CAN GENERATIVE RECOMMENDERS SERVE AS USER DATA SIMULATORS?

Anonymous authors
Paper under double-blind review

ABSTRACT

User behavior traces are the foundation of recommender systems, but high-quality traces are difficult to obtain. Online A/B testing provides direct feedback but is costly and risky, while existing logs are fixed records of past exposure policies. User simulation offers an offline alternative, yet its usefulness depends on whether generated traces provide useful signals for downstream tasks. This paper explores whether *generative recommenders can serve as user data simulators*. A generative recommender models the conditional distribution of future user behavior given user history, profile, and context; recommendation and user behavior simulation are two interfaces to the same learned behavior distribution. Building on this observation, we present GenSim, a *unified generative user data simulation framework* that turns the generative recommender backbone into a controlled simulation engine. To make the framework practical, we enhance it with two capabilities: the first on *long-horizon generation*, which combines test-time scaling with reinforcement learning on verifiable rewards to control repetition, context drift, and suffix degradation across generated traces; the second on *new user synthesis*, which synthesizes new user profiles together with compatible behavior traces to construct additional user contexts. We instantiate the framework with OpenOneRec and evaluate it along two axes: behavioral characteristics, measuring response-prediction performance and distributional alignment between generated and observed data, and downstream utility, measuring whether generated traces improve recommendation training. Our results show that generative recommenders can serve as practical engines for synthetic behavior trace generation, providing useful supervision for downstream training, with benefits varying across models and tasks.

1 INTRODUCTION

User behavior is the foundation of modern recommender systems, but obtaining high-quality behavior traces remains difficult. *Online A/B testing* provides the most direct user responses, yet it is slow, risky, and unavailable in many research and early-deployment settings. *Existing interaction logs* are easier to reuse, but they are fixed records of past exposure policies: they reveal what users did under previously deployed systems, not how they would respond under new recommenders, items, or behavioral contexts. *User simulation* (Ie et al., 2019; Zhao et al., 2023; Wang et al., 2023b; Zhang et al., 2024) offers a natural third option. By modeling how users respond to recommendations, a simulator can support offline evaluation and policy learning before online deployment. Its value, however, depends on *task-specific utility*: simulated data must be evaluated for the recommendation tasks it is intended to support.

Our key observation is that *generative recommenders offer a natural foundation for user data simulation*. Unlike conventional recommenders that primarily score candidate items, a generative recommender (GenRec) (Geng et al., 2022; Cui et al., 2022; Rajput et al., 2023; Zheng et al., 2024; Zhai et al., 2024; Deng et al., 2025; Zhou et al., 2025b;a) models the conditional distribution of future user behavior given a user's history, profile, and item or context information. This connects directly to user data simulation: given a user representation and a behavioral context, generate plausible future interactions. Thus, once recommendation is formulated generatively, recommendation and simulation become two interfaces to the same learned behavior distribution: recommendation queries it to select likely future items, while simulation samples from it to produce plausible user traces.

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.

This reinterpretation changes the scope of user simulation in three ways. First, it supports *behavior-conditioned generation*: GenRec simulators reuse user and item representations learned for recommendation, conditioning generated traces on historical preferences, recent interactions, and item semantics (Rajput et al., 2023; Zheng et al., 2024; Zhai et al., 2024) within the same item-ID space. Second, it enables *self-contained trace generation*: GenRec simulators can produce complete interaction trajectories without relying on a downstream recommender to drive every simulated exposure, making it useful even when that recommender is weak or newly initialized. Third, it promises *controllable synthesis*: GenRec simulators can be guided by selected user profiles, histories, or item regions, synthesizing targeted traces for new user personas, and cold-start or long-tail items. Together, these properties turn GenRec into a promising substrate for synthetic user behavior trace generation.

We instantiate this view in GenSim, a *unified generative user data simulation framework.* At the core of the GenSim framework is a generative recommender used as an autoregressive simulator: given a user context and interaction prefix, it rolls out future behavior traces over item IDs. This core abstraction turns a single learned behavior distribution into multiple simulation interfaces, including next-step feedback, long-horizon trace generation, cold-item augmentation, and interactive environments for policy learning. To make this abstraction practical, we add two capabilities around the same backbone. First, we enhance *long-horizon generation*. Naively sampling from a generative recommender can produce plausible prefixes but degraded suffixes, with repetition, drift, or loss of user intent (Holtzman et al., 2020). We therefore combine test-time scaling (Snell et al., 2024; Yao et al., 2023), which searches and reranks candidate continuations using sequence-level criteria, with reinforcement learning with verifiable rewards (RLVR) (Shao et al., 2024; Zheng et al., 2025), which aligns generation with explicit proxies for consistency, transition plausibility, diversity, and tail quality. Second, we enable *new user synthesis*. Rather than only extending histories of observed users, we synthesize new user profiles together with compatible behavior traces: the system identifies underrepresented regions in the learned profile space, composes coherent parent profiles, and grounds the resulting synthetic users through GenRec-generated histories. Together, these components turn generative recommenders into controlled user data simulators designed to produce diverse and task-directed behavior traces at training scale.

We implement the GenSim framework on top of OpenOneRec (Zhou et al., 2025b), a representative GenRec model, and we evaluate it around two central questions: how GenSim performs on observed response prediction and distributional diagnostics, and whether its generated traces provide useful supervision for downstream training tasks. For *behavioral characteristics*, we test offline response prediction, cold-item generalization, and distributional alignment between generated and real traces. GenSim improves AUC over the evaluated discriminative simulator (Zhao et al., 2023), including on cold-start items; distributional alignment does not improve uniformly. For *downstream utility*, we use GenSim as a synthetic user profile–trace generator for model training, comparing against both no augmentation and ordinary OpenOneRec generation. We then examine the resulting gains and limitations across sequential recommenders and downstream tasks. These results support GenRec simulators as practical data engines for recommendation training, while distinguishing synthetic-data utility from evidence that generated traces reproduce individual users.

2 BACKGROUND

User Simulation for Recommender Systems. *Discriminative simulators* learn user response models directly from logged behavior traces. Prior work in this vein spans rule-based platforms (Rohde et al., 2018; Ie et al., 2019; Mladenov et al., 2021), GAN-based environment learning (Shi et al., 2019; Bai et al., 2019; Chen et al., 2019), and real-world RL benchmarks (Wang et al., 2023a; Deffayet et al., 2024). A representative system, KuaiSim (Zhao et al., 2023), learns a multi-behavior response model from large-scale short-video logs and supports three evaluation levels: request-level list ranking, whole-session optimization, and cross-session retention. By conditioning on observed user states and item features, these simulators provide data-driven approximations of platform responses. However, they are tightly coupled to their training data distribution: responses for cold items outside the observed support can be less reliable, and composing user profiles to construct new synthetic users and their traces is not generally their primary objective.

LLM-based simulators on the other hand replace learned response models with language model agents, each equipped with a user profile, memory, and action module to mimic human preference reasoning.

RecAgent (Wang et al., 2023b) is an early example, and Agent4Rec (Zhang et al., 2024) instantiates 1,000 persona-driven agents for movie recommendation. Subsequent work extends this paradigm to conversational settings (Zhu et al., 2025; Chen et al., 2025a), RL-based training (Zhang et al., 2025), and cross-domain cold-start scenarios (Chen et al., 2025b), and uses LLM agents as cost-effective human proxies for offline evaluation (Bougie & Watanabe, 2025). Despite richer semantic expressiveness, these simulators face two practical challenges: (i) text-based user and item representations do not necessarily capture the collaborative signals learned by ID-based recommenders; and (ii) linguistically plausible preference signals still require evaluation against real logs, while repeated LLM calls add computational cost to training-scale data generation.

Many systems in both paradigms are primarily designed for *response simulation*: they answer queries from a recommender rather than directly generate complete interaction sequences. Targeted synthesis—directing generation toward selected items or user contexts and constructing additional user profile–trace pairs—therefore remains a complementary objective. A user data simulator can combine learning from behavioral logs and item-ID grounding with semantic representations, extending response simulation toward targeted, large-scale data synthesis.

Synthetic Data for Recommendation. Existing approaches to synthetic data for recommendation share a common goal—supplementing sparse interaction logs—but differ in mechanism: sequence augmentation and self-supervised methods (Zhou et al., 2020; Xie et al., 2022; Qiu et al., 2022; Liu et al., 2021a) perturb existing histories via crop, mask, reorder, or auxiliary views; reverse-direction transformer pretraining (Liu et al., 2021b) and diffusion-based augmentation (Liu et al., 2023) hallucinate pseudo-prior items to pad short sequences; counterfactual methods (Wang et al., 2023c) simulate alternative interaction paths; LLM-based approaches (Wang et al., 2024; Huang et al., 2025; Wei et al., 2024) leverage language models to infer affinities for cold-start items or augment the underlying user-item graph; and dataset-regeneration methods (Yin et al., 2024) regenerate training corpora. Despite their diversity, these methods emphasize particular augmentation objectives: sequence transformations reuse observed histories, counterfactual methods construct alternative scenarios, and LLM-based generators add semantic signals. Our focus is on reusing a pretrained generative recommender as a shared *behavioral model* for conditioned rollout and user profile–trace synthesis. Recent behavior-pretraining methods such as PANTHER (Li et al., 2025) learn transferable user embeddings from behavioral-token prediction, but their generative objective is mainly used for representation learning rather than controllable trace rollout. This motivates studying generative recommenders as a common backbone for conditioned trace generation and new-user data synthesis.

3 METHODOLOGY

3.1 PROBLEM FORMULATION AND OVERVIEW

Let $\mathcal{I}$ be the item catalog (discrete IDs). Given an observed user prefix $\mathbf{h}_{\leq\tau} = (i_1, \ldots, i_\tau)$ with $i_t \in \mathcal{I}$ and optional side context $\mathbf{x}$ (e.g., profile fields or metadata), we model future interactions $\mathbf{y} = (y_1, \ldots, y_T) \in \mathcal{I}^T$ by

$$p_\theta(\mathbf{y} \mid \mathbf{x}, \mathbf{h}_{\leq\tau}) = \prod_{t=1}^{T} p_\theta(y_t \mid \mathbf{x}, \mathbf{h}_{\leq\tau}, y_{<t})\,. \tag{1}$$

A *generative recommender* is a parameterized estimator of this conditional law, trained on logged clicks and engagements. A *user simulator* is the same model used in generation mode: it rolls out this distribution autoregressively to sample synthetic continuations. Hence, next-item prediction and user simulation are two views of the same object; the difference is whether we output one-step scores or full trajectories.

Generative recommenders are strong user simulators for three reasons. First, they are behaviorally calibrated: the model is trained on real interaction logs, so sampling from p_θ directly approximates observed user dynamics rather than hand-crafted response rules. Second, they are ID-grounded: generation happens in the same item-ID space used by production recommenders, preserving collaborative filtering structure that text-only simulators cannot fully recover. Third, they are naturally controllable: autoregressive conditional generation supports prompt-based steering toward target regimes such as cold items, while remaining consistent with user history and context. This advantage is most visible on cold items: Table 1 shows that under freq$= 0$, the discriminative baseline is near

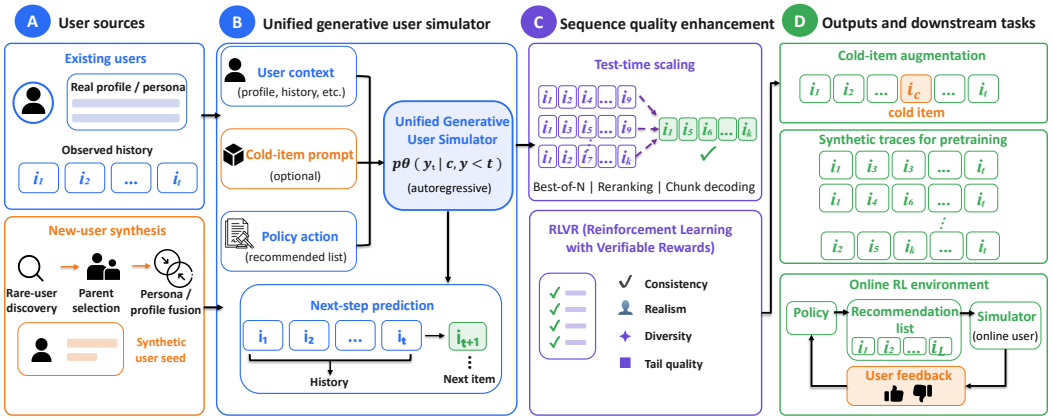


Figure 1: GenSim Overview

random (AUC $= 0.4943$) while GenRec remains strong (AUC $= 0.6963$), and the gap narrows as support grows (freq> 50: 0.6575 vs. 0.6771), matching the expected regime where abundant logs make all simulators easier.

Table 1: Cold-Start Item AUC Comparison Across Frequency Buckets

Frequency Bucket	#Samples	%Total	Pos. Rate	KuaiSim	OpenOneRec-8B-Pro
freq=0 (cold-start)	57,881	16.7%	16.3%	0.4943	**0.6963**
freq=1–5 (very sparse)	54,516	15.7%	24.2%	0.5392	**0.6835**
freq=6–50 (mid-freq)	80,704	23.3%	31.9%	0.5862	**0.6489**
freq>50 (popular)	153,089	44.2%	34.2%	0.6575	**0.6771**
ALL	346,190	100%	29.1%	0.6040	**0.6913**

As shown in Figure 1, we instantiate GenSim on top of a generative recommender p_θ and equip it with two practical capabilities: we enhance long-horizon rollouts using *test-time scaling* together with *reinforcement learning with verifiable rewards (RLVR)* to control repetition and context drift; and we expand the synthetic user population by *synthesizing new profiles and grounded histories* that remain compatible with p_θ. Together, these capabilities turn one autoregressive conditional model into a controllable engine for scalable synthetic supervision in downstream recommendation.

3.2 Enhancing Long-Horizon Generation

Long autoregressive rollouts can accumulate context drift, repeated items, and implausible transitions. We combine chunk-wise candidate verification with sequence-level policy alignment to control these failures. Both components use the recorded user context and explicit sequence-level criteria to guide the generation of synthetic behavior traces.

Test-time scaling. Given context $\mathbf{x}$, we sample N candidate continuations from $p_\theta(\cdot \mid \mathbf{x})$ and rerank them using

$$S(\mathbf{y}; \mathbf{x}) = \lambda_{\text{lm}} \ell_\theta(\mathbf{y} \mid \mathbf{x}) + \lambda_{\text{c}} \widetilde{R}_{\text{cons}} + \lambda_{\text{d}} \widetilde{R}_{\text{div}} + \lambda_{\text{r}} \widetilde{R}_{\text{real}}, \tag{2}$$

where ℓ_θ is the mean generated-token log-probability; prompt and padding tokens are excluded. The behavioral terms use frozen OpenOneRec item embeddings and statistics estimated exclusively from training histories. R_{cons} compares generated-item embeddings with long-term and recent-interest representations obtained by feedback-aware, time-decayed pooling of historical embeddings. R_{div} penalizes exact SID repetition and semantic near-duplicates, distinguishing degenerate recurrence from coherent interests. R_{real} scores transition plausibility using empirical SID-cluster transition probabilities and collaborative co-consumption statistics. Each component is normalized, clipped, and contribution-capped before combination, yielding $\widetilde{R}_k$. No held-out target interaction enters these computations. Rewards operate on item trajectories, whereas policy likelihoods are computed over the corresponding sequences of generated SID tokens.

We select

$$\mathbf{y}^* = \arg \max_{i \in \{1,\dots,N\}} S(\mathbf{y}^{(i)}; \mathbf{x}). \tag{3}$$

For long traces, we partition the requested horizon into chunks of at most B items and repeat this procedure sequentially (Snell et al., 2024; Yao et al., 2023). Writing $\mathbf{h}_m = (\mathbf{x}, \mathbf{c}_{<m})$, candidate sampling follows

$$p_\theta(\mathbf{y} \mid \mathbf{x}) = \prod_{m=1}^{M} p_\theta(\mathbf{c}_m \mid \mathbf{h}_m). \tag{4}$$

At chunk m, all N candidates share the same accepted prefix. Verification includes the prefix–candidate boundary when assessing transitions and repetition; only the selected chunk is appended before next sampling round. Invalid SID sequences and candidates violating the configured repetition constraints are rejected. Thus, search revisits the accumulated trajectory without treating generated items as observed user feedback. Candidate generation is batchable, while frozen embeddings and behavioral statistics are reused across candidates; the sampling budget grows with NM chunks.

GSPO-based RLVR for alignment. Test-time search selects continuations without updating model parameters. We additionally optimize the generator using the same behavioral components and a suffix-specific reward:

$$R(\mathbf{x}, \mathbf{y}) = \sum_{k \in \{\text{cons},\text{div},\text{real}\}} \alpha_k \widetilde{R}_k(\mathbf{x}, \mathbf{y}) + \beta R_{\text{tail}}(\mathbf{x}, \mathbf{y}). \tag{5}$$

R_{tail} evaluates consistency, realism, and non-repetition in the final third of the trace and penalizes deterioration relative to its first third. Both segments use the same frozen scoring functions, so the suffix term emphasizes accumulated degradation rather than introducing an additional learned judge. These rewards are automatically computable offline proxies, not ground-truth measurements of human engagement or causal responses.

For each context, the rollout policy $\pi_{\theta_{\text{old}}}$ samples G complete trajectories. We compute group-relative advantages

$$A^{(g)} = \frac{R(\mathbf{x}, \mathbf{y}^{(g)}) - \text{mean}_j\, R(\mathbf{x}, \mathbf{y}^{(j)})}{\text{std}_j\, R(\mathbf{x}, \mathbf{y}^{(j)}) + \delta}, \tag{6}$$

where $\delta > 0$ stabilizes groups with nearly identical rewards. Following GSPO (Zheng et al., 2025), the importance ratio is length-normalized:

$$\rho^{(g)} = \exp\left[\frac{1}{L_g} \sum_{t=1}^{L_g} \log \frac{\pi_\theta(y_t^{(g)} \mid \mathbf{x}, \mathbf{y}_{<t}^{(g)})}{\pi_{\theta_{\text{old}}}(y_t^{(g)} \mid \mathbf{x}, \mathbf{y}_{<t}^{(g)})} \right], \tag{7}$$

where L_g counts generated tokens only. Holding rollout probabilities and advantages fixed during updates, we minimize

$$\mathcal{L} = -\frac{1}{G} \sum_{g=1}^{G} \min\Big(\rho^{(g)} A^{(g)}, \text{clip}(\rho^{(g)}, 1-\epsilon, 1+\epsilon) A^{(g)} \Big) + \eta \mathcal{D}_{\text{KL}}, \tag{8}$$

where $\mathcal{D}_{\text{KL}}$ penalizes deviation from the frozen original backbone and ϵ controls clipping. Unlike token-wise clipping in GRPO (Shao et al., 2024), GSPO applies a shared ratio to each trajectory, matching the granularity of the trace reward. This motivates the optimizer choice but does not itself establish superior downstream performance.

Frozen reward extractors, bounded contributions, decoding constraints, and KL regularization limit opportunities for reward exploitation. We additionally monitor validation likelihood and distributional alignment to detect reward increases accompanied by collapse. We assess the resulting synthetic traces through distributional diagnostics and downstream training utility, rather than treating higher reward alone as evidence of better synthetic data.

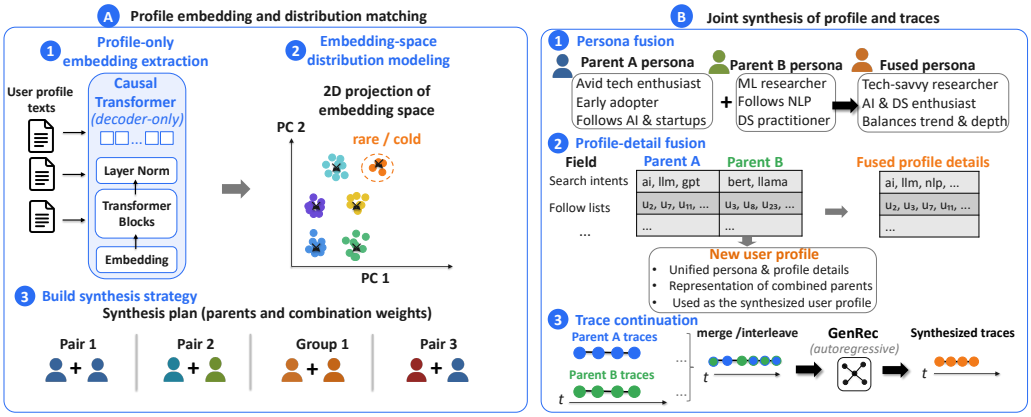


Figure 2: Workflow of New User Profiles and Traces Synthesis

3.3 Synthesis of New User Profiles and Traces

The preceding discussion simulates traces for users already present in logs, but continuing observed identities alone limits *who* the augmented data can represent. To extend simulation beyond users observed in logs, we jointly synthesize new user profiles and corresponding item-ID traces. This supports cold-start and low-activity regimes, policy stress tests under weak-history conditions, and anonymized data release. Formally, a *user profile* is the structured persona surface through which a recommender observes preferences, including demographics, intents, textual summaries, and item-ID behavioral cues.

Profile embeddings and distribution matching. As shown in Figure 2, user profiles are represented as long persona strings that combine structured fields with item-ID behavioral cues. We encode each profile document p_u using the eighth-layer hidden representation of the same OpenOneRec *profile-only* causal LM used elsewhere in this work. Given tokenized input $x_{u,1:T}$, we take the terminal non-padding hidden state $\bar{h}_u$ and normalize it as

$$z_u = \frac{\bar{h}_u}{\|\bar{h}_u\|_2 + \varepsilon}. \tag{9}$$

The resulting embedding z_u provides a compact representation of both profile semantics and behavioral cues. We use cosine similarity in this space to identify compatible parent users, favoring pairs with similar representations so that the resulting fused profiles remain coherent. The same embedding space also provides a common basis for comparing the synthesized population with the training distribution, allowing us to control local parent compatibility while monitoring population-level distributional alignment.

We instantiate these goals with a concrete *sampling-and-generation strategy* detailed below: we first restrict the parent pool to training users with non-empty profiles and histories, sample seed-user pairs under embedding-space cosine-similarity constraints, then generate profile–trace pairs and retain batches whose aggregate statistics match the training distribution.

Joint synthesis of user profiles and traces. Given a sampled seed pair (a, b), we construct the synthetic user through three stages that progressively combine seed attributes, profile semantics, and generated behavior into a single consolidated record.

(1) Symbolic preparation. We merge coarse profile information from both seeds, for example demographic and gender fields, before richer assembly or generative rewriting. This ensures that synthetic users inherit attributes from real users rather than arbitrary free-form text.

(2) User profile fusion via generative recommendation. We then fuse the two seeds' fine-grained profile information, including commerce summaries, search records, and follow/consumption cues, using the same GenRec backbone as for user behavior sequence generation. An instruction-style fusion prompt provides both seed profiles together with a mixing coefficient α, which controls their relative influence on the fused profile. Decoding uses an 8,192-token context and produces a single coherent natural-language profile that emphasizes shared signals while resolving conflicting cues

without naive field-wise averaging. We apply output filtering and up to two retries, followed by a deterministic symbolic fallback when generation remains malformed.

(3) Trace-grounded history with GenRec continuation and summarization. We sample 20% of each seed user's history without replacement, preserve within-user order, and interleave the two samples to obtain a temporary starter trajectory $h_s^{(0)}$. GenRec generates a new SID sequence up to the mean length of the two parent histories, producing $\tilde{h}_s$; the temporary seed trajectory is discarded afterward, so the synthetic trace contains only newly generated interactions. Finally, we summarize salient interests and behavioral patterns from $\tilde{h}_s$ and incorporate them into the fused profile.

To quantify distributional alignment, we compare synthesized users against the training distribution in the same representation space and find a minimal gap ($\text{MMD}^2 = 0.0806$, and 0.1110 after PCA-512). At the same time, synthesis remains non-collapsed (0.649 ± 0.148 mean cosine), and embedding-guided pairing shows a positive margin over random pairing ($+0.090$). Together, these diagnostics indicate that the strategy improves local coherence while preserving global distributional closeness.

4 Evaluation

4.1 Experimental Setup.

We experiment on RecIF-Bench (Zhou et al., 2025b), a large-scale Recommendation Instruction-Following benchmark released by the OpenOneRec project. It contains ∼96 million interactions from 160,000 users across three industrial domains: Short Video, Ads, and Product, covering 11,724,953 item IDs. We use the official chronological split of RecIF-Bench and ensure that all synthetic traces are generated only from training prefixes. For fair comparison, OneRec and GenSim use the same number of synthetic profile-trace pairs. Synthetic traces are mixed with the real training subset at the same ratio across methods, and all models are evaluated on held-out real interactions. We use a server equipped with 8 NVIDIA A100 GPUs, each with 80 GB of GPU memory.

4.2 Simulation Capability

We evaluate GenSim along three axes: distributional consistency, downstream utility from synthetic traces, and offline policy-learning transfer. For distributional consistency, Table 2 compares real traces with OneRec-1.7B autoregressive continuations and GenSim-1.7B long-horizon generations. We report popularity buckets, top-50 caption-keyword overlap, and SID-cluster JS divergence. GenSim better matches the coarse exposure distribution: it increases cold-item coverage from 4.4% to 5.1%, closer to the real 6.7%, and reduces the over-generation of popular items from 40.3% to 31.0%, close to the real 33.3%. It also improves top-50 keyword overlap from 42/50 to 44/50. GenSim improves SID-cluster alignment, reducing JS divergence for both settings. Thus, GenSim improves popularity and topical alignment, while fine-grained semantic-sequence matching remains an open challenge.

Table 2: Distribution Comparison (Real, OpenOneRec-1.7B vs. GenSim)

Category	Metric	Real	OpenOneRec	GenSim
Popularity	Cold (freq = 0)	6.7%	4.4%	5.1%
	Rare (freq 1–10)	15.5%	11.0%	18.6%
	Medium (freq 11–100)	26.2%	25.6%	28.4%
	Popular (freq 101–1K)	33.3%	40.3%	31.0%
	Hot (freq > 1K)	18.2%	18.7%	16.9%
Topic / Category	Top-50 keyword overlap	—	42/50 (84%)	44/50 (86%)
SID Cluster	JS Div. (s_a, 3,808 coarse clusters)	—	0.4495	0.4262
	JS Div. (s_a, s_b joint, 22,998 pairs)	—	0.7657	0.7390

4.2.1 Synthetic Trace Generation for Model Training

Motivation and Setup. A user data simulator can serve as a data engine for recommendation training. We use GenSim-generated profile–trace pairs as synthetic supervision for downstream recommenda-

tion models. To test whether synthetic traces transfer to real data, we train models under three settings: no synthetic data, vanilla OneRec-generated traces, and GenSim traces. Here, OneRec serves as the simulator/generator baseline because both downstream tasks follow the OpenOneRec/RecIF input-output format, and it provides the same generative backbone used by GenSim. This makes the comparison controlled: +OneRec and +GenSim use the same backbone and synthetic-data budget, while GenSim adds long-horizon enhancement and profile-consistent generation.

We evaluate two tasks. The first is profile-and-sequence to label prediction, where the model predicts a binary response label for a given user profile, historical sequence, and target item, matching cascading ranker and re-ranker settings. The second is profile-and-sequence to target prediction, where the model recovers future target items from the item catalog, matching next-token-style generative recommendation. We further ablate long-horizon enhancement (LH) and profile consistency (PC).

We choose downstream learners according to task type. For label prediction, we include InterFormer (Zeng et al., 2025) as a representative heterogeneous feature-interaction model, and TIGER (Rajput et al., 2023)/HSTU (Zhai et al., 2024) as strong sequence-aware generative recommenders adapted to the same profile-plus-history input. For target prediction, we evaluate standard and recent sequential recommenders, including SASRec (Kang & McAuley, 2018), BERT4Rec (Sun et al., 2019), ReaRec (Tang et al., 2025), TIGER (Rajput et al., 2023), and HSTU (Zhai et al., 2024), since this task directly tests future-item recovery from interaction histories.

Table 3: Result of task profile-and-sequence to label prediction.

Model	Metric	w/o synth	+OneRec	+GenSim	+GenSim w/o LH	+GenSim w/o PC
TIGER	AUC	0.6729	0.6751	**0.6768**	0.6749	0.6738
HSTU	AUC	0.6703	0.6745	**0.6758**	0.6742	0.6754
Interformer	AUC	0.6303	0.6390	0.6545	0.6411	**0.6549**

Table 4: Result of task profile-and-sequence to target prediction.

Model	Metric	w/o synth	+OneRec	+GenSim	+GenSim w/o LH	+GenSim w/o PC
SASRec	Pass@32	0.0633	0.0716	**0.0741**	0.0722	0.0735
	Recall@32	0.0195	0.0214	**0.0237**	0.0205	0.0221
BERT4Rec	Pass@32	0.0638	0.0705	**0.0752**	0.0706	0.0728
	Recall@32	0.0196	0.0214	**0.0234**	0.0213	0.0222
ReaRec	Pass@32	0.0600	0.0629	**0.0691**	0.0658	0.0651
	Recall@32	0.0184	0.0188	**0.0207**	0.0198	0.0195
TIGER	Pass@32	**0.0533**	0.0447	0.0516	0.0530	0.0509
	Recall@32	0.0159	0.0126	0.0155	**0.0163**	0.0160
HSTU	Pass@32	0.0692	0.0765	**0.0838**	0.0776	0.0817
	Recall@32	0.0215	0.0233	**0.0256**	0.0235	0.0250

Profile-and-sequence to label prediction. Table 3 shows that GenSim improves AUC over both w/o synth and +OneRec on all three label-prediction models. The gains are largest for HSTU and InterFormer: HSTU improves from 0.6703 to 0.6758, and InterFormer improves from 0.6303 to 0.6545. Vanilla OneRec traces are less reliable, even hurting TIGER and HSTU, while GenSim recovers or surpasses the no-synthetic baseline. The ablations show that LH is important for sequence-oriented backbones: removing LH drops HSTU from 0.6758 to 0.6742 and TIGER from 0.6768 to 0.6749. PC mainly affects profile-conditioned alignment, with a large drop on TIGER when removed. InterFormer is less sensitive to PC, likely because it relies more on heterogeneous feature interaction for single-target prediction.

Profile-and-sequence to target prediction. Table 4 evaluates sequence-level target recovery. GenSim improves over w/o synth for SASRec, BERT4Rec, ReaRec, and HSTU on both Pass@32 and Recall@32. The largest gain appears on HSTU, where Pass@32 increases from 0.0692 to 0.0838 and Recall@32 from 0.0215 to 0.0256. Compared with +OneRec, GenSim also provides consistent additional gains, showing that controlled trace generation is more useful than naive autoregressive synthetic data. TIGER is the exception: vanilla OneRec synthetic traces hurt performance, and

Table 5: Performance of RL agents trained with different user simulators.

Agent	Simulator	Depth ↑	Avg. Reward ↑	Total Reward ↑	nDCG@10 ↑
A2C	KuaiSim	6.10	0.5034	3.2064	0.0854
	RL4RS	5.30	0.4893	2.6351	0.0583
	DGDPO	7.10	0.3902	2.7999	0.0901
	GenSim	**8.10**	**0.7964**	**6.4749**	**0.1604**
SA2C	KuaiSim	5.80	0.4928	2.9273	0.0892
	RL4RS	5.05	0.4577	2.3765	0.0514
	DGDPO	5.95	0.3732	2.2656	0.0862
	GenSim	**8.05**	**0.8605**	**6.9365**	**0.1579**

GenSim partially recovers the drop. This suggests that generative item-tokenization models can be more sensitive to synthetic-target distribution shift.

Ablation Study. The ablation results in Tables 3 and 4 show that both LH and PC contribute to GenSim, but they affect different tasks and model families in different ways. Long-horizon is most important for sequence-output prediction, where models must exploit temporal continuity across the generated trace. This is particularly clear for HSTU and BERT4Rec: removing LH causes larger drops than removing PC in Pass@32, indicating that sequential backbones are highly sensitive to long-horizon rollout quality. Population coverage, in contrast, is more directly tied to profile-conditioned response prediction. When PC is removed, label-prediction AUC drops for TIGER and HSTU, suggesting that synthetic traces become less aligned with the user profile and therefore provide weaker supervision for profile-aware prediction. Together, these results support the design of GenSim: high-quality synthetic recommendation data requires both temporally coherent long-horizon sequences and consistency between generated behaviors and user profiles.

4.2.2 Reinforcement Learning for Recommendation

Deploying exploratory RL recommenders in production is costly and risky, making offline simulators essential for training and evaluation before online deployment (Zhao et al., 2023). We formulate recommendation as a Markov Decision Process (MDP) and use our generative recommender as the interactive environment. At each step, an RL agent issues a recommendation list, and the simulator autoregressively generates multi-behavior feedback, such as clicks and likes, together with session termination signals. To assess whether policies learned in simulation transfer to real user preferences, we train A2C and SA2C through interaction with each simulated environment and evaluate their ranking quality using held-out real long-view labels. Depth measures interaction length, while average and total reward characterize performance within the simulator; we additionally report nDCG@10 on real labels to measure alignment with real user preferences. As shown in Table 5, GenSim achieves the highest depth, average reward, total reward, and nDCG@10 for both agents. Averaged across the two agents, GenSim improves total reward by 119.5% and real-label nDCG@10 by 77.5% over the strongest competing simulator for each agent and metric.

5 Conclusion

In this work, we study generative recommenders as user data simulators that synthesize both user profiles and engagement sequences. We show how a generative recommender can reuse its learned behavior distribution to produce synthetic traces for data augmentation, cold-start analysis, and new-user synthesis. Our framework controls long-horizon generation through sequence-level candidate selection and reward optimization, and constructs additional user contexts through profile-grounded synthesis. Experiments on RecIF-Bench show that the generated traces can provide transferable supervision for downstream recommendation models, with benefits varying across tasks and architectures. We evaluate generated data through offline behavioral diagnostics and downstream transfer; these measurements do not establish that synthetic traces reproduce individual users. Thus, GenSim should be viewed as an offline user data simulator and synthetic-data engine, not a replacement for online A/B testing. Our future work will use larger industrial datasets to improve pretraining and explore continual post-training for generative user data simulators.

AI USE STATEMENT

Generative AI was used to adapt the manuscript's LaTeX formatting to the ICLR 2027 template. After that, we manually checked the paper to ensure that no content was changed. *Authors: before submission, complete this statement with any other AI use in the research and writing process, and describe how AI-assisted work was checked.*

ETHICS STATEMENT

We use publicly available recommendation data and do not attempt to identify individual users. Synthetic behavior may inherit dataset biases and should not be treated as evidence of real user preferences or consent.

REPRODUCIBILITY STATEMENT

We uploaded the code repository as the supplementary code including all hyperparameter settings and multiple random seeds. The datasets are available on Huggingface.

REFERENCES

Xueying Bai, Jian Guan, and Hongning Wang. Model-based reinforcement learning with adversarial training for online recommendation. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Nicolas Bougie and Narimawa Watanabe. Simuser: Simulating user behavior with large language models for recommender system evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pp. 43–60, 2025.

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, 2025a.

Xinshi Chen, Shuang Li, Hui Li, Shaohua Jiang, Yuan Qi, and Le Song. Generative adversarial user model for reinforcement learning based recommendation system. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1052–1061. PMLR, 2019.

Yue Chen, Susen Yang, Tong Zhang, Chao Wang, Mingyue Cheng, Chenyi Lei, and Han Li. Lasso: Large language model-based user simulator for cross-domain recommendation. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, pp. 207–216, 2025b.

Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084*, 2022.

Romain Deffayet, Thibaut Thonet, Dongyoon Hwang, Vassilissa Lehoux, Jean-Michel Renders, and Maarten de Rijke. Sardine: A simulator for automated recommendation in dynamic and interactive environments. *ACM Transactions on Recommender Systems*, 2024. doi: 10.1145/3656481.

Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*, 2025.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pp. 299–315, 2022. doi: 10.1145/3523227.3546767.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020.

Feiran Huang, Yuanchen Bei, Zhenghang Yang, Junyi Jiang, Hao Chen, Qijie Shen, Senzhang Wang, Fakhri Karray, and Philip S. Yu. Large language model simulator for cold-start recommendation. In *Proceedings of the 18th ACM International Conference on Web Search and Data Mining*, 2025. doi: 10.1145/3701551.3703546.

Eugene Ie, Chih-wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. Recsim: A configurable simulation platform for recommender systems. *arXiv preprint arXiv:1909.04847*, 2019.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206, 2018. doi: 10.1109/ICDM.2018.00035.

Guilin Li, Yun Zhang, Xiuyuan Chen, Chengqi Li, Bo Wang, Linghe Kong, Wenjia Wang, Weiran Huang, and Matthias Hwai Yong Tan. Panther: Generative pretraining beyond language for sequential user behavior modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Qidong Liu, Fan Yan, Xiangyu Zhao, Zhaocheng Du, Huifeng Guo, Ruiming Tang, and Feng Tian. Diffusion augmentation for sequential recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 1576–1586, 2023. doi: 10.1145/3583780.3614851.

Zhiwei Liu, Yongjun Chen, Jia Li, Philip S. Yu, Julian McAuley, and Caiming Xiong. Contrastive self-supervised sequential recommendation with robust augmentation. *arXiv preprint arXiv:2108.06479*, 2021a.

Zhiwei Liu, Ziwei Fan, Yu Wang, and Philip S. Yu. Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1608–1612, 2021b. doi: 10.1145/3404835.3463036.

Martin Mladenov, Chih-Wei Hsu, Vihan Jain, Eugene Ie, Christopher Colby, Nicolas Mayoraz, Hubert Pham, Dustin Tran, Ivan Vendrov, and Craig Boutilier. Recsim ng: Toward principled uncertainty modeling for recommender ecosystems. *arXiv preprint arXiv:2103.08057*, 2021.

Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. Contrastive learning for representation degeneration problem in sequential recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 813–823, 2022. doi: 10.1145/3488560.3498433.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems*, volume 36, pp. 33415–33437, 2023.

David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising. *arXiv preprint arXiv:1808.00720*, 2018.

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

Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and An-Xiang Zeng. Virtual-taobao: Virtualizing real-world online retail environment for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4902–4909, 2019. doi: 10.1609/aaai.v33i01.33014902.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1441–1450, 2019. doi: 10.1145/3357384.3357895.

Jiakai Tang, Sunhao Dai, Teng Shi, Jun Xu, Xu Chen, Wen Chen, Jian Wu, and Yuning Jiang. Think before recommend: Unleashing the latent reasoning power for sequential recommendation. *arXiv preprint arXiv:2503.22675*, 2025.

Jianling Wang, Haokai Lu, James Caverlee, Ed Chi, and Minmin Chen. Large language models as data augmenters for cold-start item recommendation. In *Companion Proceedings of the ACM Web Conference 2024*, pp. 726–729, 2024. doi: 10.1145/3589335.3651532.

Kai Wang, Zhene Zou, Yue Shang, Qilin Deng, Minghao Zhao, Runze Wu, Xudong Shen, Tangjie Lyu, and Changjie Fan. Rl4rs: A real-world dataset for reinforcement learning based recommender system. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2595–2604, 2023a. doi: 10.1145/3539618.3591899.

Lei Wang, Jingsen Zhang, Hao Yang, Zhiyuan Chen, Jiakai Tang, Zeyu Zhang, Xu Chen, Yankai Lin, Ruihua Song, Wayne Xin Zhao, Jun Xu, Zhicheng Dou, Jun Wang, and Ji-Rong Wen. User behavior simulation with large language model based agents. *arXiv preprint arXiv:2306.02552*, 2023b.

Xiaolei Wang, Kun Zhou, Xinyu Tang, Wayne Xin Zhao, Fan Pan, Zhao Cao, and Ji-Rong Wen. Improving conversational recommendation systems via counterfactual data simulation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2398–2408, 2023c.

Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 806–815, 2024. doi: 10.1145/3616855.3635853.

Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pp. 1259–1273, 2022. doi: 10.1109/ICDE53745.2022.00099.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Mingjia Yin, Hao Wang, Wei Guo, Yong Liu, Suojuan Zhang, Sirui Zhao, Defu Lian, and Enhong Chen. Dataset regeneration for sequential recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3954–3965, 2024. doi: 10.1145/3637528.3671841.

Zhichen Zeng, Xiaolong Liu, Mengyue Hang, Xiaoyi Liu, Qinghai Zhou, Chaofei Yang, Yiqun Liu, Yichen Ruan, Laming Chen, Yuxin Chen, et al. Interformer: Effective heterogeneous interaction learning for click-through rate prediction. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 6225–6233, 2025.

Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58484–58509. PMLR, 2024.

An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval*, pp. 1807–1817, 2024.

Zijian Zhang, Shuchang Liu, Ziru Liu, Rui Zhong, Qingpeng Cai, Xiangyu Zhao, Chunxu Zhang, Qidong Liu, and Peng Jiang. Llm-powered user simulator for recommender system. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 13339–13347, 2025.

Kesen Zhao, Shuchang Liu, Qingpeng Cai, Xiangyu Zhao, Ziru Liu, Dong Zheng, Peng Jiang, and Kun Gai. Kuaisim: A comprehensive simulator for recommender systems. *Advances in Neural Information Processing Systems*, 36:44880–44897, 2023.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 1435–1448, 2024. doi: 10.1109/ICDE60146.2024.00114.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Guorui Zhou, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Qiang Luo, et al. Onerec technical report. *arXiv preprint arXiv:2506.13695*, 2025a.

Guorui Zhou et al. Openonerec technical report. *arXiv preprint arXiv:2512.24762*, 2025b.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pp. 1893–1902, 2020. doi: 10.1145/3340531.3411954.

Lixi Zhu, Xiaowen Huang, and Jitao Sang. A llm-based controllable, scalable, human-involved user simulator framework for conversational recommender systems. In *Proceedings of the ACM on Web Conference 2025*, pp. 4653–4661, 2025.