SIMPERSONA: LEARNING DISCRETE BUYER PERSONAS FROM RAW CLICKSTREAMS FOR GROUNDED E-COMMERCE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM)-based web agents can navigate live storefronts, yet often collapse to a single "average buyer" policy, failing to capture the heterogeneous, distributional nature of real buyer populations. Existing personalization relies on hand-crafted prompt-based personas that are brittle, hard to scale, and unable to faithfully represent population-level behavior. We introduce *SimPersona*, a framework that learns discrete buyer types from historical traffic and exposes them to LLM agents as compact persona tokens. From raw clickstreams, a behavior-aware vector-quantized variational autoencoder (VQ-VAE) induces a discrete buyer-type space capturing the statistical structure of real buyer behavior and merchant-specific population distributions. Each buyer type maps to a dedicated persona token, and the agent is fine-tuned with these tokens on real browsing traces. At inference, each synthetic buyer is assigned a token with a single encoder forward pass, without any retraining or store-specific prompting, and population-level simulation samples types from each merchant's empirical codebook distribution. Evaluated on 8.37M buyers across 42 held-out live storefronts, *SimPersona* achieves 78% conversion-rate alignment with real buyers, exhibits interpretable behavioral variation across buyer types, and outperforms larger-parameter baselines on goal-oriented shopping tasks. We release an open-source data pipeline that converts raw e-commerce event logs into buyer representations and agent-training traces.

1 INTRODUCTION

Simulating realistic human shopping behavior has direct commercial impacts where even marginal improvements in buyer modeling translate into measurable gains in recommendation (Ni et al., 2018; Shi et al., 2025b), storefront evaluation (Lu et al., 2025), or synthetic A/B testing (Lu et al., 2026a). LLM agents have recently made such simulations feasible at scale, navigating live storefronts and executing complete shopping flows from search to checkout (Deng et al., 2023; Zhou et al., 2024; Gur et al., 2024). Yet, while these agents master the *mechanics* of web interaction, they learn a single population-level policy that produces an "average buyer": they know *how* to shop, but not *who* they are shopping as. That is, steering an agent to reflect a specific buyer segment, the *persona problem*, remains the core challenge, and reconstructing the *distribution* of buyer types that defines a store's traffic remains open (Wang et al., 2026; Lu et al., 2026b; Gebreegziabher et al., 2026).

The standard approach to the persona problem is to tell the agent who to be via prompts where memory modules (Park et al., 2023; 2024), purchase-history summaries (Shi et al., 2025a;b), or role-playing profiles (Shao et al., 2023; Chuang et al., 2024) are leveraged for steering. However, prompt-based conditioning is brittle: behavior shifts with wording (Lutz et al., 2025), demographic role-playing fails to match real responses (Chuang et al., 2024), and the best prompt-based shopping agents reproduce only 11.9% of real buyer actions (Lu et al., 2026b). Text-based personas are also expensive (requiring auxiliary LLM calls and consuming context at every step) and limited in expressiveness, since hand-crafted templates cannot capture the full diversity of real behavior. Crucially, they do not solve the *distribution* problem, because personas are crafted individually rather than learned from the full population, they carry no notion of how many buyers of each type exist, yet realistic simulation demands not just individual personas but the correct *mix* of buyer types that

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.

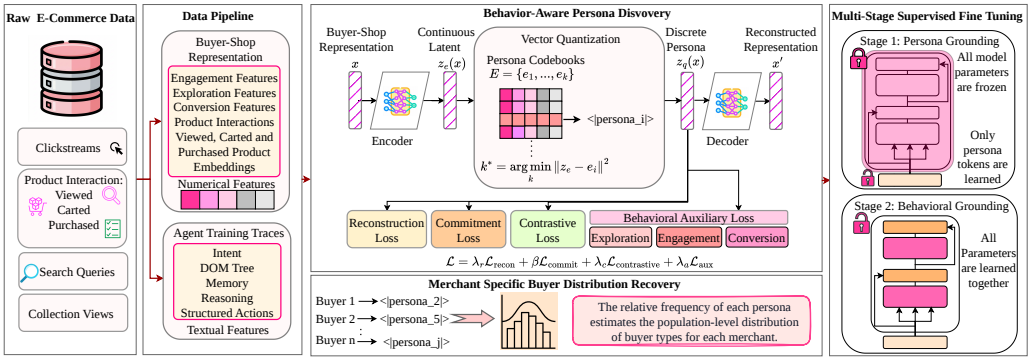


Figure 1: SIMPERSONA framework overview. Behavioral features and product embeddings are extracted from raw clickstreams and fed to a behavior-aware VQ-VAE that maps each buyer to one of K persona tokens. Then, a two-stage SFT grounds the tokens in the LLM.

defines a store's traffic. Recent RL-based agents (Wang et al., 2025; Zhang et al., 2026) face the same limitation, as their personas also derive from LLM-generated text profiles.

The user-modeling literature takes a data-driven approach, learning structured buyer representations directly from raw clickstreams (Black et al., 2024; Ni et al., 2018; Ozyurt et al., 2022; Hatt & Feuerriegel, 2022; Wang et al., 2016). These methods reveal that shopping behavior has rich latent structure that textual profiles cannot express, and in principle can capture population-level variation. While providing high expressivity, they are designed for offline user modeling; their representations are used for tasks such as prediction, recommendation, and clustering, not for driving agent behavior in closed-loop environments. User modeling solves persona *discovery* but not persona *grounding*: it captures who a buyer is without teaching an agent how to act on the provided information.

We propose SIMPERSONA (Figure 1), a framework that jointly addresses persona discovery, persona grounding, and population distribution learning by representing buyer behavior as *discrete persona tokens*: compact enough to occupy a single context position, yet expressive enough to capture real behavioral structure. A behavior-aware vector-quantized variational autoencoder (VQ-VAE) (Van Den Oord et al., 2017) maps each buyer's de-identified historical clickstream to one of K persona tokens, each encoding a coarse distinct behavioral profile. Because these tokens are learned end-to-end from real traffic, the distribution over token assignments naturally mirrors the true population mix of buyer types, enabling faithful reconstruction of store-level traffic patterns, something text-based methods fundamentally cannot provide. The learned tokens are added directly to the LLM vocabulary, making them compatible with token-based agents and assignable to new buyers with a single encoder forward pass. Because newly introduced persona tokens must be aligned with the pretrained model's existing semantic and action spaces, we introduce a two-stage persona-grounding procedure that decouples learning *what* each token means from learning *how* to act on it. This prevents shortcut learning from surface cues in the prompt and produces persona embeddings that transfer across unseen stores. Training uses only a small corpus disjoint from evaluation to encourage the model to learn reusable behavioral patterns rather than memorizing sessions. At inference, persona assignment scales to millions of buyers in seconds without retraining or per-store calibration. In summary, our contributions are:

1. SIMPERSONA, a framework that learns discrete persona tokens from raw clickstreams via a behavior-aware VQ-VAE, enabling scalable persona assignment, faithful reconstruction of buyer population distributions, and closed-loop agent simulation.
2. A two-stage persona-grounding framework that decouples *what* each token means from *how* to act on it, producing embeddings that generalize across storefronts without adaptation.
3. An open-source data pipeline for converting raw clickstreams into buyer representations and agent-training traces, providing infrastructure for buyer behavior simulation.
4. Evaluation on 8.37M unseen buyers showing 78% conversion alignment, statistically significant behavioral separation, and stronger instruction following than 8× larger baseline.

The remainder of the paper is organized as follows. Section 2 details the proposed framework, Section 3 evaluates it on 8.37M buyers, and Section 4 discusses limitations and future directions.

2 PROPOSED METHOD

As detailed below, we build persona-conditioned shopping agents from raw clickstream data by developing a data pipeline for buyer representations and agent traces (Section 2.1), a behavior-aware VQ-VAE that learns discrete personas and maps them to trainable tokens (Section 2.2), and a multi-stage fine-tuning procedure that grounds these tokens in agent behavior (Section 2.3).

2.1 DATA PIPELINE

Existing buyer-simulation methods begin from curated inputs including benchmark tasks, explicit shopping preferences, or simulated user profiles rather than the raw clickstream logs available in production e-commerce systems (Lu et al., 2025; Wang et al., 2026; Lu et al., 2026a). In real platforms, however, buyer behavior is observed through fragmented low-level events (page views, searches, cart mutations, etc) and these logs are optimized for analytics rather than for buyer modeling or agent training. As a result, they contain rich behavioral signal, but not in a form usable by representation-learning methods or LLM agents. Converting raw clickstreams into structured buyer representations and grounded multi-turn action traces is therefore a necessary but largely underexplored step.

We address this gap with a modular pipeline that performs a single enrichment pass over raw clickstreams and produces two complementary data types, Figure 2. It first produces compact aggregated buyer-level representations that combine behavioral statistics with semantic product context; enabling downstream tasks such as segmentation, clustering, and persona discovery. Then, it generates executable agent traces based on live storefront interactions. The traces are formatted as multi-turn sequences of DOM observations, structured actions, memory, and step-level reasoning for supervised fine tuning (SFT). These components are detailed below in the context of persona learning (the pipeline itself is task-agnostic and open-source[1])

2.1.1 BUYER BEHAVIORAL REPRESENTATION

We begin with raw platform logs containing page views, cart actions, search activities and other fine-grained interaction records. These records capture interactions but provide limited context about *what* the buyer was interested in or how their actions relate across a session. Hence, we enrich the logs by augmenting them with the product catalog, collection directory, and search-query records; producing aggregated buyer–session data that combine behavioral signals with semantic context (e.g., product titles, collection names, and search queries), see Figure 3.

For persona discovery, we compress each buyer–session history into a single vector that captures both *how* the buyer shops and *what* they shop for. To represent shopping style, we extract 16 scalar features organized into five groups: *exposure & volume* (total sessions, active days, session counts across product views, add-to-cart, checkout, search, and collection browsing), *engagement* (total session duration and total product views), *funnel* (add-to-cart, checkout, and browse-only rates), *intent strength*, and *dollar values* (average cart and order value); details provided in Appendix A. Product preference is captured by averaging the 768-dimensional product embeddings over each buyer's viewed, carted, and purchased items, then reducing each to 128 dimensions via PCA (retaining $\sim$85% variance) so that the three semantic channels do not overwhelm the behavioral scalars. Three binary masks are included to show which embedding channels are observed. The resulting 403-dimensional vector provides a compact buyer representation for each buyer–shop pair (see Figure 4).

Many downstream components require a shared notion of where a buyer lies in the conversion funnel. Instead of using task-specific heuristics, we define a single *funnel stratification* from aggregated event counts. Each buyer is assigned, via a priority cascade extracted from their raw behavioral signals, to one of five strata: **A** purchasers, **B** checkout abandoners, **C** cart builders, **D** window shoppers, or **E** bouncers. We reuse this label in the pipeline as it guides intent derivation for simulation, supports stratified sampling during training, and provides diagnostics for representation quality (Section 3.1).

[1]URL is anonymized for review, but source code will be provided with submission.

2.1.2 FROM CLICKSTREAMS TO AGENT TRACES

Buyer representations capture behavioral variation but fail to inform an agent about how those behaviors sequentially unroll in a storefront. To address this issue, we replay real buyer sessions on live storefronts. From the enriched session tables generated in the previous step, we reconstruct each buyer's timestamped event sequence and use an LLM (GPT-5) to rewrite it as a natural-language navigation goal that preserves the structure of the original session. A separate agent (Gemini 3 Flash) then executes this goal on the live stores, interacting with real pages and recording its trajectory in an SFT-ready format. Each generated training example consists of a shared *system prompt* that defines the agent's role, output schema, and interaction rules, paired with a *user prompt* containing the session's inferred intent, the buyer profile field (populated with the learned persona token), a cumulative progress log that serves as the agent's memory for reasoning and error recovery, and the current page's DOM snapshot. The corresponding *assistant turn* contains a structured JSON action and a reasoning trace (Figure 5). Each example thus teaches the model to interpret page state, track progress, recover from errors, and most importantly, navigate according to the learned buyer persona. An independent LLM judge (GPT-5) filters the corpus to retain only successful trajectories.

2.2 PERSONA DISCOVERY VIA BEHAVIOR-AWARE VQ-VAE

Our goal is to use raw clickstreams to learn latent buyer behavior that is usable by a language model. To this end we build a VQ-VAE where the encoder compresses high-dimensional behavioral features and the quantization layer maps them to a finite codebook whose entries map one-to-one to new trainable tokens in an LLM's vocabulary. Unlike distance-based clusterings such as k-means, our model is trained end-to-end with a behavior-aware contrastive objective and semantic auxiliary heads, encouraging codebook entries to reflect meaningful behavioral similarity rather than mere geometric proximity. This discretization also enables *distribution learning*, because the codebook is learned jointly over the full buyer population, the frequency with which buyers are assigned to each token provides a natural estimate of the latent distribution of buyer types. In Appendix B, we show that this learned distribution recovers aggregate population-level buyer behavior.

Encoder maps the input $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ through Linear→ReLU→Dropout blocks to a latent vector $\mathbf{z}_e \in \mathbb{R}^D$, which is quantized to its nearest codebook entry $\mathbf{e}_{k^*}$ from a learned codebook $\{\mathbf{e}_1, \ldots, \mathbf{e}_K\} \subset \mathbb{R}^D$:

$$k^* = \underset{k \in \{1,\ldots,K\}}{\arg\min} \|\mathbf{z}_e - \mathbf{e}_k\|_2^2, \qquad \mathbf{z}_q = \mathbf{e}_{k^*}, \tag{1}$$

with gradients propagated via the straight-through estimator (Van Den Oord et al., 2017). The decoder mirrors the encoder and reconstructs the input from $\mathbf{z}_q$. Codebook entries are initialized with k-means++ (Arthur & Vassilvitskii, 2007) over encoder outputs and updated via exponential moving averages (EMA). To prevent codebook collapse (Razavi et al., 2019), we track an EMA of code usage and mark entries as dead when their count falls below a fraction α of the mean. After an initial warmup, dead entries are reinitialized from randomly sampled active encoder outputs. Details regarding hyperparameters are provided in Appendix C (code is available in data pipeline repository).

Reconstruction loss does not guarantee semantically meaningful personas since similar features may correspond to buyers with different behaviors. We therefore augment the standard VQ-VAE objective with contrastive and auxiliary supervision to enforce semantic clustering:

$$\mathcal{L} = \lambda_r \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{commit}} + \lambda_c \mathcal{L}_{\text{contrastive}} + \lambda_a \big(\mathcal{L}_{\text{engage}} + \mathcal{L}_{\text{explore}} + \mathcal{L}_{\text{purchase}}\big). \tag{2}$$

Our reconstruction term ($\mathcal{L}_{\text{recon}}$) is group-aware. Scalar behavioral features are reconstructed with MSE, while product-preference embeddings are reconstructed with cosine distance to preserve semantic direction rather than magnitude. Also, to prevent the high-dimensional embedding channels from dominating the loss, we compute reconstruction at the level of semantic groups rather than raw dimensions, and we mask embedding terms when the corresponding interaction type is absent (a buyer who never added to cart incurs no loss on the carted-product embedding). The commitment loss $\mathcal{L}_{\text{commit}} = \|\mathbf{z}_e - \text{sg}(\mathbf{z}_q)\|_2^2$ keeps encoder outputs close to their assigned codes.

Reconstruction loss shapes what the codebook preserves, but provides no direct signal about *which buyers should share a code*. We propose a three-stage contrastive objective via InfoNCE (Oord et al., 2018) that constructs positive pairs through progressively finer gates. *Stage 1 (funnel gate)*: each buyer receives an ordinal signature from their highest interaction level (purchase>cart>view>none)

where only same-signature buyers can form pairs. *Stage 2 (product filter)*: within the same-signature pool, the top-M peers are retained by cosine similarity in product-embedding space, ensuring paired buyers engage with similar products. *Stage 3 (behavioral filter)*: from these candidate peers, the top-F neighbors ($M > F$) are selected using Euclidean distance over exploration and engagement features (product views, search sessions, collection views, session duration), enforcing similar browsing style on top of similar funnel depth and product interest. All non-self samples remain in the InfoNCE denominator, so the model is simultaneously encouraged to pull behaviorally aligned buyers together and push apart incompatible ones. See Appendix D for formulation.

To further encourage the codebook to capture interpretable shopping behavior, we attach 3 auxiliary classification heads to $\mathbf{z}_q$. Each head is a cross-entropy loss to predict a coarse three-level label (low, medium, or high) along a behavioral axis: *engagement depth*, *exploration breadth*, and *purchase intensity*. These auxiliary tasks provide supervised pressure for the codebook to preserve behavioral distinctions. The labels are derived from buyer-level aggregate scores, with binning tailored to each target's distribution. Engagement is measured by total session duration and exploration by the number of search, collection-view, and product-view sessions; both are approximately log-normal, so we log-transform them and split into percentile-based terciles. Purchase intensity is defined as $8 \times \texttt{checkout_sessions} + 3 \times \texttt{atc_sessions}$ (see Appendix E), which produces a discrete, heavily zero-inflated distribution with natural gaps between non-purchasers, light buyers, and heavy buyers. We place bin boundaries at these gaps to obtain three groups that align with distinct purchasing behaviors. To prevent the model from ignoring rare but important groups, we apply inverse-frequency class weights $w_i = N/(B \cdot n_i)$, where N is the training-set size, B=3, and n_i is the count in bin i. After training, these heads also provide interpretable labels for each codebook entry, making it straightforward to inspect what each persona code represents (see Appendix D).

2.3 Multi-stage supervised fine-tuning

The VQ-VAE produces discrete persona indices, but these indices have no inherent semantics for the language model. We therefore extend the Qwen3-14B-Base (Yang et al., 2025) tokenizer with K special tokens (<|persona_0|>, ..., <|persona_{K-1}|>), one for each codebook entry. This creates a central learning problem as the model must learn to *condition its actions on the persona token* rather than on surface cues in the prompt. End-to-end fine-tuning fails at this stage: randomly initialized embeddings inject noise early in training, while the model quickly discovers that simple lexical cues in the inferred intent such as "buy" versus "browse" provide an easier shortcut, allowing it to ignore the persona token. To address this, we propose a two-stage training framework that explicitly separates learning *what each persona token means* from learning *how to act on it*. We ablate this choice in Appendix F and find that two stage training is essential for robustness.

In Stage 1 (persona grounding), we freeze the pretrained backbone and train only the K new embeddings. Each training example is a browsing trace from the data pipeline (Section 2.1.2 and fig. 9a) in which the buyer profile field contains the persona token, and the goal is an intent for session. Crucially, all goals in this stage are *intent-neutral*: short product-interest statements such as "You are interested in skirts," with the category inferred from the buyer's interaction history. Because the goal never reveals whether the buyer will browse, cart, or purchase, the only signal distinguishing a high-conversion token from a low-conversion token is the statistical pattern of actions across training traces. This signal forces the embedding to absorb the behavioral meaning of its persona. At the end of Stage 1 each persona embedding has converged to a distinct, stable region of the representation space without perturbing the pretrained weights.

In Stage 2 (action-oriented fine-tuning), we unfreeze the backbone and continue training on richer, goal-directed traces. Goals now carry explicit intent derived from the buyer's funnel stratum: sessions from strata A–C receive prompts such as "You are here to buy skirts," while stratum-D sessions receive "You are here to browse skirts." Since the intent is now present in both the goal *and* the persona token, the model must learn to fuse two complementary signals: the token encodes the buyer's general behavioral profile while the goal specifies the intent of *this* session (see Figure 9b). Neither signal is redundant, e.g., a high-purchase token paired with a browse goal should still produce exploratory behavior and so the model cannot collapse to either cue alone. Both stages draw from a small pool of shops fully disjoint from evaluation. Because Stage 1 anchors the embeddings via aggregate behavioral co-occurrence against a frozen backbone, the learned tokens transfer to unseen storefronts without adaptation; we confirm this in Section 3.3 (architectural details in Appendix G).

3 DISCUSSION AND RESULTS

In this section, we evaluate our method along four axes: persona clustering quality (Section 3.1), conversion alignment with real buyers (Section 3.2), fine-grained behavioral fidelity (Section 3.3), and instruction-following performance against GPT-OSS-120B (Section 3.4).

3.1 DOES OUR CLUSTERING LEARN BEHAVIORALLY MEANINGFUL PERSONAS?

We train the VQ-VAE on a balanced subset of 39 shops with sufficient support across strata A–E (Section 2.1). For each shop, we sample up to $1{,}500$ buyer-shop pairs, capped at 300 per stratum, to avoid over-representing either large shops or dominant buyer types. After removing stratum E (bouncers), which contains limited behavioral signal, the resulting dataset contains $44{,}559$ buyer-shop pairs, split $85/15$ into training and validation sets. The model takes as input 16 behavioral scalars together with three product embeddings explained in Appendix A and Section 2.1.

We evaluate the learned VQ-VAE codebook by comparing it against MiniBatch k-means trained on the same buyer representations with $K{=}256$ clusters (see Appendix H for VQ-VAE robustness analysis). We assess four complementary aspects of cluster quality. *Stratum purity* measures whether each cluster respects the funnel strata from Section 2.1; we flag clusters with *incompatible mixing*, where high-funnel buyers (strata A–C) and window shoppers (stratum D) co-occur in substantial proportions. *Auxiliary-head coherence* measures behavioral consistency along the three supervised axes from Section 2.2 by reporting the percentage of buyers assigned to codes that span non-adjacent bins (e.g., low and high). *Pairwise cosine similarity* quantifies how alike buyers within the same cluster are in the original feature space. Finally, the *Calinski–Harabasz index* (Caliński & Harabasz, 1974) summarizes the ratio of between-cluster to within-cluster dispersion; higher values indicate tighter and better separated clusters (see Appendix I).

Based on the results summarized in Table 1, the most striking difference is in behavioral semantics. Because k-means has no notion of what the features *mean*, it simply minimizes Euclidean distance and freely merges buyers at opposite ends of a behavioral axis whenever doing so reduces geometric cost. This is visible in the auxiliary-head coherence: 66.7% of buyers fall in engagement-incoherent codes under k-means, meaning low-engagement and high-engagement buyers are routinely assigned to the same cluster. VQ-VAE, by contrast, keeps incoherence below 0.5% on all three heads. The same pattern appears in stratum purity (84.5% vs. 78.9%) and incompatible mixing (0 vs. 6 severely mixed codes): without semantic supervision, k-means conflates fundamentally different buyer types. On the combined coherence metrics (pairwise cosine similarity and CH index), VQ-VAE leads across the board, the modest margin is because k-means achieves strong tightness on the product-embedding dimensions, which dominate the concatenation by sheer count. But this geometric tightness is hollow as it comes at the cost of the behavioral coherence that matters for downstream persona conditioning.

3.2 DO PERSONA-CONDITIONED AGENTS MATCH HUMANS' CONVERSION RATES?

Fine-tuning uses the same 39 training shops but only a small subset of approximately $3{,}600$ sessions. We deliberately downsample this set to ensure that every codebook entry is represented while keeping the corpus small enough to discourage memorization. At inference time, we evaluate on the full buyer population, spanning 8.37M unique buyers across 42 shops, none of which overlap with the training shops. To challenge generalization beyond unseen storefronts, we deliberately evaluate across retail-domain, interface and temporal shifts, using shop-specific evaluation windows distributed across

Table 1: VQ-VAE vs. MiniBatch k-means ($K{=}256$). Coherence: $\%$ of buyers assigned to clusters spanning non-adjacent bins (↓).

	Stratum purity		*Coherence (% incoh. ↓)*			*Separation*	
Method	Mean purity	Incompat. mixing	Engage	Explore	Purchase	PW cosine	Calinski–Harabasz
k-means	78.9%	6	66.7%	4.6%	5.4%	0.722	173.8
VQ-VAE	**84.5%**	**0**	**0.5%**	**0.0%**	**0.0%**	**0.774**	**206.5**

Table 2: Conversion alignment by funnel stratum. Correct: matched token. All-Mis: averaged over all wrong tokens. 1-R: one random wrong token.

	ATC Alignment			Purchase Alignment			ARA		
Stratum	Correct	All-Mis	1-R	Correct	All-Mis	1-R	Correct	All-Mis	1-R
A	.706	.496	.522	.742	.516	.537	.724	.506	.530
B	.736	.465	.429	.760	.598	.544	.748	.532	.487
C	.767	.438	.442	.674	.755	.720	.721	.597	.581
D	.942	.653	.644	.965	.792	.769	.954	.723	.707
Stratified	**.788**	.513	.509	**.785**	.665	.643	**.787**	.589	.576

approximately one year, spanning more than 10 countries across 6 continents, vary by orders of magnitude in GMV. Of the evaluation shops, 64% operate in product verticals absent from training, and each shop's evaluation straddles a live storefront theme change. Unlike the stable single-theme periods used for training, these changes alter navigation and product presentation, testing whether persona tokens remain meaningful when the interface changes. Each agent is deployed on the *live* storefront visited by its real counterpart and is conditioned on two inputs: a persona token specifying the buyer profile, and a shopping intent (same as Stage 2 in Section 2.3). The agent then navigates the storefront autonomously, and we ask whether its conversion behavior matches that of the real buyers represented by the same persona.

Answering this question requires a metric beyond standard task-completion benchmarks (Zhou et al., 2024; Deng et al., 2023; Yao et al., 2022), where success is binary. For buyer simulation, what matters is not whether the agent *can* convert, but whether it converts at the *same rate* as the real buyers it represents. We therefore define:

$$\text{ATC alignment} = 1 - \left|\text{ATC}_{\text{real}} - \text{ATC}_{\text{agent}}\right|,$$
$$\text{Purchase alignment} = 1 - \left|\text{PUR}_{\text{real}} - \text{PUR}_{\text{agent}}\right|,$$
$$\text{Action Rate Alignment (ARA)} = 0.5 \times \text{ATC alignment} + 0.5 \times \text{Purchase alignment},$$

where ATC and PUR are the add-to-cart and checkout fractions per $(\text{shop}, \text{token})$ pair, stratified by funnel stratum to prevent dominant non-converting populations from washing out rare but important buyer types. To verify that alignment is driven by the persona token itself and not by shop-level tendencies, we compare *correct* pairings where an agent conditioned on token a is evaluated against buyers assigned to token a, against two mismatch baselines: *all-mismatch*, where the same agent is compared against buyers of every other token $b \neq a$ on that shop and averaged, and *random-mismatch*, where the agent is compared against buyers of one randomly chosen incorrect token. If persona tokens carry meaningful signal, correct pairings should consistently outperform both baselines (more ablations and analysis along with a **prompt-based persona baseline** are provided in Appendix J).

Table 2 breaks down alignment by funnel stratum. Stratum D is the easiest to match, since these buyers rarely add to cart or proceed to checkout; the target behavior is largely to browse without converting, and the agent reaches 95.4% ARA. The task becomes harder deeper in the funnel. Stratum C requires the agent to add to cart at the right rate *without* proceeding to checkout, while strata B and A demand even finer control over when to cart, when to continue through checkout, and when to complete a purchase. These higher-funnel groups are also more behaviorally diverse, yet the agent still achieves strong alignment. Critically, on the stratified aggregate, correct pairings outperform the all-mismatch and random-mismatch baselines by 19.8 and 21.1pp in ARA, respectively. The widest gap is on add-to-cart alignment (27.5–27.9pp), a discretionary, persona-specific action with high variance across buyer types, while purchase alignment shows a smaller but consistent gain (12.0–14.2pp). The stratified ARA of 78.7% versus 58.9% under mismatch confirms that a single discrete token meaningfully steers conversion behavior to match the real buyer population, not just at the easy end of the funnel but across all levels of behavioral complexity.

3.3 Do persona-conditioned agents preserve behavioral dimensions?

In this section, we ask a finer-grained question: do the behavioral dimensions learned by the auxiliary heads also manifest in the agent's simulated behavior? As explained in Section 2.2, each persona token

Table 3: Behavioral separation across persona-token bins. Mean score is the average event count per simulation. Statistical significance is assessed at $\alpha = 0.05$.

	Mean score per bin			*Statistical tests*			
Head	Low	Med.	High	Cohen's d	t-test p	KW p	Perm. p
Purchase	0.06	2.94	7.25	2.80	5.5×10^{-22}	4.5×10^{-14}	< 0.0001
Engagement	1.55	2.00	2.41	0.76	4.1×10^{-3}	1.2×10^{-2}	0.003
Exploration	1.34	1.29	1.39	0.10	0.81	0.78	0.48

Purchase-bin funnel progression

Purchase bin	Low	Medium	High
Action	Rate	Rate	Rate
Add-to-Cart	1.0%	29.4%	72.0%
Checkout	0.3%	13.7%	43.3%

encodes a Low/Medium/High bin on three behavioral axes. If these latent distinctions are preserved end-to-end, tokens in higher bins should produce systematically different simulated behavior than those in lower bins. We test this by grouping tokens according to their auxiliary-head bin and comparing simulated behavioral scores across groups using four complementary statistics: Cohen's d for the standardized LOW-vs-HIGH effect size (Cohen, 2013), Welch's t-test for pairwise mean differences (Welch, 1947), the Kruskal–Wallis test for any difference across the three groups (Kruskal & Wallis, 1952), and a permutation test with $10{,}000$ random label reassignments (Fisher, 1935).

Purchase intensity is the clearest example of why persona conditioning matters beyond coarse task intent. In our simulation setup, the intent specifies whether a session is broadly *buy*- or *browse*-oriented, but this single bit groups fundamentally different buyers across strata A, B, and C. Without persona conditioning, the agent has no way to distinguish these behaviors; it can only produce a single "average buyer" response to the buy prompt. The persona token resolves this ambiguity. As shown in the funnel progression section of Table 3, Low-purchase tokens almost never trigger conversion actions, Medium tokens convert at moderate rates, and High tokens convert routinely, all under the *same* buy-oriented intent. This separation is not marginal: the mean purchase intensity score for High tokens is over $100\times$ that of Low tokens (Table 3), confirmed by a very large effect size (Cohen's $d = 2.80$) and highly significant statistical tests across all four measures ($p < 10^{-14}$).

Engagement provides a second, distinct form of behavioral transfer. The VQ-VAE engagement head is defined from total session duration over a buyer's history, but simulated sessions lack wall-clock time. We therefore evaluate engagement using a behavioral proxy: the total number of actions the agent takes per session. As shown in Table 3, the mean engagement score rises monotonically from Low to High tokens, indicating that the persona token shapes not only *whether* the agent converts, but also *how actively* it participates in the session; High-engagement tokens produce agents that browse more pages and persist longer, while Low-engagement tokens produce agents that act quickly and leave. All four statistical tests confirm this separation is significant among all three levels (Table 3), with the permutation test ($p = 0.003$) providing the strongest evidence: fewer than 3 in $1{,}000$ random label shuffles produced a gradient as strong as the observed one.

Exploration is the only auxiliary head that does not transfer reliably to agent behavior (Table 3). Mean scores are nearly flat across bins ($\sim$1.3 events), with negligible effect size (Cohen's $d = 0.10$) and no significant differences. Several factors may contribute to this weak separation. First, the underlying human signal is itself subtle: even among real buyers, Medium and High exploration differ by only 0.12 events per session, making the distinction difficult to recover from a single session. Second, the label distribution is heavily skewed, because our exploration metric aggregates product views, searches, and collection browsing over buyers who actively engaged (excluding bouncers), the VQ-VAE assigns most tokens to the High bin (72% High vs. 5% Low), potentially leaving insufficient contrast to learn the distinction. Third, our simulation setup introduces a floor effect; every agent receives a goal naming a product category, so even Low-exploration personas must perform at least one search or product view to pursue their objective, eliminating near-zero sessions and pushing all bins toward a similar baseline of $\sim$1.3 events.

Table 4: Instruction-following performance on 70 deterministic tasks (10 repetitions each).

Metric	SimPersona	GPT-OSS	GPT-4.1	GPT-5.4	Qwen
Avg. steps per run	14.9	11.4	12.3	14.9	17.7
Self-reported goal reached	77.7%	93.9%	76.5%	65.3%	54.9%
Funnel progression (required action reached)					
Cart → added to cart	**91.4%**	81.7%	59.9%	49.7%	62.6%
Checkout → reached checkout	**76.9%**	53.1%	38.9%	33.7%	45.3%
Purchase → reached checkout	**59.3%**	52.2%	28.9%	33.6%	38.8%

3.4 Does fine-tuning improve instruction following?

Beyond behavioral alignment, we ask whether *SimPersona* also improves the agent's ability to execute shopping tasks in live environments. To this aim, we compare *SimPersona* against two groups of baselines; the first consists of strong general-purpose models, GPT-OSS-120B, GPT-4.1, and GPT-5.4, which test whether a small, specialized model can match much larger instruction-followers. These baselines, however, do not separate the contribution of persona learning from that of domain fine-tuning. We therefore add a second baseline fine-tuned on the same shopping tasks but conditioned with a prompt-based persona: Qwen3.5-9B with personas constructed following Li et al. (2026). It receives the same domain fine-tuning data as *SimPersona* but differs in its persona mechanism (text profile vs. learned token). It also has visual access to the rendered page, which *SimPersona* lacks. Since this baseline also sees the rendered page while *SimPersona* does not, any advantage of *SimPersona* cannot come from better perception, and is instead attributable to its persona representation. For comparison, rather than using a fixed benchmark, we derive each task from a sampled buyer's historical session so that task difficulty reflects the underlying behavioral profile: low-engagement buyers receive short lookup-style tasks, while more engaged buyers receive longer flows. We uniformly sample 70 such tasks to cover persona-bin combinations, conversion outcomes, and complexity levels ranging from 3 navigation actions to more than 20 sequential steps (Table 12). Each task is repeated 10 times per model, and an independent LLM judge (GPT-5) determines whether the objective was completed.

Table 4 shows that SimPersona attains the highest actual task completion of every model on every funnel metric, with the advantage scaling with task difficulty. On simple cart tasks the strong general-purpose models remain competitive, but the gap widens on checkout tasks, which require chaining multi-page form navigation in a single session ($+23.8$pp over GPT-OSS), and persists on purchase tasks, the hardest category, demanding end-to-end transactions within 30 steps. The most direct comparison is against the fine-tuned Qwen baseline, which reaches checkout in only 45.3% of tasks versus SimPersona's 76.9% and trails on cart and purchase as well; because Qwen is fine-tuned on the same tasks, this gap cannot be attributed to fine-tuning alone and instead isolates the value of the learned persona representation over a prompt-based one.

4 Conclusion

We presented SimPersona, a framework that bridges user modeling and agent simulation by learning discrete persona tokens from raw clickstreams via a behavior-aware VQ-VAE and grounding them in an LLM through two-stage fine-tuning. On live storefronts unseen during training, SimPersona reproduces real conversion patterns, separates behavioral dimensions across personas, and outperforms a $8\times$ larger baseline; all with a single token per buyer and no per-store calibration.

Several directions remain open for future work; exploration behavior is less cleanly separated, suggesting richer signals are needed. DOM-based page representations omit visual cues that shape real decisions, motivating multimodal perception, and the current pipeline does not fully utilize the learned VQ-VAE embeddings after assignment; using them to initialize LLM token embeddings could reduce or eliminate the first grounding stage. Additionally, newly opened stores before clickstream history accumulates remain unsupported, as the framework requires buyer history.

AI USE STATEMENT

LLMs are used in this paper in different ways. First, GPT-5 synthesizes natural-language navigation goals from real buyer session histories, converting timestamped event sequences into structured task descriptions (Section 2.1.2). Second, Gemini 3 Flash executes these goals on live storefronts to generate agent trajectories in SFT-ready format (Section 2.1.2). Third, GPT-5 serves as an independent judge for filtering successful traces during data generation and for evaluating instruction-following performance (Sections 2.1.2 and 3.4). The fine-tuned Qwen3-14B (Yang et al., 2025) is the agent backbone that receives persona tokens and generates shopping actions; its architecture and two-stage training recipe are fully described in Section 2.3 and Appendix G. Additionally, Nano Banana 2 was used to generate schematic figures in Appendix. We also used LLMs for polishing the paper.

We have reviewed all AI-assisted work. Trajectories generated by Gemini 3 Flash were filtered via GPT-5 judge scoring and further validated by the authors to ensure format correctness and task alignment; LLM-judge evaluation outputs were cross-checked by human to verify reliability; all AI-generated figures were reviewed, revised as needed, and approved by at least two authors before inclusion. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We have read and adhere to the conference's code of ethics. We highlight the ethical considerations most relevant to this work below.

Respect Privacy: Our data pipeline is built on real buyer session histories, and we address privacy through data minimization and privacy-preserving design. The encoder ingests only behavioral and commercial aggregates and never ingests any demographic, identity, or geographic attribute; the buyer key is pseudonymous and is never used as a feature, so a code cannot recover an attribute the model never observes. The representation is also privacy-preserving by construction: each buyer is quantized to one of 256 discrete codes, all of which are in use, and every code is shared by a large cohort (minimum 530 buyers, median 2000 at evaluation). Each token therefore carries at most 7.75 of a possible 8 bits and always denotes a segment of several hundred people rather than an individual, so it cannot be used to re-identify a buyer. The natural-language navigation goals synthesized by GPT-5 describe task intent rather than user identity, and the agent trajectories generated on live storefronts contain no user-level personal data.

Be Fair and Take Action not to Discriminate: We designed our data and evaluation to represent buyers broadly rather than privileging any single population or behavior type. Training and evaluation span $10+$ countries across six continents, and cover the full range of shopping behaviors from heavy, high-intent purchasers to light browsers and skimmers, so the personas are not fit to a narrow, dominant user profile. Our evaluation is stratum-based, reporting performance across buyer segments which prevents strong average numbers from masking poor behavior on smaller groups. To ensure that rare but important groups (e.g., heavy purchasers) are not ignored during training, we apply inverse-frequency class weights (explain in Section 2.2) so underrepresented populations contribute proportionally to the objective. Finally, our evaluation population of 8.3M buyers spans a highly varied set of storefronts, from small independent shops to large merchants with $10M$–$100M$ USD GMV, so the learned personas reflect a diverse commercial ecosystem rather than a single tier of sellers.

Be Honest, Trustworthy, and Transparent: We are open about the capabilities and limitations of our proposed method, and these limitations are stated explicitly in the paper. In the results section, we explicitly note that exploration behavior is less cleanly separated, suggesting richer signals are needed (Section 3.3). The conclusion section motivates several future directions: our DOM-based page representations omit visual cues and motivate multimodal perception, the pipeline does not yet fully utilize the learned VQ-VAE embeddings after assignment, and newly opened stores remain unsupported until sufficient clickstream history accumulates. To ensure our reported gains are trustworthy and not artifacts of a single favorable setup, our claims are evaluated against multiple models and supported by statistical testing, so that results reflect genuine effects rather than bias toward any single aspect. All related and competing work is properly cited and described fairly

Reproducibility Statement

We have made substantial efforts to ensure that our results are reproducible. The behavior-aware VQ-VAE architecture, quantization mechanism, loss construction, and auxiliary head design are described in Sections 2.1 and 2.2. The complete set of hyperparameters, the data split, and the stratified sampling strategy are provided in Appendix C, and the input feature definitions are given in Appendix A. The two-stage fine-tuning framework, data format, goal-synthesis procedure, dataset sizes, and base model are detailed in Section 2.3, with the full training configuration in Appendix G and C. All metrics and their computation procedures are formally defined in Sections 3.2 and 3.3, the statistical tests and significance threshold are specified alongside them, and the instruction-following benchmark construction is described in Section 3.4. The mismatch baselines and the held-out evaluation storefronts are also specified there.

We release an open-source repository containing the complete data pipeline that converts raw event logs into buyer representations and agent-training traces, together with the VQ-VAE training code (source code is provided as supplementary material). The underlying clickstream data cannot be publicly released due to privacy; to mitigate this, we provide the full preprocessing and training code so that practitioners with access to analogous event logs can reproduce the pipeline end-to-end.

References

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Soda*, volume 7, pp. 1027–1035, 2007.

William Black, Alexander Manlove, Jack Pennington, Andrea Marchini, Ercument Ilhan, and Vilda Markeviciute. Trace: Transformer-based user representations from attributed clickstream event sequences. *arXiv preprint arXiv:2409.12972*, 2024.

Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

Yun-Shiuan Chuang, Krirk Nirunwiroj, Zach Studdiford, Agam Goyal, Vincent V Frigo, Sijia Yang, Dhavan V Shah, Junjie Hu, and Timothy T Rogers. Beyond demographics: Aligning role-playing llm-based agents using human belief networks. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14010–14026, 2024.

Jacob Cohen. *Statistical power analysis for the behavioral sciences*. routledge, 2013.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

Ronald Aylmer Fisher. *The design of experiments*. Oliver and Boyd, Edinburgh, 1935.

Simret Araya Gebreegziabher, Yukun Yang, Charles Chiang, Hojun Yoo, Chaoran Chen, Hyo Jin Do, Zahra Ashktorab, Werner Geyer, Diego Gómez-Zará, and Toby Jia-Jun Li. The behavioral fabric of llm-powered gui agents: Human values and interaction outcomes. In *Proceedings of the 31st International Conference on Intelligent User Interfaces*, pp. 909–927, 2026.

Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. In *International Conference on Learning Representations*, volume 2024, pp. 52690–52717, 2024.

Tobias Hatt and Stefan Feuerriegel. Detecting user exits from online behavior: A duration-dependent latent state model. *arXiv preprint arXiv:2208.03937*, 2022.

William H Kruskal and W Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

Han Li, Vibhor Malik, Zahra Zanjani Foumani, Alberto Castelo, Shuang Xie, Ailin Fan, Keat Yang Koay, Yuanzheng Zhu, Meysam Feghhi, Ronie Uliana, et al. Simgym: A framework for a/b test simulation in e-commerce with traffic-grounded vlm agents. *arXiv preprint arXiv:2605.19219*, 2026.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

Yuxuan Lu, Bingsheng Yao, Hansu Gu, Jing Huang, Zheshen 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 *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2025.

Yuxuan Lu, Ting-Yao Hsu, Hansu Gu, Limeng Cui, Yaochen Xie, William P Headden III, Bingsheng Yao, Akash Veeragouni, Jiapeng Liu, Sreyashi Nag, et al. Agent a/b: Automated and scalable a/b testing on live websites with interactive llm agents. In *Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2026a.

Yuxuan Lu, Jing Huang, Yan Han, Bingsheng Yao, Sisong Bei, Yaochen Xie, Yisi Sang, Qi He, and Dakuo Wang. Can llm agents simulate multi-turn human behavior? evidence from real online customer behavior data. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43961–43977, 2026b.

Marlene Lutz, Indira Sen, Georg Ahnert, Elisa Rogers, and Markus Strohmaier. The prompt makes the person (a): A systematic evaluation of sociodemographic persona prompting for large language models. In *EMNLP (Findings)*, pp. 23212–23237, 2025.

Saab Mansour, Leonardo Perelli, Lorenzo Mainetti, George Davidson, and Stefano D'Amato. Paars: Persona aligned agentic retail shoppers. In *Proceedings of the 1st workshop for research on agent language models (REALM 2025)*, pp. 143–159, 2025.

Yabo Ni, Dan Ou, Shichen Liu, Xiang Li, Wenwu Ou, Anxiang Zeng, and Luo Si. Perceive your users in depth: Learning universal user representations from multiple e-commerce tasks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 596–605, 2018.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Yilmazcan Ozyurt, Tobias Hatt, Ce Zhang, and Stefan Feuerriegel. A deep markov model for clickstream analytics in online shopping. In *Proceedings of the ACM Web Conference 2022*, pp. 3071–3081, 2022.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Joon Sung Park, Carolyn Q Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S Bernstein. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109*, 52, 2024.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. Character-llm: A trainable agent for role-playing. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13153–13187, 2023.

Yimin Shi, Yang Fei, Shiqi Zhang, Haixun Wang, and Xiaokui Xiao. You are what you bought: Generating customer personas for e-commerce applications. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1810–1819, 2025a.

Yunxiao Shi, Wujiang Xu, Zhang Zeqi, Xing Zi, Qiang Wu, and Min Xu. Personax: A recommendation agent-oriented user modeling framework for long behavior sequence. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 5764–5787, 2025b.

Aaron Van Den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Gang Wang, Xinyi Zhang, Shiliang Tang, Haitao Zheng, and Ben Y Zhao. Unsupervised clickstream clustering for user behavior analysis. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pp. 225–236, 2016.

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. *arXiv preprint arXiv:2510.07230*, 2025.

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, et al. Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43942–43960, 2026.

Bernard L Welch. The generalization of 'student's'problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Yimeng Zhang, Tian Wang, Jiri Gesi, Ziyi Wang, Yuxuan Lu, Jiacheng Lin, Simon Zhan, Vianne Gao, Ruochen Jiao, Junze Liu, et al. Shop-r1: Rewarding llms to simulate human behavior in online shopping via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8406–8423, 2026.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024.

A Data Pipeline

Figure 2 illustrates our end-to-end data pipeline described in Section 2.1. As mentioned in Section 2.1.1, we join the de-identified raw inputs including the event-level information with the product catalog, collection directory, and search-query logs to produce enriched session-level records that combine numerical behavioral signals (e.g., session counts, conversion rates, session durations) with semantic information (e.g., product titles, search queries, collection names) shown in Figure 3.

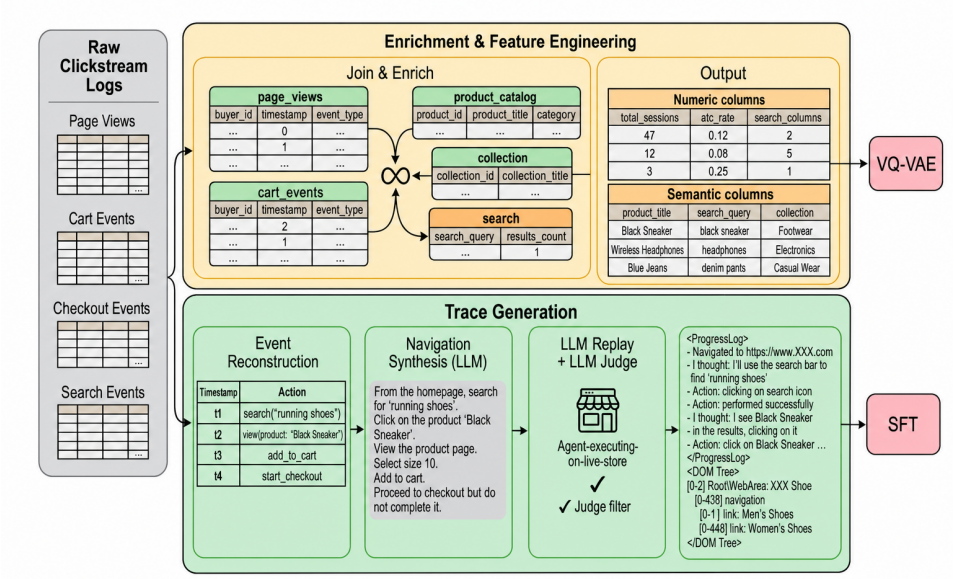


Figure 2: Data pipeline overview. A single enrichment pass over raw clickstream logs produces two outputs: (**top**) numeric and semantic buyer-level features for VQ-VAE persona discovery, and (**bottom**) executable multi-turn agent traces for SFT, generated by replaying real sessions on live storefronts.

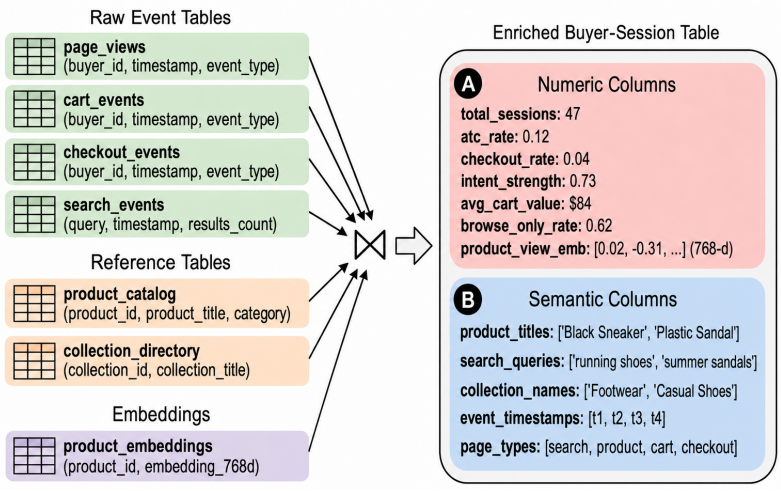


Figure 3: Data enrichment. Raw event-level tables are joined with the product catalog, collection directory, and product embeddings to produce an enriched buyer–session table containing (A) numeric behavioral features used for VQ-VAE persona discovery, and (B) semantic columns used for SFT trace generation.

Table 5: Feature normalization pipeline for the 403-dimensional VQ-VAE inputs.

Feature Group	**Dims**	**Transform**	**Rationale**
Exposure & Volume	8	$\log(1+x) \to$ rob-z	Right-skewed counts
Engagement Depth	2	$\log(1+x) \to$ rob-z	Cumulative totals with heavy tails
Funnel Rates	3	$\frac{n+\alpha}{N+\alpha+\beta} \to$ logit $\to$ rob-z	Smoothing stabilizes small N
Intent Strength	1	$\text{logit}_\epsilon \to$ rob-z	Bounded composite score
Dollar Values	2	$\log(1+x) \to$ rob-z	\$0 for non-buyers, heavy tail
Product Embeddings	3×128	dim-$z \to$ PCA $768\to128$	Heterogeneous dims; retains 85% var
Embedding Masks	3	binary (as-is)	Flags observed vs. zero-filled channels
Total	**403**		

From these enriched records, we extract the 16 scalar behavioral features described in Section 2.1.1. Among them, *Intent strength* is a composite score calculated by weighting actions according to their commercial commitment:

$$s_{\text{intent}} = \frac{w_{\text{atc}} \cdot n_{\text{atc}} + w_{\text{co}} \cdot n_{\text{co_start}} + w_{\text{pur}} \cdot n_{\text{purchase}}}{n_{\text{sessions}} + w_{\text{atc}} \cdot n_{\text{atc}} + w_{\text{co}} \cdot n_{\text{co_start}} + w_{\text{pur}} \cdot n_{\text{purchase}}} \tag{3}$$

where n_{atc}, $n_{\text{co_start}}$, and n_{purchase} count sessions containing an add-to-cart, checkout initiation, and completed purchase, respectively. The weights w_{atc}=3, w_{co}=5, w_{pur}=8 are monotonically increasing with funnel depth, following the same coprime design principle used for the auxiliary purchase head (Appendix E): w_{atc} and w_{pur} match the purchase composite weights, while w_{co}=5 is their midpoint, capturing checkout initiation as an intermediate commitment signal between add-to-cart and completed purchase. The denominator ensures $s_{\text{intent}} \in [0, 1)$: a pure window shopper scores 0, while a buyer who converts every session approaches $\sim$0.94. Because the 16 features span different scales and distributions, each group is normalized with a tailored transform (Table 5). Heavy-tailed counts and dollar values are log-transformed and then robust z-scored, computed as $(x - \tilde{x})/\text{IQR}$ with MAD $\times$ 1.4826 fallback when IQR=0. Bounded rates are first Bayesian-smoothed to replace the raw rate n/N with $\tilde{p} = (n+\alpha)/(N+\alpha+\beta)$ under a Laplace prior (α=β=1) to prevent extreme logit values from buyers with few sessions, then logit-transformed and robust z-scored. These 16

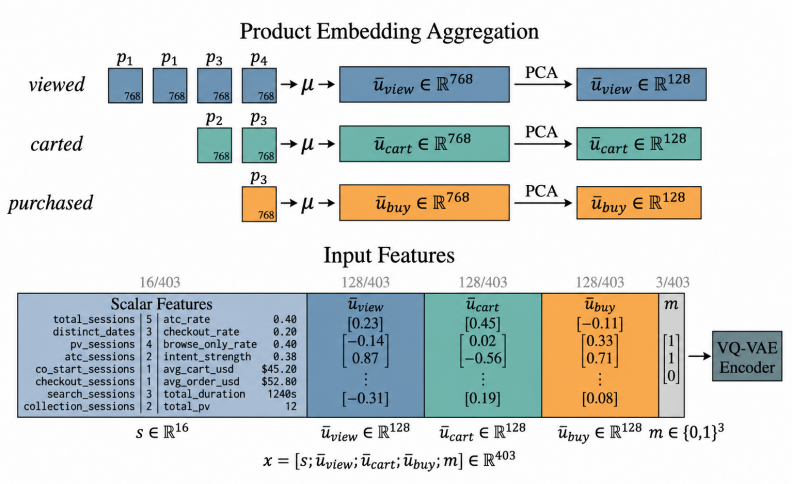


Figure 4: VQ-VAE input construction for a single buyer–shop pair. **top:** product embeddings are computed by averaging the 768-dimensional catalog vectors of all products a buyer viewed, carted, or purchased, then compressed to 128 dimensions via PCA. **Bottom:** the final 403-dimensional input concatenates 16 z-scored behavioral scalars, three 128-d product embeddings, and a 3-bit evidence mask indicating which product channels are present.

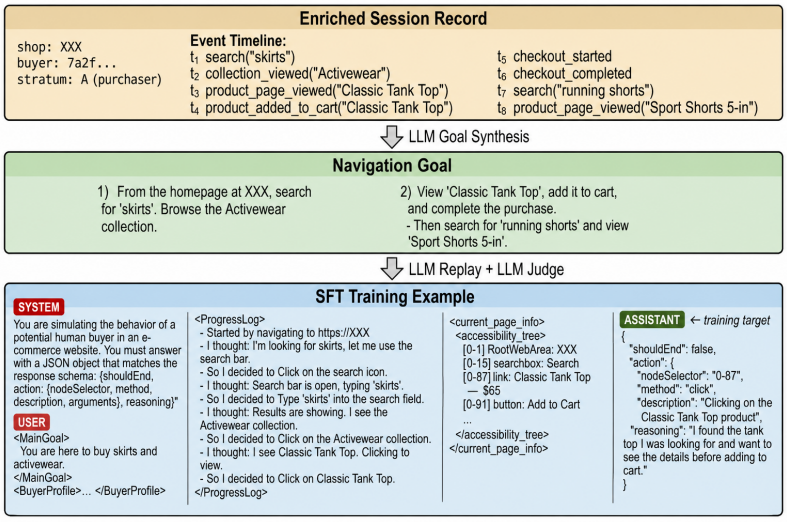


Figure 5: SFT trace generation from enriched clickstreams. **Row 1**: the enriched session record contains the buyer's event timeline with semantic labels (product names, collection titles, search queries) alongside metadata (shop ID, funnel stratum, persona token). **Row 2**: an LLM synthesizes the event sequence into a natural-language navigation goal, which is replayed on the live storefront by an agent and verified by an LLM judge. **Row 3**: the verified replay is converted into a multi-turn SFT example; system prompt defines the task format, user turn provides the navigation goal and DOM snapshot, and the assistant turn serves as the training target.

scalars are concatenated with three PCA-compressed product embeddings (3×128 dimensions) and three binary evidence masks to form the 403-dimensional VQ-VAE input illustrated in Figure 4.

Figure 5 illustrates the trace generation pipeline described in Section 2.1.2: a buyer's raw clickstream is first synthesized into a natural-language navigation goal, then an agent replays that goal on the live storefront, producing the multi-turn interaction traces used for SFT.

B Population Distribution Recovery

A distinctive property of our proposed method is that it recovers not only *which* behavioral types exist in the buyer population, but *how they are distributed* within each store, a capability that is structurally absent from prompt-based persona methods, where persona types are defined by manual specification and no mechanism exists to estimate the mixing proportions ($p_s(k)$) from observed traffic. In our method however, because each buyer b is mapped to a discrete token via a single encoder forward pass followed by nearest-entry quantization $q(\cdot)$, the empirical population distribution over persona tokens for store s is given by:

$$\hat{p}_s(k) \;=\; \frac{1}{|\mathcal{B}_s|} \sum_{b \in \mathcal{B}_s} \mathbf{1}[\, q(z_b) = k \,], \qquad k = 1, \ldots, K, \tag{4}$$

where $\mathcal{B}_s$ is the set of buyers observed at that store. This distribution is the sampling measure used to construct simulated buyer populations: agents are assigned persona tokens in proportion to $\hat{p}_s(k)$. As a result, store-level simulation statistics such as add-to-cart rate, checkout rate, and navigation depth depend not only on how each token behaves, but also on whether the learned token mixture matches the real buyer population. If high-intent tokens are overrepresented, simulated conversion will be inflated; if low-intent tokens dominate incorrectly, it will be suppressed. Recovering $\hat{p}_s$ is therefore a necessary condition for faithful population-level simulation.

We test this property on the same 42 held-out storefronts covering 8.37M buyers used in Section 3, using two complementary analyses. The first asks whether the learned token mixture can recover each store's funnel-stage composition. The second asks whether the same token mixture can reconstruct continuous store-level behavioral features. Together, these tests evaluate whether the codebook

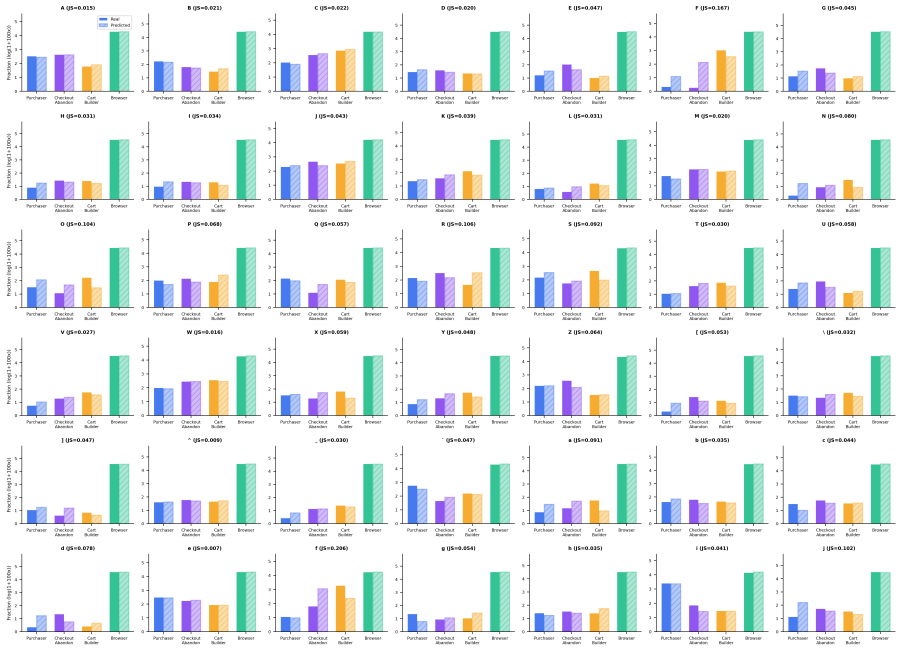


Figure 6: Stratum distribution recovery across all 42 storefronts. Solid bars show the real funnel-stage composition and hatched bars show the distribution predicted from each store's VQ-VAE token distribution. Bar heights are shown as $\log(1 + 100x)$, where x is the stratum fraction, to make low-frequency strata visible alongside the dominant browser segment. The predicted and real distributions remain closely matched under this visualization (mean JS divergence $= 0.054$), confirming that the learned codebook faithfully recovers the buyer population mix.

captures real population structure rather than merely clustering individual sessions. We first evaluate recovery of funnel strata. As explained in Section 2.1.1, each buyer is independently assigned to one of four rule-based strata from its historical behavior: Purchaser (A), Checkout Abandoner (B), Cart Builder (C), or Window Shopper (D). These labels are not used to train the VQ-VAE, which only sees continuous behavioral features, product embeddings, and evidence masks. This makes stratum recovery a stringent test: if the token distribution can reconstruct the stratum mix of a store, then the unsupervised codebook has recovered behaviorally meaningful population structure.

For each token k, we estimate its global stratum profile $P(\text{stratum} \mid k)$ from all buyers assigned to that token. We then predict the stratum distribution of store s by mixing these token profiles according to the store's token distribution:

$$\hat{P}_s(\text{stratum}) = \sum_{k=1}^{K} \hat{p}_s(k)\, P(\text{stratum} \mid k). \tag{5}$$

Intuitively, this asks if we know which persona tokens a store's buyers fall into, and we know what funnel stage each token typically represents, can we recover the store's real funnel-stage mix? We measure the agreement using the Jensen–Shannon (JS) divergence (Lin, 1991), a symmetric, bounded measure of similarity between two probability distributions. JS divergence ranges from 0 (identical distributions) to 1 (completely disjoint support); values below 0.05 indicate near-perfect agreement, while values above 0.15 suggest meaningful distributional differences.

Figure 6 compares the predicted and real stratum distributions across all evaluation storefronts, from the largest store with 3.1M buyers to the smallest with under $1{,}000$. The mean Jensen–Shannon divergence is 0.054 (median 0.045), with 36 of 42 stores below 0.10. The recovery is equally accurate on stores with over one million buyers, where even small distributional errors would affect hundreds of thousands of sampled agents, and on stores with fewer than $10{,}000$ buyers, where the token distribution is estimated from a smaller sample. Visually, the solid (real) and hatched (predicted) bars are nearly indistinguishable across stores, despite wide variation in buyer composition; from browser-dominated stores ($> 90\%$ stratum D) to stores with $30\%+$ purchasers. This level of agreement

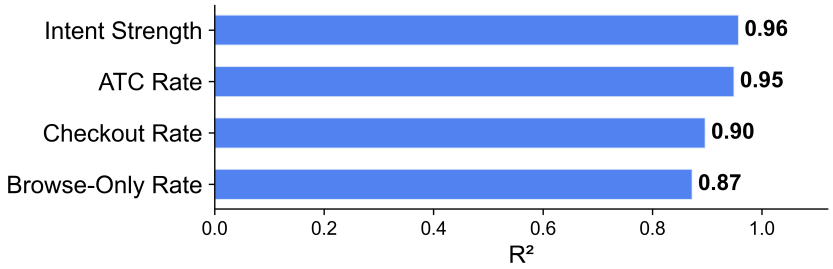


Figure 7: Store-level behavioral reconstruction from persona token distributions. The codebook achieves $R^2 \geq 0.87$ on the four features most critical to shopping simulation, confirming that it preserves behaviorally relevant population structure while remaining invariant to volume metrics.

between a hard rule-based labeling and an unsupervised discrete codebook trained on continuous features, confirms that the VQ-VAE codebook has learned behaviorally meaningful clusters whose population-level statistics faithfully reflect the real buyer distribution at scale.

We next test whether the same token distribution preserves continuous behavioral information beyond categorical strata. For each token k, we compute its behavioral centroid $\bar{\mathbf{x}}_k \in \mathbb{R}^d$; the mean raw feature vector of all buyers assigned to that token across all stores. For a given store s, we reconstruct its aggregate buyer profile by weighting these centroids by the store's token distribution:

$$\hat{\mathbf{x}}_s = \sum_{k=1}^{K} \hat{p}_s(k)\, \bar{\mathbf{x}}_k. \tag{6}$$

We compare $\hat{\mathbf{x}}_s$ to the true store-level mean $\mathbf{x}_s$ and report per-feature R^2 across all 42 stores (Figure 7). R^2 measures the fraction of store-to-store variance in each feature that is explained by the token-weighted prediction: a value of 0.96 means that 96% of the cross-store variation in that feature can be recovered from the token distribution alone, without access to any individual buyer data. The reconstruction is strongest on the behavioral dimensions most critical to simulation fidelity: intent strength ($R^2 = 0.96$), add-to-cart rate (0.95), checkout rate (0.90), and browse-only rate (0.87). These are precisely the features that determine whether a simulated agent browses passively, expresses purchase interest, or advances toward checkout, the core axes along which persona tokens must differentiate buyer behavior. Their consistently high R^2 (≥ 0.87) confirms that by knowing a store's token mix we can predict its real aggregate behavioral profile: if two stores differ substantially in how purchase-oriented their visitors are, the VQ-VAE token distribution captures that difference.

C VQ-VAE ARCHITECTURE AND TRAINING DETAILS

As explained in Section 2.2, codebook entries are initialized with k-means++ (Arthur & Vassilvitskii, 2007) over encoder outputs from a full pass through the training set. During training, entries are updated via exponential moving averages rather than gradient descent:

$$\mathbf{e}_k \leftarrow \gamma\, \mathbf{e}_k + (1-\gamma)\, \bar{\mathbf{z}}_k, \tag{7}$$

where $\bar{\mathbf{z}}_k$ is the mean of encoder outputs assigned to entry k in the current batch and $\gamma \in [0, 1)$ controls the memory of past assignments. To prevent codebook collapse (Razavi et al., 2019), we maintain an EMA of per-entry assignment counts $\hat{n}_k$ and declare an entry dead when

$$\hat{n}_k \;<\; \alpha \cdot \frac{1}{K} \sum_{j=1}^{K} \hat{n}_j, \tag{8}$$

i.e., when its usage falls below a fraction α of the mean. Dead entries are reinitialized from randomly sampled encoder outputs plus small Gaussian noise. This reset activates only after a warmup period so that usage statistics reflect a stable latent space.

The encoder, codebook, and decoder have layer sizes $[403 \to 256 \to 128 \to 96]$, $K = 256$ entries in $\mathbb{R}^{96}$, and a mirrored decoder, respectively. Reconstruction is down-weighted ($\lambda_r = 0.3$) to prioritize

behavioral partitioning over per-feature fidelity; remaining hyperparameters are $\beta = 0.75$, $\lambda_c = 0.15$, $\lambda_a = 0.5$, EMA decay $\gamma = 0.9$ (Equation (7)), and dead-code replacement with threshold $\alpha = 0.1$ every 50 steps after a 100-step warmup (Equation (8)). All values were selected through hyperparameter search optimizing for behavioral purity of the resulting codebook entries. We highlight that the training data is constructed by aggregating user behavior over the 4 months of November 2025 to February 2026.

D CONTRASTIVE AND AUXILIARY LOSS

The contrastive term $\mathcal{L}_{\text{contrast}}$ is an InfoNCE loss (Oord et al., 2018) computed over the encoder outputs $\mathbf{z}_e$:

$$\mathcal{L}_{\text{contrast}} = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \log \frac{\exp(\text{sim}(\mathbf{z}_e^{(i)}, \mathbf{z}_e^{(i+)})/\tau)}{\sum_{j \in \mathcal{B}} \exp(\text{sim}(\mathbf{z}_e^{(i)}, \mathbf{z}_e^{(j)})/\tau)}, \tag{9}$$

where $\text{sim}(\cdot, \cdot)$ denotes cosine similarity, τ is a temperature parameter, and $i+$ is the positive pair for anchor i. As explained in Section 2.2, we define positive pairs through 3 steps, and all non-self samples in the batch remain in the InfoNCE denominator (Equation (9)), so the model simultaneously pulls behaviorally aligned buyers together and pushes apart incompatible ones.

We set $\tau = 0.1$, $M = 10$, and $F = 3$, chosen via hyperparameter search over codebook purity optimization. The constraint $M > F$ is by design: the product filter retrieves a broad pool of peers who engage with similar merchandise, and the behavioral filter then narrows this pool to the F whose browsing *style* most closely matches the anchor buyer. If $F \geq M$, the behavioral stage would pass through every product-filtered candidate, collapsing the two-stage selection into a single filter and losing the ability to separate *what* buyers shop for from *how* they shop.

The auxiliary loss $\mathcal{L}_{\text{aux}}$ adds supervised pressure for the codebook to preserve behaviorally meaningful distinctions. Three single-layer linear heads operate on the quantized representation $\mathbf{z}_q$, each predicting a coarse three-way label (Low / Medium / High) for one behavioral axis:

$$\mathcal{L}_{\text{aux}} = \sum_{h \in \{\text{engage, explore, purchase}\}} \mathcal{L}_h, \tag{10}$$

where each $\mathcal{L}_h$ is a weighted cross-entropy loss:

$$\mathcal{L}_h = -\sum_{c=1}^{3} w_c^{(h)}\, y_c^{(h)} \log \hat{p}_c^{(h)}(\mathbf{z}_q), \qquad \hat{p}^{(h)}(\mathbf{z}_q) = \text{softmax}\big(\text{MLP}_h(\mathbf{z}_q)\big). \tag{11}$$

Here $y_c^{(h)} \in \{0, 1\}$ is the one-hot ground-truth bin for head h, and $\hat{p}_c^{(h)}$ is the predicted probability for bin c. Because some bins are much smaller than others (e.g., heavy purchasers are rare), we apply inverse-frequency class weights $w_c^{(h)} = N/(3 \cdot n_c^{(h)})$, where N is the training-set size and $n_c^{(h)}$ is the count in bin c. This ensures the loss does not ignore rare but behaviorally important groups.

E AUXILIARY HEAD WEIGHT SELECTION

The auxiliary purchase head (Sections 2.2 and D) supervises the codebook to preserve coarse purchase-intensity structure. Each buyer receives a three-level label (***Low / Medium / High***) derived from the composite score $s_{\text{pur}} = w_{\texttt{co}}\, n_{\texttt{checkout}} + w_{\texttt{atc}}\, n_{\texttt{atc}}$, where $n_{\texttt{checkout}}$ and $n_{\texttt{atc}}$ count sessions

Table 6: Purchase-intensity bins induced by $s_{\text{pur}} = 8\, n_{\texttt{checkout}} + 3\, n_{\texttt{atc}}$. The same partition is obtained for all 105 tested monotone weight pairs satisfying $w_{\texttt{co}} > w_{\texttt{atc}} > 0$.

Bin	**Semantics**	N	Checkout Rate	ATC Rate	Purchase Intensity
0	Browse only	17,453	.000	.000	.000
1	Interest only	16,472	.000	.892	.706
2	Purchase-committed	10,634	.533	.536	.776

containing a completed checkout and an add-to-cart, respectively. The only structural assumption is the funnel ordering $w_{\texttt{co}} > w_{\texttt{atc}} > 0$: checkout reflects stronger commitment than add-to-cart. The resulting integer scores are discretized into three bins by a count-based procedure that greedily merges adjacent values until three groups remain.

A key question is whether the auxiliary labels depend sensitively on the particular numerical choice of $(w_{\texttt{co}}, w_{\texttt{atc}})$. To test this, we swept all monotone integer pairs with $w_{\texttt{co}} \in [2, 15]$ and $w_{\texttt{atc}} \in [1, w_{\texttt{co}} - 1]$, for a total of 105 candidate configurations. On the $44{,}559$ VQ-VAE training buyers, every one of these configurations produced exactly the same three-way partition ($17{,}453/16{,}472/10{,}634$ buyers). In other words, the induced labels are not driven by any special tuning of the weights. Instead, the buyers already separate naturally into three coarse behavioral groups: those with no commercial activity, those with add-to-cart activity but no checkout behavior, and those with checkout-bearing sessions. As long as the monotone ordering $w_{\texttt{co}} > w_{\texttt{atc}}$ is respected, the exact ratio does not change this partition.

Since the partition is invariant, $(w_{\texttt{co}}, w_{\texttt{atc}}) = (8, 3)$ is chosen for interpretability rather than as a tuned parameter. The values are coprime ($\gcd(8, 3) = 1$), which maximizes the number of composite scores that uniquely decompose into their constituent events: a score of 11 can only mean one checkout plus one add-to-cart ($8 + 3$), and 6 can only mean two add-to-carts (3×2). A non-coprime pair like $(8, 2)$ would make scores like 16 ambiguous; two checkouts (8×2) or eight add-to-carts (2×8), conflating fundamentally different funnel behaviors. Beyond coprimality, the gap between 8 and 3 is large enough that the smallest checkout-containing score (8) strictly exceeds two add-to-carts (6), cleanly separating repeated interest from genuine funnel progression in the low-count regime that dominates real buyer data. The resulting half-integer bin edges $[1.5, 4.5]$ further guarantee that no buyer's integer-valued score falls on a boundary, making every bin assignment unambiguous.

F TWO-STAGE VS. SINGLE-STAGE TRAINING COMPARISON

A central design choice in SIMPERSONA is the two-stage SFT framework described in Section 2.3. In Stage 1, we freeze the LLM backbone and train only the 256 persona-token embeddings, and in Stage 2, we unfreeze the full model and jointly fine-tune all parameters. To test whether this decomposition is necessary, we compare against a single-stage baseline that trains all parameters jointly from initialization. Both variants use the same architecture, data, and optimization settings; the only difference is whether persona grounding is decoupled from action learning.

We evaluate both models on the same simulation setup used in Section 3.2. Each simulation run has one of four terminal outcomes: *goal reached*, *steps-limit*, *time-limit*, or *other*. A steps-limit outcome means that the agent continued to produce valid actions but did not complete the assigned task within the 30-step budget. A time-limit outcome means that the run exceeded the wall-clock timeout. The *other* category captures malformed or non-executable outputs, including invalid JSON and structurally broken action strings. Table 7 reports aggregate outcome counts, while Table 9 summarizes the distribution of per-shop error rates.

The aggregate goal-reached rates (83.5% vs. 82.8%) and overall error rates (16.5% vs. 17.2%) appear nearly identical, and a paired t-test on per-shop error rates confirms no significant difference in the mean (p-value $= 0.72$). However, focusing on the mean is misleading. The single-stage model's comparable average is an artifact of compensating extremes: it achieves very low error on some

Table 7: Two-stage vs. single-stage SFT across 42 storefronts. Goal reached = successful task completion. Steps-limit = step budget exhausted. Other = malformed or unparseable agent output.

	Two-Stage		**Single-Stage**	
Outcome	Count	%	Count	%
Goal reached	42,083	83.5	41,720	82.8
Steps-limit errors	6,625	13.1	6,333	12.6
Time-limit errors	598	1.2	7	0.0
Other errors	1,094	2.2	2,340	4.6

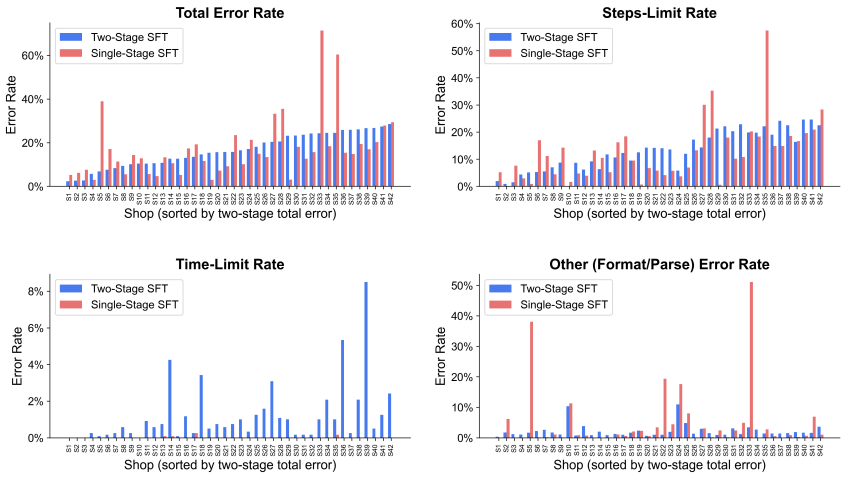


Figure 8: Per-shop error-rate comparison between two-stage and single-stage SFT (sorted by two-stage total error). The two-stage model keeps error bounded below 29% on every shop, while single-stage training produces catastrophic outliers exceeding 70%. The non-zero time-limit rate under two-stage training reflects deeper storefront engagement rather than inefficiency.

storefronts while collapsing catastrophically on others. The coefficient of variation, which measures how dispersed per-shop error rates are relative to their average, nearly doubles from 0.46 (two-stage) to 0.81 (single-stage). A coefficient of 0.81 indicates that the typical deviation from the mean is almost as large as the mean itself, meaning that the aggregate error rate is a poor predictor of what any individual storefront will experience. The per-shop error range tells the same story: under two-stage training, the gap between the best and worst shop is 26.3 pp (2.2%–28.5%), whereas under single-stage training it balloons to 68.4 pp (2.9%–71.3%), a 2.6× wider spread. In practical terms, this means a merchant deploying the single-stage model has no reliable guarantee of agent quality: while some storefronts enjoy error rates below 5%, others see more than half of all simulation episodes fail. As Table 9 shows, our two-stage training framework does not primarily improve the mean, it eliminates tail risk, compress the error distribution so that no storefront exceeds 30% error and every shop maintains at least 71.5% goal completion.

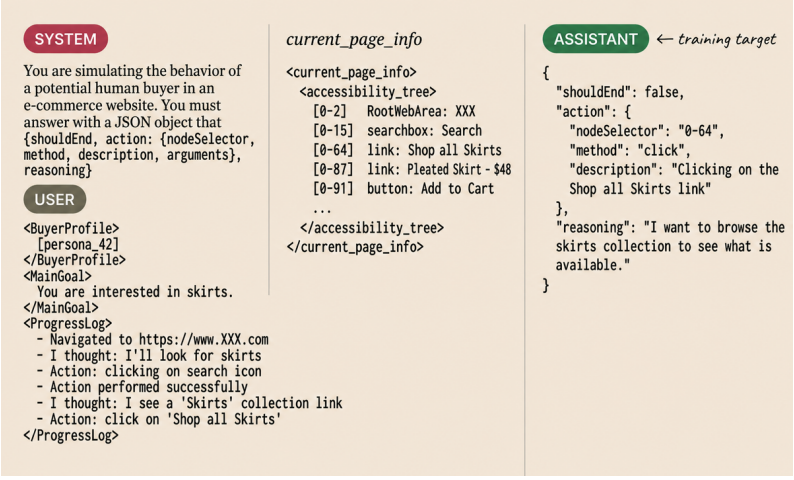
Figure 8 makes this distributional difference visually concrete. The two-stage error bars rise smoothly across shops and stay below 29%, while single-stage bars spike on several storefronts, revealing two distinct catastrophic failure modes. The first is *structural output collapse*: on the worst shop, 51.1% of episodes produce unparseable output, driving the total error to 71.3%; under two-stage training the same shop drops to 24.2% error with only 3.4% other errors; the entire 47 pp improvement comes from eliminating malformed outputs. The second mode is *degenerate navigation*: another shop reaches 60.3% error with 57.4% of episodes exhausting the step budget in repetitive loops; two-stage training reduces this to 24.5% by enabling more efficient action planning.

These worst cases reflect a systematic pattern. Total *other* errors drop 53% (from 2,340 to 1,094), and the per-shop other-error rate falls from 4.6% mean (std 10.0 pp, max 51.1%) to 2.2% mean (std 2.1 pp, max 10.9%). Across shops, the reduction in total error correlates strongly with the reduction in other errors (Pearson $r = 0.69$, $p < 10^{-6}$), confirming that structural output failures are the dominant source of variance eliminated by two-stage training. We attribute this to optimization interference: when 256 randomly initialized persona-token embeddings are updated jointly with all backbone parameters, the large, initially noisy gradients can destabilize the structured-generation pathways responsible for producing valid JSON actions before the tokens acquire meaningful signal. The two-stage framework avoids this by freezing the backbone in Stage 1, letting the embeddings converge to coherent representations without corrupting action generation, so that Stage 2 fine-tuning starts from an aligned rather than conflicting initialization.

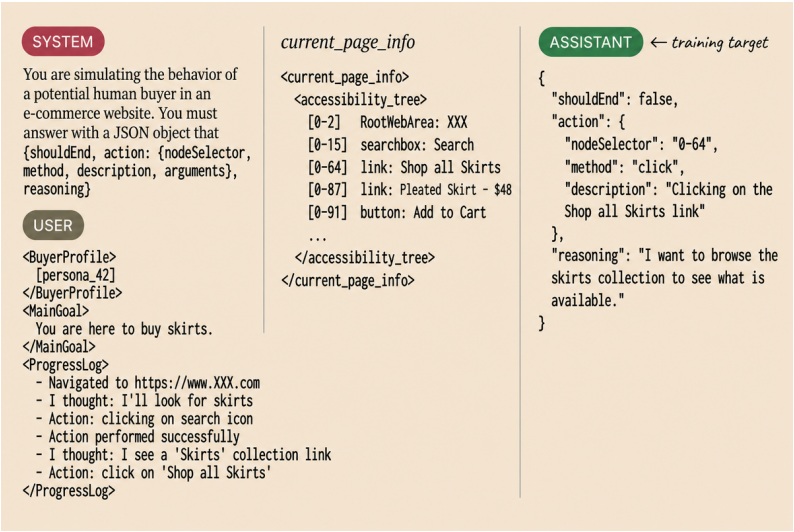
The remaining error-type panels reinforce this interpretation. Steps-limit rates are comparable in aggregate (13.1% vs. 12.6%) but far more variable under single-stage training (std 10.8 pp vs. 7.3 pp; one shop reaches 57.4%, while no two-stage shop exceeds 25%). The time-limit panel shows a

counterintuitive asymmetry: the two-stage model incurs 1.2% mean time-limit rate, while single-stage reports near-zero. The time-limit errors require the agent to actively render pages and interact with DOM elements long enough to exhaust the wall-clock budget. The single-stage model on its hardest shops fails before reaching this regime, either through unparseable output or step-count exhaustion. The two-stage model engages deeply enough that it occasionally runs out of time while making genuine progress, making the presence of time-limit errors a proxy for *behavioral depth*.

G TWO STAGE TRAINING DETAILS



(a) Stage 1: persona grounding (backbone frozen, only token embeddings updated).



(b) Stage 2: action-oriented fine-tuning (all parameters updated).

Figure 9: Two-stage persona-grounding SFT examples. Each training example consists of a system prompt specifying the output schema, a user turn containing the persona token, session goal, progress log (memory), and current page DOM, and an assistant turn with a structured JSON action and reasoning trace. Stage 1 teaches the model *what* each token means; Stage 2 teaches it *how* to act on that knowledge.

Both stages use Qwen3-14B as the base architecture, trained on 2 nodes $\times$ 8 NVIDIA H200 GPUs (16 GPUs total) with Fully Sharded Data Parallel (FSDP). In Stage 1, the backbone is fully frozen; only the 256 persona-token embeddings are trainable. We train for 1 epoch on approximately 34,000 multi-turn traces generated from 3,600 sessions across 39 shops (each session is on average 10

steps), with an effective batch size of 64 (1 per device $\times$ 16 GPUs $\times$ 4 gradient-accumulation steps), learning rate 1×10^{-5} with cosine annealing and 50 warmup steps, weight decay 0.01, and maximum sequence length 16,384 tokens. Loss is computed on assistant turns only. Stage 2 follows the same configuration except that all model parameters are unfrozen and trained. The 39 training shops are fully disjoint from the 42 evaluation storefronts used in Sections 3.2 to 3.4; no buyer appearing in the training set is included in any evaluation. Also, Figure 9 shows a schematic example of the data used for our two stage training described in Section 2.3.

H Robustness of the learned Persona tokens

To illustrate that the learned persona tokens reflect reproducible buyer structure rather than a particular VQ-VAE configuration, we examine their consistency across random initializations and codebook sizes by evaluating both the quality of the resulting groups and the agreement between independently learned partitions.

We first test the robustness and consistency of the learned persona tokens across random initializations. Training with five random seeds at $K{=}256$ yields stratum purity of $(84.5 \pm 0.5)\%$ and pairwise cosine similarity of 0.774 ± 0.009, indicating little variation in clustering quality. Beyond these aggregate scores, we examine whether independent runs recover the same buyer groups. Their partitions achieve an adjusted Rand index of 0.89 ± 0.04 and normalized mutual information of 0.90 ± 0.05, indicating strong agreement regardless of the numerical identifiers assigned to the tokens. The strong agreement, alongside consistent clustering quality, further supports the reproducibility of the discovered persona tokens across random initializations.

We next vary the vocabulary size over $K \in \{64, 128, 256, 512\}$ and compare VQ-VAE with k-means at each resolution (Table 8). VQ-VAE achieves higher stratum purity and lower engagement, exploration, and purchase incoherence at every tested K. Its purity increases from 80.6% to 88.9%, while purchase incoherence remains zero throughout and exploration incoherence reaches zero from $K{=}128$ onward. Engagement incoherence remains low, ranging from 0.2% to 1.5%, compared with 59.5%-78.2% for k-means. Thus, the advantage of the learned representation persists across an eightfold change in vocabulary size and is not specific to $K{=}256$.

The sweep also explains our choice of codebook size as a trade-off rather than a uniquely optimal setting. Increasing K produces finer partitions, so improvements in purity must be considered alongside within-persona coherence and vocabulary size. To examine product grounding, we additionally measure *product coherence*: the cosine similarity in product-embedding space between each persona's most-typical and most-peripheral members. This score increases from 0.476 at $K{=}128$ to 0.573 at $K{=}256$ and 0.690 at $K{=}512$. The larger vocabulary therefore provides more product-specific groups, but does not improve every behavioral criterion: engagement incoherence increases from 0.5% at $K{=}256$ to 1.0% at $K{=}512$, while exploration and purchase incoherence remain unchanged. We adopt $K{=}256$ as a compromise between product specificity, low behavioral incoherence, and a compact conditioning vocabulary.

Table 8: Sensitivity of persona discovery to codebook size.

K	Method	Purity ↑	Eng. inc. ↓	Exp. inc. ↓	Pur. inc. ↓	PW-Cos ↑	CH ↑
64	VQ-VAE	80.6%	1.5%	1.2%	0.0%	0.756	585.6
	k-means	57.7%	78.2%	29.4%	63.8%	0.686	160.9
128	VQ-VAE	83.5%	0.2%	0.0%	0.0%	0.776	375.8
	k-means	77.6%	71.8%	10.5%	18.4%	0.690	154.9
256	VQ-VAE	84.5%	0.5%	0.0%	0.0%	0.774	206.5
	k-means	78.9%	66.7%	4.6%	5.4%	0.722	173.8
512	VQ-VAE	88.9%	1.0%	0.0%	0.0%	0.798	121.6
	k-means	85.7%	59.5%	3.5%	3.2%	0.696	56.2

I CALINSKI–HARABASZ INDEX

The Calinski–Harabasz (CH) index (Caliński & Harabasz, 1974) is defined as

$$\mathrm{CH} = \frac{\mathrm{SS}_{\mathrm{between}} \,/\, (K-1)}{\mathrm{SS}_{\mathrm{within}} \,/\, (N-K)}, \tag{12}$$

where K is the number of clusters, 256 in our case, N is the total number of buyers, $\mathrm{SS}_{\mathrm{between}}$ is the sum of squared distances from each cluster centroid to the global centroid weighted by cluster size, and $\mathrm{SS}_{\mathrm{within}}$ is the sum of squared distances from each buyer to its assigned cluster centroid. Higher values indicate tighter, better-separated clusters.

J ADDITIONAL ANALYSES OF SIMPERSONA

This section presents two further analyses. We first examine how persona tokens shape agent behavior in Section J.1, and then compare *SimPersona* against a prompt-based persona method in Section J.2.

J.1 EFFECTS OF PERSONA TOKENS UNDER NEUTRAL INTENTS

To further test the effect of persona tokens on agent behavior, we compare simulations with persona-token conditioning against simulations without. To ensure that the agent is not guided by intent-specific information, both conditions use a neutral, generic intent ("you are interested in product X"). The rest of the experimental setup is identical to Section 3.2.

Our analysis reveals that removing persona tokens triples the rate of simulation crashes: $7{,}862$ (15.6%) without tokens versus $2{,}650$ (5.3%) with tokens (Table 10). The gap is dominated by a single error category: `StagehandTargetClosedError`, a browser-level failure in which the page session terminates mid-interaction. This error occurs $4.6\times$ more frequently without persona tokens ($6{,}579$ vs. $1{,}430$). To verify that this gap reflects a difference in navigation behavior rather than a difference in output formatting, we use context-length errors as a control: these errors are triggered when the model's prompt or response exceeds the token limit, a failure mode that depends entirely on output length and is independent of how the agent navigates the storefront. Context-length errors are virtually identical across conditions ($1{,}119$ vs. $1{,}141$; ratio 0.98), confirming that persona tokens do not change *what* the agents generate but rather *how* they interact with the page. The crash reduction is not confined to a few outlier shops: 38 of 42 storefronts exhibit a higher crash rate without tokens, with per-shop differences exceeding 40 pp on the most affected shops (Figure 10, left).

`StagehandTargetClosedError` occurs when the agent's actions destabilize the browser session, for example, by clicking elements that no longer exist after a page transition, issuing navigation commands faster than the renderer can process, or interacting with the DOM during partial page loads. These failures are deterministic consequences of the agent's action sequence, not stochastic infrastructure events. In our simulations, both conditions receive only a neutral intent ("you are interested in product X") that specifies a product but provides no behavioral signal. Without a persona token, the agent lacks any prior over trajectory structure and it must select actions conditioned

Table 9: Distributional statistics of per-shop error rate across 42 storefronts.

Statistic	**Two-Stage**	**Single-Stage**
Std of error rate	7.6pp	13.9pp
Min error rate	2.2%	2.9%
Max error rate	28.5%	71.3%
Shops <10% error	8	13
Shops >30% error	0	5
Shops >50% error	0	2
Min goal-reached rate	71.5%	28.7%
Shops <50% goal reached	0	2
Shops <70% goal reached	0	5

Table 10: Persona token ablation under neutral intents. Persona tokens reduce browser crashes by 66% and produce deeper storefront engagement across all behavioral metrics.

Metric	With Token	No Token	Ratio
Episode outcomes (all simulations)			
Simulation crashes	2,650 (5.3%)	7,862 (15.6%)	0.34×
StagehandTargetClosed	1,430	6,579	0.22×
Context length	1,141	1,119	1.02×
Goal reached	39,664 (78.7%)	37,942 (75.3%)	1.05×
Behavioral engagement (all simulations)			
Add-to-cart events	16,132	13,698	1.18×
Checkout reached	7,823	6,664	1.17×
Navigation depth (succeeded simulations only)			
Avg actions / session	11.1	9.1	1.23×
Avg pages visited	3.9	3.6	1.10×
Avg products viewed	1.39	1.29	1.08×
Crash rate consistency			
Shops w/ higher crash rate	3/42	38/42	—

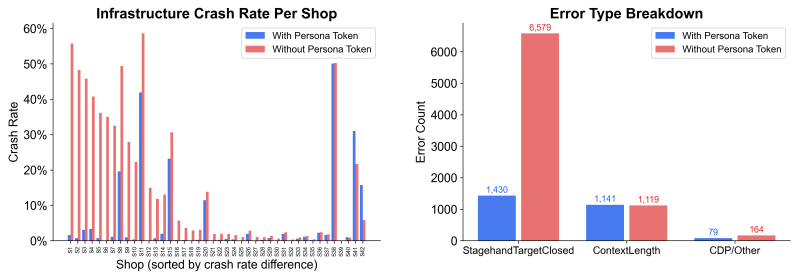


Figure 10: Persona token ablation under neutral intents. **Left**: per-shop infrastructure crash rate (sorted by difference); 38 of 42 shops crash more frequently without persona tokens. **Right**: error-type breakdown showing that browser crashes (`StagehandTargetClosed`) are 4.6× more frequent without tokens, while context-length errors remain identical (≈1,130), isolating navigation behavior as the mechanism.

solely on page observations and a generic goal. This yields a high-entropy action distribution that produces erratic navigation like frequent page switches, interactions with transient UI elements, and action–observation desynchronization during asynchronous page loads. These are precisely the trajectory patterns that maximize the probability of acting on stale page state. A persona token provides the missing behavioral signal. Trained on real buyer navigation traces, each token encodes a coherent browsing style that constrains the agent's action distribution toward structured, sequential trajectories. Because storefronts are designed for human navigation, these human-like trajectories are inherently more compatible with the browser environment, explaining why the token reduces `StagehandTargetClosed` crashes by 78% without affecting context-length errors at all.

Behavioral engagement metrics provide further evidence that persona tokens produce deeper, more purposeful navigation rather than merely preventing crashes (Table 10). Across all simulations, persona-conditioned agents add items to cart 18% more often (16,132 vs. 13,698) and reach checkout 17% more frequently (7,823 vs. 6,664). On clean (non-crash) simulations where both conditions complete the session without infrastructure failure, persona-conditioned agents still perform 23% more actions per session (11.1 vs. 9.1), visit 10% more pages (3.9 vs. 3.6), and view 8% more products (1.39 vs. 1.29). The persona token therefore provides a genuine behavioral prior that shapes navigation even in the absence of persona-specific intent: the agent browses more deeply, interacts with more product pages, and progresses further through the purchase funnel, producing sessions that more closely resemble real buyer behavior.

J.2 COMPARISON TO A PROMPT-BASED PERSONA BASELINE

In this section, we compare *SimPersona* against a prompt-based baseline, as prompt-based methods are among the most widely used approaches for persona learning in the literature. Such methods, however, are incompatible with the evaluation used in Section 3.2. Our primary metric measures whether the agents assigned to a persona reproduce the conversion rates of the real buyers who share it, computed per (shop, token) pair and stratified by funnel stratum so that dominant, largely non-converting populations do not wash out rarer but important buyer types. This requires many buyers to map to a shared persona and to carry funnel-stratum labels, and prompt-based methods provide neither: each buyer instead receives a unique profile, typically mined from individual browsing or search histories, so no two buyers share a persona and no stratum structure is exposed against which alignment can be measured. Comparing the two methods therefore demands a separate, coarser metric, which we introduce below.

Most existing prompt-based methods are inapplicable to our *SimPersona* comparison, for either representational or environmental reasons. PAARS (Mansour et al., 2025), for instance, operates in a substantially narrower setting: a single English, text-only marketplace in which each episode begins from a search query and the agent's action space is limited to search, view, and cart. Our evaluation instead spans diverse verticals, languages, and interfaces, where many sessions never begin with a search. We therefore adopt SimGym (Li et al., 2026) as our prompt-based baseline, as it is among the few agentic shopping simulators whose agents browse live storefronts freely rather than following a scripted, search-first trajectory, directly matching our open setting. SimGym constructs a persona in three stages: it clusters sessions on behavioral features, prompts an LLM to extract product preferences from shop metadata and cluster-level summaries, and then prompts the LLM on aggregated buyer histories to produce a structured archetype scoring traits such as price sensitivity, exploration depth, and premium, performance, and ethical focus. This archetype, together with the extracted product preferences and a shopping intent, conditions the browser agent.

As mentioned earlier, the coarser metric this comparison requires is alignment at the storefront level, the only granularity at which the two methods can be compared. Because SimGym assigns each buyer a distinct text profile, its outputs cannot be grouped into shared persona tokens or funnel strata: it neither exposes the shared-persona structure our token-level metric requires nor draws on the buyer-level signals from which our strata are derived. Imposing either grouping would mean importing SimPersona's representation into SimGym, thereby evaluating a hybrid of the two methods rather than SimGym itself. At the storefront level, no such labels are needed: for each shop, we pool all agent runs into a single simulated conversion rate and all real buyers into a single observed rate, and measure their alignment. The SimGym agents use Gemini 3 Flash in both text-only and multimodal configurations, allowing us to isolate whether visual access to the rendered page is what a prompt-based agent requires.

Table 11: Shop-level conversion alignment. Each row reports pooled simulated (sim) versus observed (obs) ATC and purchase rates; ARA is computed within each shop and then averaged across shops.

Method	ATC (sim / obs)	Purchase (sim / obs)	ARA ↑
SimGym (multimodal)	0.568 / 0.128	0.434/ 0.078	0.578
SimGym (text-only)	0.517 / 0.128	0.408 / 0.078	0.603
SimPersona	0.128 / 0.128	0.072 / 0.078	0.927

Table 11 reveals a substantial gap. Both prompt-based variants markedly over-convert, simulating add-to-cart rates of $0.52 - 0.57$ and purchase rates of $0.41 - 0.43$, whereas the real buyers convert at only 0.128 and $.078$, respectively. Visual access does not close this gap; the multimodal and text-only variants behave almost identically, indicating that the shortfall is not attributable to what the model can perceive on the page. *SimPersona*, by contrast, tracks the real cohort almost exactly. We note that the *SimPersona* ARA here is higher than the stratified ARA we report in Table 2 (0.787). These are consistent. The shop-level number is pooled across all buyers, so it is dominated by the largest stratum, stratum D, who rarely convert but make up most traffic, which is also the stratum our model

aligns most accurately (Table 2, D ARA 0.954). Pooling therefore weights toward the largest group, lifting the number. Avoiding this dominance is the reason for our stratified evaluation metric proposed in Section 3.2.

K INSTRUCTION-FOLLOWING TASK DIVERSITY

To evaluate instruction-following quality (Section 3.4), we construct a deterministic benchmark of navigation tasks with varying complexity. Each task specifies a concrete sequence of actions the agent must execute on a live storefront e.g., searching for specific products, viewing product pages, adding items to cart, and proceeding through checkout. Table 12 shows representative examples ordered by complexity: the simplest tasks require 3 actions (search, view, cart), while the most complex involve 27 actions spanning multiple search queries, product comparisons, cart modifications, and a completed purchase.

Table 12: Representative tasks from the deterministic benchmark, ordered by complexity. Bins denote (Engagement / Exploration / Purchase); *Nav.* counts distinct navigation actions in the instruction.

Bins (E/X/P)	**Nav.**	**Task description**
0 / 2 / 1	3	Search for one product, view it, add to cart, end session
0 / 1 / 1	5	Search for one product, view it, add to cart, abandon checkout
1 / 2 / 0	7	Search two products, view each, add one to cart, abandon checkout
2 / 2 / 1	10	Search 3 products, view each, add one, revisit pages, abandon
2 / 2 / 2	16	View 4 products, add one to cart, view 2 more, add another, checkout
2 / 2 / 2	27	Search and view 6 products, add 3 to cart, revisit 2, purchase