| < \tau)$ **do** ▷ C is the covered product list
  for $y \in \mathcal{Y}$ **do**
    $A_y \leftarrow \{s \in S' \setminus X : y_s = y\}$ ▷ A_y is available sessions for outcome y
    if $A_y = \emptyset$ **then**
      return X with an infeasibility warning
    end if
    $s_y^* \leftarrow \arg\max_{s \in A_y} |P_s \setminus C|$
    $X \leftarrow X \cup \{s_y^*\}$
    $C \leftarrow C \cup P_{s_y^*}$
  end for
end while
return X

Table 6: Fidelity comparison between different generation methods on the `gpt-5.6-sol` model. The best results are shown in **bold**. Statistically significant improvements ($p < 0.05$) over all other baselines are marked with *.

	Outcome-level	Sequence-level			Semantic-level			
Method	Outcome JSD ↓ [0, 1]	Action Freq. JSD ↓ [0, 1]	Transition Matrix L_1 Distance ↓ [0, 1]	Trajectory Levenshtein ↓ [0, 1]	Product Coherence Gap ↓ [0, 1]	Product Diversity Ratio ↓ [0, ∞)	Product Alignment ↑ [−1, 1]	Rationale Alignment ↑ [−1, 1]
Prompting-based methods								
Persona	1.94×10^{-1}	4.39×10^{-2}	3.82×10^{-1}	6.08×10^{-1}	6.06×10^{-2}	1.03×10^{-1}	5.97×10^{-1}	$\mathbf{6.47\times10^{-1}}$
Trajectory	3.98×10^{-3}	1.65×10^{-2}	2.28×10^{-1}	2.59×10^{-1}	7.80×10^{-2}	6.45×10^{-2}	5.41×10^{-1}	6.38×10^{-1}
Trajectory + Persona	1.20×10^{-3}	2.03×10^{-2}	2.38×10^{-1}	2.80×10^{-1}	8.00×10^{-2}	7.14×10^{-2}	5.89×10^{-1}	6.34×10^{-1}
Agentic methods								
UXAgent	7.69×10^{-4}	1.25×10^{-2}	2.53×10^{-1}	3.71×10^{-1}	1.00×10^{-1}	$\mathbf{2.58\times10^{-2}}$	5.92×10^{-1}	6.07×10^{-1}
SimTrace	$\mathbf{0.00}^*$	$\mathbf{5.59\times10^{-3}}$	$\mathbf{2.27\times10^{-1}}$	$\mathbf{2.39\times10^{-1}}$	$\mathbf{5.79\times10^{-2}}$	3.90×10^{-2}	$\mathbf{6.18\times10^{-1}}$	6.41×10^{-1}

B Simulate Twin Environment

We construct a simulated twin of each original website using an autonomous exploration and generation pipeline adapted from ShopGym (Savadikar et al., 2026). The objective is to reproduce the visual organization and interaction patterns that shape the user's navigation experience, while allowing identifying content to be replaced with synthetic alternatives due to sensitivity information. Exploration produces specifications supported by browser evidence. Then the generation module uses these specifications and the captured or synthetic static data to implement a runnable sandbox.

Below, we describe the exploration components, the specification representation, and the generation procedure.

B.1 EXPLORATION

Domain-specific interface-area Configuration. To reconstruct a website faithfully, the agent must first identify both its visual structure and its interactive behavior. We therefore formulate exploration as a configurable inspection process that specifies which interface areas the agent should examine and what evidence it should collect. This configuration serves as a coverage checklist for the coding agent. While some interface areas are common across domains, such as the homepage, header, and navigation, others depend on the target website. The exploration configuration can therefore be adapted to different website domains.

In our e-commerce setting, we configure the following nine interface areas:

1. **Homepage:** the hero region and subsequent sections, including their ordering, layouts, content, and interactive elements.
2. **Header and navigation:** header structure, announcement bars.
3. **Cart:** its presentation as a drawer, popup, or page, together with empty and populated states, quantity changes, and item removal.
4. **Product pages:** image galleries, product information, variant selectors, quantity controls, availability signals, and recommendations.
5. **Collections:** product-card layouts, filtering, sorting, and pagination.
6. **Search:** the search entry point, predictive suggestions, and the presentation of results.
7. **Footer:** link groups, policy links, payment indicators, and social links.
8. **Information pages:** shipping, returns, privacy, terms, frequently asked questions, and contact and about pages when present.
9. **Floating interfaces:** cookie banners, newsletter popups, chat widgets, and age gates, including their appearance and dismissal behavior.

The exploration plan includes all configured areas. When a feature is absent or an interaction cannot be exercised, the agent explicitly records this outcome rather than inferring its behavior. The agent may also append new exploration tasks when it encounters functionality not covered by the predefined configuration.

Interactive inspection. The coding agent executes the exploration plan using Playwright browser automation. Rather than inspecting only static page layouts, it also exercises relevant interactions to capture state transitions. For example, cart inspection includes adding an item, changing its quantity, and removing it, while search inspection includes entering query prefixes and observing the suggestion panel. For each distinct interface state, the agent records a full-page screenshot together with the corresponding accessibility snapshot. We organize this evidence by exploration task so that each observation remains associated with the interface area, interaction, and state that produced it.

Structured properties recording. Screenshots alone do not fully specify how an interface behaves. We therefore complement visual evidence with structured descriptions of each observed element. The agent records six properties: *type*, *layout*, *content*, *behavior*, *styling*, and *UX notes*. These fields capture the element's functional role, spatial arrangement, displayed information, response to user interaction, visual appearance, and relevant edge cases. For example, the description of a navigation panel records its opening trigger, screen position, entry hierarchy, and dismissal behavior. This representation makes interactive functionality and state transitions explicit while retaining screenshots as visual references.

B.2 SPECIFICATIONS

The exploration outputs are consolidated into a specification manual with three complementary components. First, a natural-language design manual summarizes the overall website content, structure

and visual style. Second, a structured element-level specification records interface-specific properties for each area defined in the exploration configuration, including layout, navigation-menu design, filtering dimensions, pagination behavior, product-option controls, etc. Third, we retain prefetched static data files, such as product and collection JSON files, which provide the structured content required to populate the reconstructed interface.

Together, these components summarize the website's interface structure and behavior from its underlying content and provide the coding agent with sufficient information to reproduce the observed environment.

B.3 GENERATION

Given the resulting specification, the generation stage constructs a simulated twin that preserves the structure and interaction patterns of the original environment while replacing sensitive vendor-specific content.

Synthetic content generation. In our e-commerce setting, we treat vendor-specific product information as sensitive content, including product titles, descriptions, vendor identities, and images. We therefore transform the prefetched static product data before using it in the reconstructed website.

For each product, we first identify its original vendor information and generate a synthetic vendor identity. We then regenerate textual product content, including titles and descriptions, conditioned on the synthetic vendor identity while preserving the product's underlying semantic attributes. Product images require additional care because directly transforming the original images may retain identifying or copyrighted visual content. We therefore use a two-stage generation pipeline. The first model produces a textual description of the original product image. A second image-generation model then receives only this description and the synthetic vendor information and generates a new image without access to the original image. This separation preserves high-level product semantics while reducing direct visual reproduction of the source content.

Stepwise source-code generation. Generating the entire website in a single step makes it difficult to maintain consistency and localize errors. Instead, the coding agent implements the configured interface areas incrementally. For each generation task, it receives the relevant portion of the design manual, the associated structured specification and static data, and the current source code. It then implements the corresponding layouts, content, and interactions.

The generation plan, source code, and intermediate artifacts persist across iterations. Consequently, later tasks can build directly on previously implemented components, while errors can be corrected locally without regenerating the complete website. The final application, together with its synthetic content and supporting data, constitutes the simulated twin used for downstream user simulation.

B.4 COST

Constructing a simulated twin requires approximately 28–30M tokens and 2.9–4.2 hours end-to-end using the Pi harness with GPT-5.5, with an estimated API cost of $60–64 per website. Token usage includes cached input, uncached input, and output tokens.

C Dataset

C.1 Persona Example

Persona Example


```
{
    "behavioral": {
        "exploration_depth": 0.45,
        "price_sensitivity": "mid-range"
    },
    "values": {
        "ethics": 0.1,
        "performance": 0.5,
        "premium": 0.2
    },
    "reasoning": {
        "ethics": "No indicators of environmental or ethical
        considerations (e.g., non-toxic plastics, sustainable
        packaging, or local manufacturing) were found in the
        product data. Score: 0.1",

        "exploration_depth": "The user spent a significant amount
        of time (avg 347.5s) relative to the number of products
        viewed (2), suggesting they were carefully reading
        descriptions rather than browsing broadly. Score: 0.45",

        "performance": "100% of browsed products focus on the tactile
        performance and functional outcomes ('slow-rise experience',
        'groovy tactile experience', 'stress relief'). Focus is on
        how the product functions as a sensory tool. Score: 0.5
        (max for browsing only)",

        "premium": "Description mentions 'velvety texture' and
        'perfected the formula', signaling a preference for
        established brand quality over generic toys, though it
        stops short of luxury or exclusive keywords. Score: 0.2",

        "price_sensitivity": "Browsed price bucket is standard for
        name-brand sensory fidget toys. There is no evidence of
        extreme budget-seeking or high-end collector-grade pricing.
        Classification: 'mid-range'"
    },
    "confidence": {
        "behavioral": 0.4,
        "values": 0.2
    }
}
```


C.2 Dataset Details

We use `gemini-3-flash` (Google, 2025) to generate the e-commerce synthetic dataset. We release the dataset under the Creative Commons Attribution-NonCommercial 4.0 International license (CC BY-NC 4.0). The following example illustrates the structure of a complete session in the generated dataset. For readability, the `simplified_dom` and `persona` fields are truncated.


```
{
  "store_id": "68e1563a-676a-694b-ce31-820f79080bad",
  "session_id": "d79f58cc-30fc-046f-bc52-38014c1642a7",
  "user_id": "4c5381ad-ab85-6e0d-ca9f-71ce393443f1",
  "timestamp": "2026-07-01T21:56:48.861744",
  "action_type": "click",
  "target": "needoh",
```

```
  "rationale": "I'll start by checking out the NeeDoh brand collection,
    as I'm looking for
                high-quality sensory toys for my collection.",
  "input_text": null,
  "simplified_dom": "<html><body parser-is-focused=\"true\">
                     ...
                     <a href=\"/collections/novelty-toys\"
                        parser-semantic-id=\"needoh\"
                        parser-clickable=\"true\">
                       <span>NeeDoh</span>
                     </a>
                     ...
                     </body></html>",
  "action_json": {
        "action": "click",
        "description": "Clicking on the NeeDoh brand collection.",
        "rationale": "I'll start by checking out the NeeDoh brand
                      collection, as I'm looking for high-quality
    sensory
                      toys for my collection.",
        "target": "needoh",
        "url": "/collections/novelty-toys"
    },
  "image": "41fd9aa7f6e8c0e5f260c3da.png",
  "intent": "I'm looking for Sensory Toys, greeting cards, sensory toys,
     Stuffed Animals."
  "persona": {
        "behavioral": {
            "exploration_depth": 0.45,
            "price_sensitivity": "mid-range"
            },
        "values": {
            "ethics": 0.1,
            "performance": 0.5,
            "premium": 0.2
            },
        "reasoning": {...},
        "confidence": {...}
    }
}
```


D FIDELITY METRICS

This section defines the eight fidelity metrics reported in Table 4 and Table 6. Let A denote the semantic action set defined in Table 1, with $|A| = 9$. This standardized taxonomy abstracts away website-specific actions and enables the calculation of distributional metrics, such as Jensen–Shannon divergence (JSD) and Transition Matrix L1 Distance.

Outcome JSD measures the divergence between the session outcome distributions of real and synthetic data. We categorize each session as *Browser*, *Cart Abandoner*, or *Checkout*. Jensen–Shannon divergence (JSD) (Lin, 1991) is widely used to evaluate user simulators by quantifying the discrepancy between simulated and real user behavior (Xia et al., 2024). Using base-2 logarithms, the score ranges from 0 to 1, with lower values indicating more similar outcome distributions.

Action Frequency JSD measures the divergence between the marginal action distributions of real and synthetic data. We pool all actions in each corpus into a normalized frequency vector over A and compute the JSD between the two vectors. Unlike Outcome JSD, which considers only terminal outcomes, this metric captures how frequently each action occurs throughout a session. For example, a generator may reproduce the real checkout rate while omitting the `search` and `detail` actions that typically precede a purchase. The score ranges from 0 to 1, with lower values indicating more similar action distributions.

Transition Matrix L_1 Distance measures the similarity of local action-to-action dynamics following prior work (Lu et al., 2026a). For each corpus, we estimate a row-normalized first-order transition matrix $P \in [0,1]^{|A|\times|A|}$, where $P_{ij} = \Pr(a_{t+1} = j \mid a_t = i)$. We compute the normalized L_1 distance as $\frac{1}{|A|}\sum_i \frac{1}{2}\sum_j |P_{ij}^{\text{real}} - P_{ij}^{\text{syn}}|$. This metric captures sequential structure that marginal action frequencies cannot distinguish. For example, replacing every `detail`→`add` transition with `add`→`detail` preserves action frequencies but increases the transition distance. The score ranges from 0 to 1, with lower values indicating more similar transition patterns.

Trajectory Levenshtein measures the normalized Levenshtein edit distance between the semantic action sequences of real and synthetic sessions, following prior work (Sun et al., 2025). A score of 0 indicates that the agent replayed its reference trajectory exactly.

Product Coherence Gap measures whether synthetic sessions maintain the same degree of topical consistency as real sessions. For each session containing at least two products, we compute the mean pairwise cosine similarity between the embeddings of the viewed products. We rescale the value to $[0,1]$. We then report the absolute difference between the mean coherence scores of the real and synthetic corpora. The lower values indicating more similar within-session coherence. A large gap in either direction suggests that the agent either browses across unrelated product categories or focuses more narrowly than real users.

Product Diversity Ratio measures the relative difference in overall product coverage between the real and synthetic corpora. Let n^{real} and n^{syn} denote the numbers of unique products browsed across all real and synthetic sessions, respectively. We compute $|n^{\text{syn}}/n^{\text{real}} - 1|$. A score of 0 indicates that the synthetic corpus covers the same number of unique products as the real corpus, with lower values indicating more similar product diversity. Although this metric is unbounded in principle, the synthetic corpora in our experiments consistently cover fewer products than the real data. This result highlights the need to increase product diversity in future synthetic data generation.

Product Alignment measures whether the real and synthetic sessions involve semantically similar products. For each product in a synthetic session, we retrieve the most similar product from the matched real session using inner-product search over L_2-normalized embeddings and average the resulting values within each session across all matched pairs. The score ranges from -1 to 1, with higher values indicating stronger product alignment.

Rationale Alignment measures whether matched real and synthetic sessions reflect similar underlying shopping intents. Given the action sequence and product context of each session, an LLM generates a natural-language description of the inferred intent. We embed these descriptions and report the mean cosine similarity between the descriptions of each matched real–synthetic pair. This metric captures semantic agreement despite differences in specific actions or products; for example, two sessions may follow different trajectories while both reflecting comparison shopping for a budget-friendly gift.

E User Model System Prompt

The template below is the system message used for training and evaluation. The {`persona`} and {`intent`} fields are filled per session. The user turn supplies the current observation as # `context` followed by the HTML of the page, and assistant turn is the single JSON object the template requires.

User Model System Prompt Template

```
You pretend to be a user browsing the store website and do
shopping based on your intent. Your task is to predict the next
action and provide rationale for the action based on your
persona, intent, previous actions and context. The history
action (with details described below), rationale, context and
the user persona will be provided to you.

# Action Space
Each action object must include an 'action' key specifying one
```

```
of the following types. Include required fields exactly as
shown.
## Click
{ "action": "click", "target": "<element_semantic_id>",
  "description": "Clicking ..." }
## Type (with optional Enter submit)
{ "action": "type", "target": "<input_semantic_id>",
  "text": "<text>", "enter": true,
  "description": "Typing and submitting ..." }
## Select (e.g., dropdowns)
{ "action": "select", "target": "<select_semantic_id>",
  "value": "<option_value>", "description": "Selecting ..." }
## Clear (clear an input field)
{ "action": "clear", "target": "<input_semantic_id>",
  "description": "Clearing ..." }
## Scroll (scroll the chat window or current page)
{ "action": "scroll",
  "target": "<optional_element_semantic_id>",
  "direction": "up", "amount": 300,
  "description": "Scrolling up ..." }
{ "action": "scroll",
  "target": "<optional_element_semantic_id>",
  "direction": "down", "amount": 300,
  "description": "Scrolling down ..." }
## Navigation (use when the chatbot provides a URL to open)
{ "action": "goto_url", "url": "https://example.com",
  "description": "Navigating ..." }
{ "action": "back",    "description": "Going back ..." }
{ "action": "forward", "description": "Going forward ..." }
{ "action": "refresh", "description": "Refreshing ..." }
## Terminate (only if explicitly instructed by the step)
{ "action": "terminate", "description": "Terminating ..." }

# Rationale
The rationale is a first-person sentence (<=25 words)
explaining why you are taking the action. Do not mention HTML,
tag ids, or the simulation.

# Context
Your context will be a HTML of the webpage you are looking at.

# Persona
The user persona reflects the user's price sensitivity,
exploration and preference.
Here is your persona:
{persona}

# Intent
Here is your intent:
{intent}

# Output Format
You need to predict the next action and provide rationale for
the action. Your output should be a single, flat JSON object
with `rationale`, `action`, and the type-specific fields shown
above. For example:
{"rationale": "I want to search for a necklace.",
 "action": "type", "target": "search", "text": "necklace",
 "enter": true, "description": "Typing and submitting necklace
 in the search bar."}

<IMPORTANT>
OUTPUT A SINGLE JSON OBJECT, NOTHING ELSE.
```

```
</IMPORTANT>
```

F Discussion for Next-Action Prediction

F.1 Training Details

We use `Qwen3.5-9B` as the base model for all next-action prediction experiments. Both SFT and RL use full-parameter fine-tuning with BF16 mixed precision and gradient checkpointing. All training is conducted on NVIDIA H200 GPUs.

SFT. We formulate each session as a multi-turn conversation, as described in Section 4.3. To accommodate long web observations and interaction trajectories, we segment each session using a 15-turn sliding window and truncate each example to 32K tokens. We fine-tune the model for up to three epochs with a learning rate of 2×10^{-5}.

RL. We initialize RL training from the model obtained through SFT on synthetic data and optimize it using Group Reward-Decoupled Normalization Policy Optimization (GDPO) (Liu et al., 2026). We uses eight GPUs in total: four for full-parameter policy optimization with DeepSpeed ZeRO-3 and four for response generation with vLLM using tensor parallelism. We use a group size of 4, a global batch size of 32, and a learning rate of 1×10^{-6}. Based on empirical tuning, we set $w_{\text{sub}} = 0.6$, $w_{\text{valid}} = 0.1$, and $w_{\text{exact}} = 0.3$ for R_{target}. The RL stage takes approximately 24 hours per run.

F.2 Performance Gap Between Synthetic and Real Supervision

Figure 2 shows that synthetic supervision from SimTrace improves the base model substantially but still trails supervision from real data. To understand this gap, we examined predictions from the synthetic-only model on the OPeRA test set. The residual gap concentrates in three mismatches between the source platform used for synthetic data generation and the target platform used for evaluation.

Interaction granularity. The two platforms record user behavior at different levels of granularity. The source logs primarily capture actions that trigger page transitions, whereas the target platform also records interactions that modify the current view without navigating to a new page. Examples include cycling through a product image carousel, expanding a review panel, and scrolling through a long product page. Because these fine-grained interactions are absent from the synthetic corpus, the model receives no supervision for predicting them.

Target vocabulary. The platforms expose different interface elements, leaving several target-domain identifiers without source-domain counterparts. For example, targets rooted at `customers_also_bought` or `review` correspond to recommendation and review modules that rarely appear in the source data. This mismatch further compounds the target sparsity discussed in Section 4.3, where 77.4% of unique targets occur only once. By assigning partial credit based on target similarity, RL mitigates this problem but cannot fully recover targets that receive little or no supervision.

Operation semantics. Visually similar elements may also behave differently across platforms. For example, a search button opens the search interface on the source platform but submits a typed query on the target platform. The correct interaction order is therefore click-then-type in the source environment but type-then-click in the target environment. A model trained only on source-domain data learns the former ordering and transfers it to the target platform, producing action sequences that the target interface cannot execute.

The gap may narrow when synthetic interactions are generated within a simulated twin of the target interface, or within the original interface when direct agent interaction is feasible and does not expose sensitive information. Such settings would reduce the interface and functionality mismatches identified above. In our experiments, however, the original activity logs do not contain the web observations associated with each action and therefore cannot directly supervise next-action prediction. SimTrace addresses this limitation by using a powerful teacher agent to replay the logged

behavioral trajectories and reconstruct multimodal web observation–action pairs. From this perspective, SIMTRACE serves not only as an anonymized synthetic data generation framework but also as an effective data enrichment method that transforms otherwise incomplete activity logs into training examples suitable for downstream user-modeling tasks.

F.3 BENEFITS OF SYNTHETIC DATA

Our error analysis indicates that more than half of the exact-match improvement occurs in cases where synthetic data expands the model's effective coverage of the target space. With limited real training data, the model tends to overpredict a small set of frequent targets rather than distinguish among less common interface elements. Synthetic data exposes the model to a broader range of interactions, including suggested-item exploration and product-option selection, thereby improving its ability to identify diverse targets.

Search suggestions provide a representative example. Among the 40 test cases whose ground-truth target was a specific suggested term (e.g., `nav_bar.suggested_terms.bike_bottle_hold`), the model trained only on real data failed to predict any target correctly. Instead, it typically predicted `search_input` or `search_button`, two frequent targets that together account for 24.03% of the real training data. After adding synthetic data, the model correctly identified 22 of the 40 suggested terms, achieving 55.00% accuracy on these cases, while preserving its accuracy on `search_input`.

We observe a similar pattern for product-option attributes. The test set contains four option types: color, scent, size, and style. Because scent does not appear in the real training data, the model trained only on real data never predicts it. Although the synthetic data also contains no scent examples, it introduces related attributes "flavor". After training with these examples, the model correctly predicts scent in some test cases, likely because flavor and scent are semantically related and occupy similar structural roles in the target schema. This result suggests that synthetic data can improve not only direct target coverage but also generalization to structurally and semantically related targets.

F.4 BENEFITS OF REINFORCEMENT LEARNING

RL improves both output validity and target selection. Under the RL objective, which includes a format reward, the rate of malformed JSON predictions decreases from 1.99% to 0%. The target reward also assigns partial credit based on structural similarity, encouraging the model to identify the correct interface region even when it does not recover the exact target.

Our error analysis shows that more than half of the exact-match improvement comes from cases in which RL redirects a prediction from an unrelated top-level UI component to the correct region. For example, given the ground-truth target `buybox.purchase_form.add_to_cart`, the SFT model predicts the unrelated target `reviews.popover.review_images.next`, whereas the RL-trained model recovers the exact target. The remaining improvements primarily occur when both models identify the correct parent path but RL corrects the final leaf segment. Consistent with this pattern, after removing the final leaf segment from each target, the RL-trained model achieves 44.26% parent-path accuracy, compared with 40.71% for the SFT model, an improvement of 3.55 percentage points.