APeB: Benchmarking Personalization Ability of Large Language Model Agents

Garry Yang[1,*], Zizhe Chen[1,*], Xinru Chen[2], Yongqiang Chen[1]
Jianxiang Wang[2], Deyu Zou[1], Linyi Ding[2], Jialiang Wu[2]
Yunzhong He[2], Yu Gong[2], James Cheng[1,†], Huaixiao Tou[2,†]
[1]The Chinese University of Hong Kong [2]ByteDance
hcyang@cse.cuhk.edu.hk

Abstract

LLM-powered agents struggle with personalization when users issue raw, underspecified queries. In this setting, agents must infer latent intent, extract preferences from noisy interaction histories, and select among competing alternatives. Existing benchmarks rarely test this capability, as they often rely on user-refined queries or simplified histories. We introduce personalized product search (PPS), a testbed for agentic personalization under raw queries and diverse histories. We construct **A**gent **Pe**rsonalized **B**enchmark (**APeB**) from action logs, pairing underspecified intents with rich histories and user-viewed candidate items. Evaluating state-of-the-art LLMs with multi-step agent workflows, we find that models handle explicit queries well but struggle with early-stage queries requiring intent and preference discovery. Rubric analysis attributes this gap mainly to ineffective history use. A simple history-aware query-refinement pipeline, VQRA, yields consistent gains, highlighting the need for dedicated history-utilization modules in personalized agents.

1 Introduction

Large Language Model (LLM)-powered agents extend foundation models with additional modules for memory, planning, and execution, enabling multi-step, goal-directed reasoning over long contexts (Peng et al., 2025; Wang et al., 2024; Zhang et al., 2025d; Hu et al., 2026). As shown in Figure 1, a central challenge for LLM-powered agents is *personalization*: user objectives are not explicitly specified; preferences are implicit and partially observed through histories; and correctness is user-centric rather than globally optimal (Russell, 2021). In practice, users implicitly reason over their own past experience, translate vague intent into specific criteria, and discriminate between plausible options

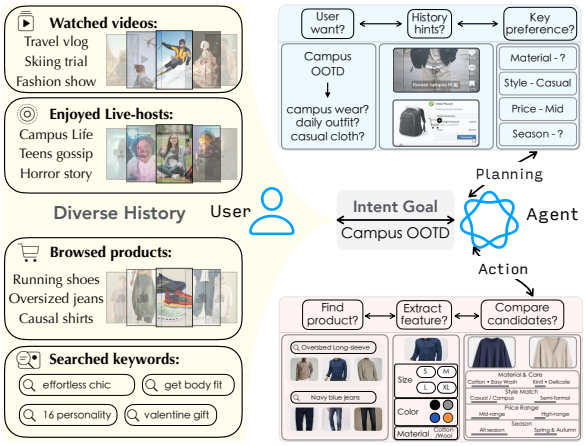


Figure 1: Personalization challenge for agents: infer intent and preferences from diverse history to make a preference-aligned decision.

(Downey et al., 2008; Punj and Moore, 2009; Liu et al., 2023a). Personalized decision-making requires agents to align their planning and actions with a user's intent and preferences (Zhang et al., 2025d). To study this capability when user goals are not fully specified, we ask:

How can we benchmark the agentic personalization capability of LLMs?

To our knowledge, existing benchmarks individually address one or two of these axes but rarely combine noisy heterogeneous histories, vague-to-refined search trajectories, and closely competing candidates. *Recommendation benchmarks* cast personalization as next action prediction from historical signals, emphasizing preference deduction over objective understanding (Zhu et al., 2018; Liu et al., 2016). *Rationale prediction benchmarks* evaluate localized reasoning, focusing on generating intermediate justifications in isolation rather than reaching a user-aligned outcome (Wang et al., 2025c; Yang et al., 2025b). *Multi-turn Memory benchmarks* model personalization over largely homogeneous conversations, evaluating relevant infor-

[*]Equal contribution.
[†]Corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 6647–6678
October 24-29, 2026 ©2026 Association for Computational Linguistics

Category	Persona	User Goal	Candidate Set	Evaluation
Generic Shopping	–	Explicit, multi-constraint	Web-scale / open-ended	Multi-step web shopping
Recommendation	Homogeneous, low-semantic	–	Open-ended	Next-item prediction
Personalized Product Search	Homogeneous, low-semantic	Explicit, user-refined	Randomly sampled	Product finding
Rationale Prediction	Synthetic, surveys	–	Random / open-ended	Localized reasoning
Multi-turn Memory	Homogeneous, synthetic	Explicit, instruction-driven	–	Memory-aware generation
APeB	**Heterogeneous, latent**	**Implicit, ambiguous**	**Few closely competing**	**Personalized goal completion**

Table 1: Comparison of existing benchmarks by whether and how they capture core personalization challenges.

mation extraction instead of intent inference under ambiguity (Hu et al., 2026; Huang et al., 2025). In all, these benchmarks sidestep the need to jointly reason over noisy histories, unrefined user queries, and closely competing candidates. As a result, they do not directly evaluate whether agentic pipelines (e.g., ReAct) contribute to intent inference and preference-aligned actions (Du et al., 2025).

Motivated by these limitations, we construct **APeB** (**A**gent **Pe**rsonalized **B**enchmark) to evaluate agents' personalization ability through the *Personalized Product Search* (**PPS**) task. PPS frames personalization as a user-specific choice among competing products conditioned on a shopping intent and long-term behavioral preference (Robertson and Zaragoza, 2009; Duan and Zhai, 2015). On online platforms, user behavior is noisy across domains, shopping intent is often vague and early-stage, and purchases occur only after comparing similar products (Liu et al., 2025; Reddy et al., 2022; Downey et al., 2008; Rendle et al., 2012). However, prior *PPS benchmarks* rely on sanitized histories, clarified queries, and easily separable candidates (Liu et al., 2023a; Hou et al., 2026), thereby failing to evaluate personalization under ambiguous, reasoning-intensive conditions. The preference-aligned purchase also contrasts with *generic shopping benchmarks* that focus on isolated sub-tasks (e.g., product matching) without grounding decisions in user-specific intents and histories (Jin et al., 2024; Yao et al., 2022). Table 1 summarizes this comparison.

In turn, **APeB** contains intricate cases drawn from a content platform with shopping functionality. It includes 5,648 cases from 10+ categories sampled over a 30-day window, with up to 60 days of history preceding the first query of each retained session (Section 3.1.1). Users primarily consume videos and livestreams, yet can seamlessly transition to product search, browsing, and purchase within the same platform. This end-to-end flow yields rich, heterogeneous interaction histories that capture real-world personalization challenges (see Appendix A.1). We disentangle user behavior into a long-term history period and a personalized shopping session, delimited by the timestamp of a user-issued search (Section 3.1.1). Sessions are retained only if the search leads to a query-aligned purchase following reasoning-intensive browsing (Section 3.1.2). The history period comprises diverse interactions with products and contextual content (e.g., videos and livestreams), providing long-term personalization signals (Section 3.2.1). The shopping session contains user searches spanning vanilla intent to refined keywords (Section 3.2.2), together with a set of heavily viewed competitive candidates (Section 3.2.3) and a final observed purchase, enabling intent evaluation and verifiable user-centric assessment. We evaluate whether personalization agents can first infer a user's latent preferences from vague intents and history hints (*Planning*) and select among close alternatives (*Action*) on the user's behalf (the pipeline in Figure 1).

We test latest LLMs with agentic workflows on APeB. Evaluation spans intent and refined queries with random and hard candidates, and includes comparison against a PPS model UniSAR (Shi et al., 2024). Results show that (1) with either well-specified refined queries or random candidate sets, LLMs substantially outperform UniSAR, and ReAct workflows further improve performance; (2) when vague, early-stage intent queries are paired with hard candidate sets, LLMs fall to the level of trained UniSAR, with limited gains from ReAct workflows; (3) rubric analysis shows that intent queries expose weak intent and history attribution, with ReAct traces showing lower reasoning coherence and history-retrieval utility; (4) we find that a simple history-grounded Vague Query Refinement Agent (VQRA) pipeline improves intent inference and Hit@1 over single-prompt under intent. Overall, APeB provides a behavior-grounded testbed for evaluating how agentic systems use long-term histories to infer and fulfill underspecified user goals.

2 Related Work

LLM-powered personalization agents. LLM-powered personalization agents use user models and long-term memories to condition current decisions on past interactions (Wang et al., 2024; Packer et al., 2024; Wang et al., 2025b). User representations range from explicit profiles to latent personas (Salemi et al., 2024; Sun et al., 2025; Magister et al., 2024; Richardson et al., 2023; Huang et al., 2025); memory systems store and retrieve task-relevant evidence from past interactions (Wang et al., 2023); and personalized search methods use history-derived preference signals for query-aware refinement and product reranking (Peng et al., 2026). Such systems must identify which interactions remain relevant as preferences and intent evolve, then ground the resulting evidence in fine-grained candidate comparison. Resolving vague intents requires coordinating history retrieval, preference inference, query formulation, and candidate comparison (Wang et al., 2025b; Zhang et al., 2025a; Yao et al., 2023). Multi-step LLM agents can struggle with intermediate decisions (Zou et al., 2026b,a; Chen et al., 2026); in personalization, this may surface as failures to identify or apply task-relevant preferences.

Generic e-commerce benchmarks. E-commerce benchmarks cover product search, web navigation, and shopping knowledge. Evaluation spans retrieval accuracy, task success, generation quality, and rubric scores (Reddy et al., 2022; Peng et al., 2024). Search datasets center on direct queries or synthetically constructed multi-constraint requests (Hou et al., 2026; Liu et al., 2023a; Lyu et al., 2025; Tou et al., 2025). Web-agent settings add interactive navigation and product comparison, often through synthetic or annotator-authored goals (Zhou et al., 2024; Yao et al., 2022; Deng et al., 2023; Wang et al., 2025a). Knowledge benchmarks decompose shopping into QA-style concept and attribute reasoning (Peng et al., 2024; Xie et al., 2025; Jin et al., 2024). These settings primarily evaluate explicit task completion or isolated reasoning components.

Behavior datasets for personalization. Recommendation datasets learn from user–item histories through next-item or rating prediction (Jin et al., 2023; Liu et al., 2016; Zhu et al., 2018; Ben-Shimon et al., 2015). PPS further combines preference and query intent, progressing from user–query–item representations to query-aware and LLM-based retrieval (Ai et al., 2017; Fan et al., 2022; Cheng et al., 2022; Si et al., 2023; Shi et al., 2024; Zhang et al., 2025b). Semantic and LLM-based variants improve textual generalization, while jointly using long histories, vague queries, and fine-grained item attributes remains challenging (Chen et al., 2025; He et al., 2022; Lin et al., 2024; Liu et al., 2023b). Existing datasets commonly pair observed targets with unobserved-item negatives (Liu et al., 2023a; Hou et al., 2026), while memory benchmarks test retention of user-specific evidence across synthetic dialogues or sessions (Huang et al., 2025; Hu et al., 2026; Shang et al., 2025; Yan et al., 2025). APeB joins heterogeneous histories and user-issued vague queries with viewed hard candidates and purchase-linked evaluation, connecting history-grounded reasoning to the user's realized choice.

3 APeB Construction

We introduce **APeB**, a benchmark designed for evaluating **agentic personalization capability** on the PPS task. To retain cases in which the personalized decisions are time-consuming and interaction-heavy (Section 3.1.2), we collect **non-trivial user shopping sessions** with their heterogeneous histories mined from holistic user behavior logs (Appendix A). As shown in Figure 2, a *session* is defined as a time-bounded search-to-order window that begins with a search and ends with a purchase (Section 3.1.1). Non-trivial sessions involve query reformulation and extensive product comparison before an intent-aligned order (Section 3.1.2), reflecting how user history shapes query formulation, exploration, and decision-making (Shang et al., 2025; Zuo et al., 2022). Together, these components are designed to reduce shortcuts based solely on surface-level semantic matching and increase the need for joint reasoning over historical preference signals, query intent, and fine-grained candidate differences. For filtered sessions, we construct three benchmark components: (i) **Heterogeneous histories**–preference signals (Figure 2 Left, Section 3.2.1); (ii) **User-issued intent (vague, early-stage) and refined (well-specified, late-stage) queries**–different levels of query specificity (Figure 2 Right, Section 3.2.2); and (iii) **Hard candidate set**–products the user considered, together with the **observed purchased item** (Figure 2 Right, Section 3.2.3).

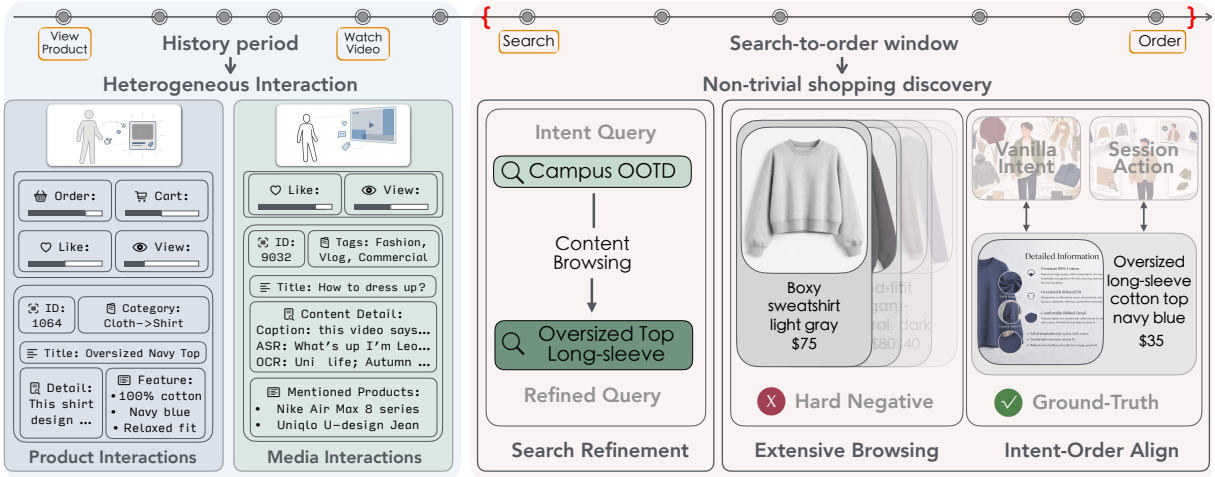


Figure 2: Construction Pipeline of **APeB**: from behavioral logs to benchmark components: curated heterogeneous interactions, non-trivial shopping session discovery (search refinement, extensive browsing, and intent–order alignment), and intent/refined query–candidate pair identification. APeB targets challenging personalization grounded in holistic user behavior.

3.1 Non-trivial Shopping Session Discovery

We identify non-trivial shopping sessions over a 30-day sampling window. On each day, we first retain users with at least five product or media interactions on every day of the preceding 30 days (Appendix A.2). Following the long–short period division paradigm (Liu et al., 2016), we then extract **search-to-order sessions** by anchoring boundaries at search timestamps. From these, we further select **non-trivial cases** characterized by query evolution, extensive content browsing, and cross-product comparison prior to a personalized purchase decision.

3.1.1 Search-to-Order Session Disentangle

We disentangle each user's interaction sequence by the search action timestamp ('{' in Figure 2). For a user u, let $\mathcal{I}_u = \{(x_u^k, t_u^k)\}_{k=1}^{N_u}$ denote the chronologically ordered interaction log, where x_u^k is an action and t_u^k its timestamp. We consider an initial search action $x_u^m = \text{query}(q)$ at time t_u^m and a final purchase action $x_u^n = \text{order}(p)$ at time t_u^n. A *search-to-order session* is identified if the purchase occurs within a fixed time window $\Delta = 30$ minutes after the search: $t_u^n - t_u^m \leq \Delta$. Users without any such session are removed. After applying the session-level criteria below, if a user has multiple qualifying sessions across the 30-day sampling window, we retain the earliest one. For each retained session, the search time t_u^m and the purchase time t_u^n define the disentanglement boundary. We set the pre-session history horizon to $\Delta_H = 60$ days, measured backward from the initial search timestamp t_u^m:

$$\mathcal{H}_u = \{(x_u^k, t_u^k) : t_u^m - \Delta_H \leq t_u^k < t_u^m\}, \quad (1)$$

$$\mathcal{S}_u = \{(x_u^k, t_u^k) \in \mathcal{I}_u : t_u^m \leq t_u^k \leq t_u^n\}. \quad (2)$$

Here, $\mathcal{H}_u$ captures long-term preference signals, while $\mathcal{S}_u$ contains goal-driven queries and candidates. This disentanglement separates preference inference from task execution , enabling reliable evaluation of personalization reasoning.

3.1.2 Non-trivial Session Filtering

For each user u with a session of ordered events $\mathcal{S}_u$ (Section 3.1.1), each event is one of *search*, *product browsing*, *media (video/livestream) browsing*, or *purchase*. It can be represented as

$$\mathcal{S}_u = \left(\mathcal{Q}_u^{\text{search}},\ \mathcal{P}_u^{\text{view}},\ \mathcal{V}_u^{\text{view}},\ \mathcal{P}_u^{\text{order}}\right),$$

where $\mathcal{Q}_u^{\text{search}}$ is the ordered list of search queries, $\mathcal{P}_u^{\text{view}}$ the product-interactions, $\mathcal{V}_u^{\text{view}}$ the media-interactions, and $\mathcal{P}_u^{\text{order}}$ the purchased products. To obtain non-trivial shopping sessions, we formulate these three conditions:

(1) Iterative query refinement. The user progressively clarifies intent through multiple searches:

$$|\mathcal{Q}_u^{\text{search}}| \geq 2. \quad (3)$$

(2) Extensive-browsing behavior. The user exhibits substantial exploration prior to purchase:

$$|\mathcal{P}_u^{\text{view}}| + |\mathcal{V}_u^{\text{view}}| \geq \tau_v, \quad (4)$$

We set $\tau_v = 10$, requiring at least ten product or media interactions before purchase.

(3) Intent–order alignment. The purchase must align with a query and have sufficient same-category alternatives. Specifically, there exists a purchased product $p_u^o \in \mathcal{P}_u^{\text{order}}$ such that

$$\exists\, q \in \mathcal{Q}_u^{\text{search}} \text{ s.t. } \mathbb{I}[\text{Align}(q, p_u^o)] = 1,$$
$$\left|\left\{v \in \mathcal{P}_u^{\text{view}} \setminus \{p_u^o\} : \text{cat}(v) = \text{cat}(p_u^o)\right\}\right| \geq \tau_a,$$

We set $\tau_a = 9$ for non-purchased products in the purchase category; all viewed products remain candidates, and one purchase is retained.

Rule-based selection enforces the category floor (Appendix A.3.1); GPT-4.1 verifies query–purchase alignment with human validation (Appendix A.3.2). We retain one qualifying session per user. These criteria select active, exploration-heavy, short-horizon shopping sessions with observable query refinement: each retained session contains at least two queries and at least ten product or media interactions before a purchase within 30 minutes of the initial search (Appendix A.5.2).

3.2 History & Query-Candidate Identification

For users with retained sessions, we construct their heterogeneous histories and aligned query–candidate pairs.

3.2.1 Heterogeneous History Curation

Heterogeneous long-term interactions provide rich signals for inferring user persona. For a user u with historical interaction log $\mathcal{H}_u$ (Eq. 1), we first filter out low–dwell-time records (Appendix A.4.1). We then consider two complementary types for the remaining interactions.

(1) Product interactions. Each product interaction is defined as

$$x_{u,\text{product}}^k = (\text{meta}_p^k,\ \text{type}^k,\ \Delta t^k), \tag{5}$$

where $\text{meta}_p^k = (\text{id},\ \text{cat},\ \text{title},\ \text{detail},\ \text{features})$ encodes product metadata, $\text{type}^k \in \{\text{view}, \text{like}, \text{cart}, \text{order}\}$, and Δt^k is dwell time.

(2) Media interactions. A media (video or livestream) interaction is represented as

$$x_{u,\text{media}}^k = (\text{meta}_m^k,\ \Delta t^k), \tag{6}$$

where $\text{meta}_m^k = (\text{id},\ \text{tags},\ \text{title},\ \text{detail},\ \mathcal{P}_{\text{refer}}^k)$, and $\mathcal{P}_{\text{refer}}^k$ denotes the set of products referenced or mentioned in the media, while Δt^k captures user engagement duration (see Appendix A.4.2).

3.2.2 Intent & Refined Query

We distinguish two query types from the query refinement trajectory based on their position and role. For each user u, let $\mathcal{Q}_u^{\text{search}} = (q_u^1, \ldots, q_u^{|\mathcal{Q}_u^{\text{search}}|})$ denote the temporally ordered search queries in the retained search-to-order session, and let p_u^o be the purchased product. We define:

(i) **Intent query** q_u^{intent}: the earliest query in $\mathcal{Q}_u^{\text{search}}$ semantically aligned with p_u^o, denoting an intentional but ambiguous goal (e.g. *campus ootd*).

(ii) **Refined query** q_u^{refined}: the last query issued before purchase that is explicit in product type/attributes, representing a well-specified intent (e.g. *oversized long-sleeve*).

Query–product alignment is judged using GPT-4.1 with human verification. We discard cases containing only a single aligned query (Appendix A.3.3). Blinded construction audits show strong agreement for both session alignment and intent/refined-query localization (Appendices A.3.2 and A.3.3). Unlike refined, which primarily evaluates semantic retrieval (Liu et al., 2023a), intent reflects early-stage search behavior requiring intent inference (Downey et al., 2008). The distinction is defined by the session trajectory, not a fixed semantic boundary—identical query forms can reflect different levels of ambiguity at different stages of refinement. Accordingly, intent queries stress an agent's capacity to resolve underspecified goals into actionable intent under personalization.

3.2.3 Hard Candidate Identification

All unbought viewed products are candidates; at least nine share the purchase category. For the retained session of user u, the hard candidate set is defined as

$$\mathcal{C}_u^{\text{hard}} = (\mathcal{P}_u^{\text{view}} \setminus \{p_u^o\}) \cup \{p_u^o\}, \tag{7}$$

where each candidate $c \in \mathcal{C}_u^{\text{hard}}$ is represented by its product metadata meta_c. Candidate metadata comprises both commercial attributes, such as price and promotion, and fine-grained product attributes, including material and ingredients. Candidate-side ablations show that commercial attributes provide modest auxiliary value, whereas full metadata substantially outperforms commercial attributes alone (Appendix A.5.1). This indicates that distinguishing p_u^o from $\mathcal{C}_u^{\text{hard}}$ requires agents to replicate the user's personalized, preference-aligned comparison process over fine-grained attributes beyond commercial cues.

3.3 Benchmark Evaluation

APeB evaluates models under two query settings:

Input. For query variant $v \in \{\text{intent}, \text{refined}\}$:

$$\text{Input}_u^v = (q_u^v, \mathcal{H}_u, \mathcal{C}_u^{\text{hard}}).$$

Output and Metrics. We return five items and average the instance-level scores:

$$(\hat{c}_{u,k})_{k=1}^5 = \text{Rank}_\theta(\text{Input}_u^v)_{1:5},$$
$$\text{Hit@K}(u) = \mathbb{I}[p_u^o \in \{\hat{c}_{u,k}\}_{k=1}^K], \quad K \in \{1, 5\}.$$

4 Benchmark Post-Processing

Anonymization Before evaluation, user behavior data are anonymized to prevent personal information leakage. Raw user identifiers are removed, and each interaction sequence is mapped to a randomly assigned surrogate ID with no cross-split linkability. Temporal signals are discretized into coarse-grained time bins to avoid fine-grained behavioral tracing. We retain public item and content metadata to support heterogeneous personalization tasks. Meanwhile, we apply automated personally identifiable information redaction to all free-form text—particularly OCR and ASR outputs—to remove explicit and implicit identifiers. APeB will be released through the project website[1] after compliance and privacy review, with data minimization; Appendix A.4.3 details the release scope and governance.

Statistics APeB contains 5,648 retained English cases spanning major categories such as Personal Care, Clothing, Food & Beverages, and Home Supplies. On average, each user has 187 historical interactions, including 82 product and 105 media interactions. These histories contain an average of 18 historical orders. After excluding $>$ 25-candidate sessions, candidate sets contain 14.1 ± 2.8 items, including at least nine non-purchased products in the purchase category (Appendix A.5.1).

5 Experiments and Analysis

We evaluate personalized shopping goal completion and intermediate reasoning on **APeB** under refined and intent across 8 LLMs and their agentic variants, addressing the following questions:

RQ1: How do LLM workflows perform compared to recommender PPS methods? We compare hit rates on intent and refined queries with different history/candidate sets.

RQ2: Which types of flaws most adversely affect personalization tasks? We conduct a rubric analysis to attribute performance drop to intermediate reasoning errors.

RQ3: How can the failures in RQ2 be addressed? We use a Vague Query Refinement Agent (VQRA) to demonstrate that explicit history utilization modules reduce this gap.

5.1 Experimental Setup

5.1.1 Evaluated Models

We evaluate: (1) PPS baseline: UniSAR (Shi et al., 2024). (2) LLMs: Qwen3-235B (Yang et al., 2025a), DeepSeek-R1 (Guo et al., 2025); GPT-4/5 series (OpenAI et al., 2024; Singh et al., 2025), Gemini-2.5-Pro (Google DeepMind, 2025). (3) LLM Agents: ReAct (Yao et al., 2023), Deerflow (ByteDance et al., 2025). Appendix B.3 discusses additional baselines.

5.1.2 Benchmark Utilization

Due to the large semantic volume, we utilize **APeB**:

(1) UniSAR - Supervised Training Following Sun et al. (2023), we construct SAR training data from historical query–order pairs and isolated purchases. The training history is defined as $\mathcal{H}_u^{\text{SAR}} = \{i_u^k\}_{k=1}^{|\mathcal{H}_u^{\text{SAR}}|}$, where each interaction $i_u^k \in \{(q_u^k, p_u^k), p_u^k\}$.

(2) LLM - Single Prompt To fit histories, queries, and candidates within the LLM context window, we retain k records from $\mathcal{H}_u$. We evaluate category-prior and time-prior history selection (same-category label associated with the intent query vs. most-recent records), keeping only essential fields per record (title, price). Details are in Appendix B.4.

(3) Agent - External Retrieval Database Agents use a retrieval tool backed by a database over history and candidate metadata. At each step, agents issue queries to retrieve extra information beyond the prompt (Appendix B.5, B.6).

5.1.3 Evaluation Method

Structured Output: To evaluate both task success and intermediate steps, Algorithm 1 prompts models to produce structured outputs, including user target, history summarization, and recommend items. For ReAct, reasoning steps Re_i and retrieval queries Rq_i at each step i are logged.

[1] https://voxxxx1874.github.io/APeB-Website/

Query	Method	Metric (%)	Gemini	4.1mini	GPT4o	Qwen3	R1	GPT4.1	GPT5.2	GPT5.2Think	UniSAR*
Intent	Single-Prompt	Hit@1	22.1	23.6	24.9	22.7	24.2	23.9	25.6	24.2	**25.9**
		Hit@5	76.6	71.1	75.7	71.5	74.7	73.3	**78.3**	77.8	59.2
	ReAct	Hit@1	24.2	26.0	26.3	24.6	24.8	25.8	26.2	**26.4**	–
		Hit@5	76.2	72.2	75.8	71.1	73.9	77.9	77.5	77.3	–
	Deerflow	Hit@1	21.1	22.6	23.2	23.1	22.9	22.2	23.5	23.5	–
		Hit@5	71.3	69.4	71.5	72.6	70.9	72.3	72.5	73.8	–
	VQRA	Hit@1	25.2	23.9	26.3	25.0	26.2	24.9	26.4	**26.9**	–
		Hit@5	76.9	72.2	75.7	75.0	76.3	74.8	**78.5**	78.2	–
Refined	Single-Prompt	Hit@1	33.9	34.4	37.0	35.8	36.2	34.5	**37.4**	37.0	29.2
		Hit@5	81.9	79.6	82.8	81.6	80.7	78.8	**84.2**	83.6	64.5
	ReAct	Hit@1	35.7	36.4	37.8	37.3	36.0	40.4	**43.3**	42.8	–
		Hit@5	81.8	79.7	84.2	82.7	81.7	83.2	**84.4**	83.4	–

Table 2: Hit@1/5 performance for LLM workflows (Single-Prompt (60 category-prior selected records), ReAct, Deerflow and VQRA) and PPS model UniSAR under intent and refined queries. VQRA denotes the setting where the intent query has been rewritten with a shared backbone (GPT-4.1) using selected history context before evaluation.

Algorithm 1 Structured Product Recommendation Step

Input: intent/refined query q, histories $\mathcal{H}$, candidates $\mathcal{C}^{\text{hard}}$
Target inference: Infer user target τ from $(q, \mathcal{H})$.
Relevant history selection: $\mathcal{H}^\star \leftarrow$ histories salient to τ.
History summarization: $\mathcal{F} = \{(f_i, \pi_i, \mathcal{H}_i^\star)\}_{i=1}^m$, where f_i is a preference fact, π_i its relevance rank, and $\mathcal{H}_i^\star \subseteq \mathcal{H}^\star$ its supporting records.
Recommend items: $\mathcal{R}^\star = ((c_j, \rho_j, \mathcal{F}_j))_{j=1}^5$, where j is the recommendation rank, c_j the candidate, ρ_j its rationale, and $\mathcal{F}_j \subseteq \mathcal{F}$ its supporting preferences.

Outcome & Reasoning: Personalization performance is measured using Hit@1/5 on the recommended items $\mathcal{R}^\star$. A model succeeds when it ranks the purchased item above user-considered alternatives. We further apply rubric evaluation to intermediate outputs using *LLM-as-a-Judge*, scoring each component independently. The intermediate rubrics are anchored to the same user session evidence: they diagnose whether failures arise from intent inference, history-grounded preference extraction, recommendation alignment, or multi-step reasoning. Detailed evaluation is in Section 5.3.

5.2 Overall Results and Analysis (RQ1)

5.2.1 LLMs Underperform on Intent Queries

Table 2 shows that while single-prompt LLMs achieve strong Hit@5 across workflows under both query types, they slightly underperform UniSAR on Hit@1 for intent queries. This highlights their **weakness in precise personalized ranking under vanilla goals and noisy histories**, where user modeling outweighs semantic matching. A scaling experiment on Qwen series further shows that model scaling improves Hit@1 but does not eliminate the persistent intent–refined gap (Appendix B.1.1).

We further evaluate agents with multi-step reasoning and tool use. In Table 2, the refined-query block shows that ReAct agents generally outperform single-prompt LLMs, while the intent-query block shows that ReAct yields only marginal gains and Deerflow shows little benefit or even degradation. These results indicate that **increased semantic capacity alone does not guarantee improved personalization**: when user goals are weakly specified, additional planning and retrieval steps can instead amplify reasoning errors. Detailed analysis is in Section 5.3. Additional paired bootstrap analysis with confidence intervals is provided in Appendix B.2.

5.2.2 Hard Candidates and Noisy History

LLMs struggle to benefit from history. Table 3 shows that with no-history, models demonstrate basic semantic matching ability, while intent remains more challenging. When user history is introduced, trained UniSAR achieves substantial gains. Short and long category-prior histories (@6/@60 CATE) yield very similar Hit@1 under intent queries; while the refined-query block shows larger improvements once intent ambiguity is reduced. Comparisons between intent and refined queries imply that history possesses usable preference signal but current LLM workflows do not naturally benefit possibly because LLMs do not ex-

History	UniSAR	Qwen3	GPT5.2	R1
Intent query				
@0	14.1	21.2	23.8	22.2
@6 Cate	–	21.6	24.7	23.7
@60 Cate	↑25.9 (+84%)	↑22.7 (+7.1%)	↑25.6 (+7.6%)	↑24.2 (+9.0%)
Refined query				
@0	18.3	30.4	33.4	31.9
@6 Cate	–	33.1	35.2	35.0
@60 Cate	↑29.2 (+59.6%)	↑35.8 (+17.8%)	↑37.4 (+12.0%)	↑37.0 (+16.0%)

Table 3: History ablation: Hit@1 (%) on intent/refined queries. @0 = no history; @k Cate = k category-prior records; arrows = change vs. @0.

Candidates	UniSAR	Qwen3	GPT4.1	GPT5.2	R1
Random	81.3	84.8	86.3	**95.9**	93.1
Hard	**25.9**	22.7	23.9	25.6	24.2

Table 4: Candidate ablation on intent queries: Hit@1 (%) with random vs. hard candidates.

tract recoverable preferences under intent queries. History interventions show that user-matched histories outperform shuffled and counterfactual histories, while simply expanding the context yields limited benefit (Appendices B.4.3 and B.4.4). Replacing raw histories with explicitly extracted preferences yields only modest gains for GPT-4.1-mini and Qwen3 and leaves GPT-5.2 unchanged, showing that a separate preference-extraction step alone provides limited end-to-end improvement. The remaining difficulty may stem from incomplete recovery of task-relevant preferences, ineffective use of those preferences in fine-grained candidate comparison (Table 22). Further analysis in Section 5.4 provides stepwise evidence.

Personalization choice lies beyond semantic matching. Table 4 shows that, even under vague queries, LLMs outperform the UniSAR baseline when evaluated with random candidate sets, which are sampled from all possible candidates in the dataset. This gain mainly stems from LLMs' ability to exploit coarse semantic cues in underspecified intents to filter out obviously mismatched negatives and locate the query-aligned purchased item. State-of-the-art models (e.g., GPT-5.2) further widen this margin over older or open-source models (Qwen3, GPT-4.1). Under hard candidates, both gains over UniSAR and inter-LLM performance gaps largely disappear, indicating that current LLMs struggle to discriminate fine-grained attributes among user-viewed candidates. This also shows that: random negatives often require coarse semantic matching rather than true personalization. Hard candidates drawn from the same shopping trajectory better test whether a model can distinguish the purchased item based on user-specific preferences.

5.3 Rubrics on Intermediate Reasoning (RQ2)

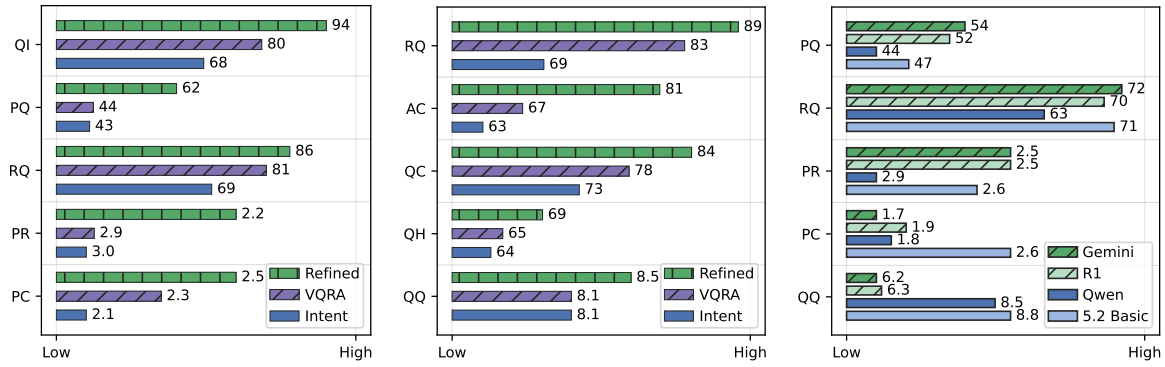
We conduct rubric-based evaluation of intermediate outputs in Section 5.1.3 to diagnose why LLMs and ReAct agents underperform on intent queries and to compare behaviors across LLMs. We first infer a diagnostic goal proxy g from in-session interactions (Eq.2) (Appendix B.8.1), and evaluate each output against g using *LLM-as-a-Judge*. The diagnostic scores are averaged across Gemini-2.5-Pro, GPT-5.2, and GPT-4.1 judges with human verification. Appendix B.8.3 maps the four stages below to their evaluated outputs and diagnostic criteria.

Multi-Level Rubric We decompose the evaluation into stepwise complementary aspects (Appendix B.8.2): (1) *Query Inference (QI)*—whether the model understands the query and identifies its intended candidate category; (2) *Historical Preference Reasoning*—via i. *Preference Quality (PQ)*, assessing whether extracted preferences $\mathcal{F}$ faithfully summarize the query-specific aspects the user cares about from history, and ii. Salience Statistics: minimum rank (*PR*) and total count (*PC*) of preferences supporting the purchased item; (3) *Recommendation Quality (RQ)*—whether the recommended items $\mathcal{R}^\star$ and their rationales ρ address the query and align with the user's preferences; (4) *Agent Reasoning (AR)*—coherence of reasoning steps Re_i (AC) and retrieval utility, including candidate/history queries (QC, QH) and query quantity (QQ), toward achieving g without hallucination. Rubric–Hit@K alignment and judge-sensitivity checks in Appendix B.8.3 and B.8.4 support using these scores as diagnostic evidence.

Intermediate Results Figure 3 shows rubric scores for intent and refined queries under single-prompt and ReAct workflows. We observe:

(1) Failures originate from intent understanding. As shown in Figure 3a, *query inference* performance drops substantially on intent compared to refined queries, undermining downstream preference grounding and leading to degraded *recommendation quality*. Notably, *preference reasoning* remains limited even under explicit refined queries, indicating that identifying and summarizing relevant signals from heterogeneous histories remain an open challenge, even with clarified goals. Under vague intents, inferring meaningful historical preferences is particularly difficult.

(2) Agentic workflows amplify strengths and



(a) Averaged scores over LLM models on Intent/VQRA/Refined query

(b) Averaged scores over ReAct models on Intent/VQRA/Refined query

(c) Per-model analysis of intermediate reasoning components on Intent query

Figure 3: Rubric scores for evaluating models across intermediate stepwise outputs (QI: Query Inference; PQ: Preference Quality; RQ: Recommendation Quality; PR: Minimum Observed-Purchase Preference Rank ↓; PC: Total Observed-Purchase Preference Count; AC: Agent Coherence; QC: Query–Candidate; QH: Query–History; QQ: Query Quantity). Diagnostic scores are averaged over Gemini-2.5-Pro, GPT-5.2, and GPT-4.1 judges.

weaknesses. Comparing Figures 3a and 3b, multi-step retrieval improves recommendation quality under explicit refined queries but offers little benefit for intent queries. Under vague intents, low reasoning coherence yields ineffective retrieval queries, causing agentic reasoning to compound intent errors, consistent with the intent-query block of Table 2. Moreover, retrieving relevant signals from user histories is substantially harder than retrieving candidate information under both query types. Appendix B.5.1 further analyzes ReAct's retrieval utility, round depth and reasoning coherence.

(3) Models exhibit different intermediate behaviors. Figure 3c shows that models with comparable Hit@1 diverge in their intermediate reasoning steps. Advanced reasoning models (e.g., Gemini-2.5-Pro, DeepSeek-R1) achieve better intermediate performance in history preference reasoning and recommendation quality. They rank purchase-supporting preferences earlier (lower PR) but find fewer (lower PC). They also issue fewer retrieval queries under ReAct (lower QQ). These might become detrimental especially under intent queries where insufficient preference coverage undermines recommendation.

5.4 VQRA: Refine Intent Query First (RQ3)

Section 5.3 and Figure 3 attribute the performance drop on intent queries to weak query inference due to underexplored histories. To isolate this gap, we evaluate a history-conditioned query-refinement agent, VQRA, in which a shared backbone (GPT-4.1) first rewrites the intent query using selected same-category histories and the LLM backbone is then evaluated with the rewritten query under the single-prompt pipeline. Table 2 shows that VQRA improves Hit@1, with the largest gain on GPT-5.2Think. Appendix B.10 shows similar results with the same-backbone-rewrite variant. Rubric scores in Figure 3a, 3b suggest that query refinement improves *query inference (QI)* toward the shopping intent, leading to gains in *recommendation quality (RQ)*. These indicate that inferring intent from vague queries and noisy histories remains underutilized: with explicit history grounding, LLMs already produce useful refinements, suggesting headroom for personalized agent design.

6 Conclusion

We introduce APeB, a benchmark for agentic personalization that evaluates personalized product search under underspecified queries and noisy interaction histories. Through APeB, we show that current LLMs handle explicit queries well but struggle in early-stage personalization, where they must infer intent, recover preferences, and choose among competing alternatives. Rubric analysis links this gap to ineffective history use: models struggle to extract task-relevant, well-grounded preferences from user histories and apply them to candidate selection. Furthermore, VQRA, a simple history-aware query-refinement pipeline, improves performance across backbones, highlighting the value of explicit history-utilization modules. APeB serves as a testbed for robust user modeling in real-world personalization.

Limitations

APeB is drawn from a single content-commerce platform; behavior patterns, query styles, and candidate structures may not transfer to other e-commerce ecosystems or non-shopping personalization domains. All evaluation is static and offline: fixed candidate sets and recorded purchases do not reflect real-time inventory, pricing, promotions, or multi-turn interaction. Observed purchases serve as ground-truth preference proxies but are confounded by ranking exposure, stock availability, and platform mediation. We explore category- and time-based history selection without exhaustively studying advanced memory architectures or adaptive context-length strategies. Rubric analysis relies on LLM-as-a-Judge, which may introduce systematic biases despite strong human agreement. The benchmark covers limited locale and language; intent expression and attribute salience may vary across cultures. The retained benchmark focuses on active, exploration-heavy shopping sessions with an attributable query–purchase outcome within a 30-minute window; future work may extend the attribution protocol to sparse and cross-day journeys. The evaluation input represents heterogeneous media histories through postprocessed summaries, captions, anonymized OCR/ASR text, and referenced products; future work may add raw-media inputs.

Ethical Considerations

This work studies personalization benchmarks derived from postprocessed behavioral shopping records. Its positive impact is to make evaluation of preference-conditioned product search more realistic and auditable, especially when users issue ambiguous queries. The main risks are privacy leakage from long histories or product/media text, re-identification through rare behavior patterns, category or platform bias, and misuse of preference inference for manipulative over-personalization. We mitigate these risks through a layered release pipeline that exposes only postprocessed, anonymized benchmark instances rather than raw logs. We apply coordinated semantics-preserving substitution across queries, histories, and candidate descriptions: source-specific expressions are replaced with semantically matched text, while query–history–candidate alignment and task-critical product attributes—including product type, size, material, and ingredients—are retained. We further remove direct identifiers and platform-private fields, assign surrogate identifiers without cross-split linkability, coarsen temporal and price signals, minimize retained interactions, and redact personally identifiable content from free-form OCR/ASR text.

Acknowledgments

We thank the reviewers for their constructive comments and suggestions. Garry Yang, Zizhe Chen, Deyu Zou, Yongqiang Chen, and James Cheng were supported by a CRF (No. C2005-24Y) from the RGC of Hong Kong. This work was a collaboration between Husky Data Lab at CUHK and ByteDance, supported by a ByteDance University Collaboration Project Grant. Generative AI assistants were used for language polishing, writing and coding assistance, and preparation of submission materials. All AI-assisted content and outputs were reviewed and verified by the authors, who take full responsibility for the work.

References

Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W. Bruce Croft. 2017. Learning a hierarchical embedding model for personalized product search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 645–654, New York, NY, USA. Association for Computing Machinery.

David Ben-Shimon, Alexander Tsikinovsky, Michael Friedmann, Bracha Shapira, Lior Rokach, and Johannes Hoerle. 2015. Recsys challenge 2015 and the YOOCHOOSE dataset. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015*, pages 357–358. ACM.

ByteDance, Daniel Walnut, and Henry Li. 2025. DeerFlow: Deep exploration and efficient research flow. https://github.com/bytedance/deer-flow/. Repository snapshot, commit ba7509d9ae10158a6e241fc15b154a29788fe467.

Ben Chen, Xian Guo, Siyuan Wang, Zihan Liang, Yue Lv, Yufei Ma, Xinlong Xiao, Bowen Xue, Xuxin Zhang, Ying Yang, Huangyu Dai, Xing Xu, Tong Zhao, Mingcan Peng, Xiaoyang Zheng, Chao Wang, Qihang Zhao, Zhixin Zhai, Yang Zhao, and 9 others. 2025. Onesearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search. *Preprint*, arXiv:2509.03236.

Zizhe Chen, Jiqian Dong, Yizhou Tian, Garry Yang, Yongqiang Chen, Zhitang Chen, and James Cheng. 2026. Hista and numca: Estimate state value effectively for llm reinforcement learning. *Preprint*, arXiv:2605.29782.

Dian Cheng, Jiawei Chen, Wenjun Peng, Wenqin Ye, Fuyu Lv, Tao Zhuang, Xiaoyi Zeng, and Xiangnan He. 2022. Ihgnn: Interactive hypergraph neural network for personalized product search. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 256–265, New York, NY, USA. Association for Computing Machinery.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready AI agents with scalable long-term memory. *CoRR*, abs/2504.19413.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Doug Downey, Susan T. Dumais, Daniel J. Liebling, and Eric Horvitz. 2008. Understanding the relationship between searchers' queries and information goals. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*, pages 449–458. ACM.

Yiming Du, Wenyu Huang, Danna Zheng, Zhaowei Wang, Sebastien Montella, Mirella Lapata, Kam-Fai Wong, and Jeff Z. Pan. 2025. Rethinking memory in llm based agents: Representations, operations, and emerging topics. *Preprint*, arXiv:2505.00675.

Huizhong Duan and ChengXiang Zhai. 2015. Mining coordinated intent representation for entity search and recommendation. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 333–342, New York, NY, USA. Association for Computing Machinery.

Lu Fan, Qimai Li, Bo Liu, Xiao-Ming Wu, Xiaotong Zhang, Fuyu Lv, Guli Lin, Sen Li, Taiwei Jin, and Keping Yang. 2022. Modeling user behavior with graph convolution for personalized product search. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 203–212, New York, NY, USA. Association for Computing Machinery.

Google DeepMind. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *Preprint*, arXiv:2507.06261.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian Mcauley. 2022. Query-aware sequential recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 4019–4023, New York, NY, USA. Association for Computing Machinery.

Yupeng Hou, Jiacheng Li, Xiangjun Fu, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2026. Bridging language and items for retrieval and recommendation: Benchmarking LLMs as semantic encoders. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3251–3265, San Diego, California, United States. Association for Computational Linguistics.

Yuanzhe Hu, Yu Wang, and Julian McAuley. 2026. Evaluating memory in LLM agents via incremental multi-turn interactions. In *The Fourteenth International Conference on Learning Representations*.

Zhaopei Huang, Qifeng Dai, Guozheng Wu, Xiaopeng Wu, Kehan Chen, Chuan Yu, Xubin Li, Tiezheng Ge, Wenxuan Wang, and Qin Jin. 2025. Mem-pal: Towards memory-based personalized dialogue assistants for long-term user-agent interaction. *Preprint*, arXiv:2511.13410.

Wei Jin, Haitao Mao, Zheng Li, Haoming Jiang, Chen Luo, Hongzhi Wen, Haoyu Han, Hanqing Lu, Zhengyang Wang, Ruirui Li, Zhen Li, Monica Xiao Cheng, Rahul Goutam, Haiyang Zhang, Karthik Subbian, Suhang Wang, Yizhou Sun, Jiliang Tang, Bing Yin, and Xianfeng Tang. 2023. Amazon-m2: A multilingual multi-locale shopping session dataset for recommendation and text generation. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yilun Jin, Zheng Li, Chenwei Zhang, Tianyu Cao, Yifan Gao, Pratik Jayarao, Mao Li, Xin Liu, Ritesh Sarkhel, Xianfeng Tang, Haodong Wang, Zhengyang Wang, Wenju Xu, Jingfeng Yang, Qingyu Yin, Xian Li, Priyanka Nigam, Yi Xu, Kai Chen, and 3 others. 2024. Shopping MMLU: A massive multi-task online shopping benchmark for large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Hao Zhang, Yong Liu, Chuhan Wu, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. 2024. How can recommender systems benefit from large language models: A survey. *Preprint*, arXiv:2306.05817.

Guimei Liu, Tam T. Nguyen, Gang Zhao, Wei Zha, Jianbo Yang, Jianneng Cao, Min Wu, Peilin Zhao, and Wei Chen. 2016. Repeat buyer prediction for

e-commerce. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 155–164. ACM.

Jiongnan Liu, Zhicheng Dou, Guoyu Tang, and Sulong Xu. 2023a. Jdsearch: A personalized product search dataset with real queries and full interactions. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2945–2952, New York, NY, USA. Association for Computing Machinery.

Peng Liu, Lemei Zhang, and Jon Atle Gulla. 2023b. Pre-train, prompt and recommendation: A comprehensive survey of language modelling paradigm adaptations in recommender systems. *Preprint*, arXiv:2302.03735.

Zhanyu Liu, Shiyao Wang, Xingmei Wang, Rongzhou Zhang, Jiaxin Deng, Honghui Bao, Jinghao Zhang, Wuchao Li, Pengfei Zheng, Xiangyu Wu, Yifei Hu, Qigen Hu, Xinchen Luo, Lejian Ren, Zixing Zhang, Qianqian Wang, Kuo Cai, Yunfan Wu, Hongtao Cheng, and 7 others. 2025. Onerec-think: In-text reasoning for generative recommendation. *Preprint*, arXiv:2510.11639.

Yougang Lyu, Xiaoyu Zhang, Lingyong Yan, Maarten de Rijke, Zhaochun Ren, and Xiuying Chen. 2025. Deepshop: A benchmark for deep research shopping agents. *Preprint*, arXiv:2506.02839.

Lucie Charlotte Magister, Katherine Metcalf, Yizhe Zhang, and Maartje ter Hoeve. 2024. On the way to llm personalization: Learning to remember user conversations. *Preprint*, arXiv:2411.13405.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2024. Memgpt: Towards llms as operating systems. *Preprint*, arXiv:2310.08560.

Bo Peng, Xinyi Ling, Ziru Chen, Huan Sun, and Xia Ning. 2024. ecellm: Generalizing large language models for e-commerce from large-scale, high-quality instruction data. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Qiyao Peng, Hongtao Liu, Hua Huang, Qing Yang, and Minglai Shao. 2025. A survey on llm-powered agents for recommender systems. *Preprint*, arXiv:2502.10050.

Zhiyuan Peng, Xuyang Wu, Huaixiao Tou, Yi Fang, and Yu Gong. 2026. Memrerank: Preference memory for personalized product reranking. *Preprint*, arXiv:2603.29247.

Girish Punj and Robert Moore. 2009. Information search and consideration set formation in a web-based store environment. *Journal of Business Research*, 62(6):644–650. Retailing Evolution Research.

Chandan K. Reddy, Lluís Màrquez, Fran Valero, Nikhil Rao, Hugo Zaragoza, Sambaran Bandyopadhyay, Arnab Biswas, Anlu Xing, and Karthik Subbian. 2022. Shopping queries dataset: A large-scale esci benchmark for improving product search. *Preprint*, arXiv:2206.06588.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. Bpr: Bayesian personalized ranking from implicit feedback. *Preprint*, arXiv:1205.2618.

Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy, Omar Zia Khan, and Abhinav Sethy. 2023. Integrating summarization and retrieval for enhanced personalization via large language models. *Preprint*, arXiv:2310.20081.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Stuart Russell. 2021. Human-compatible artificial intelligence. In *Human-Like Machine Intelligence*. Oxford University Press.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. Lamp: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 7370–7392. Association for Computational Linguistics.

Yu Shang, Peijie Liu, Yuwei Yan, Zijing Wu, Leheng Sheng, Yuanqing Yu, Chumeng Jiang, An Zhang, Fengli Xu, Yu Wang, Min Zhang, and Yong Li. 2025. Agentrecbench: Benchmarking llm agent-based personalized recommender systems. *Preprint*, arXiv:2505.19623.

Chenhui Shen, Liying Cheng, Xuan-Phi Nguyen, Yang You, and Lidong Bing. 2023. Large language models are not yet human-level evaluators for abstractive summarization. *Preprint*, arXiv:2305.13091.

Teng Shi, Zihua Si, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Dewei Leng, Yanan Niu, and Yang Song. 2024. UniSAR: Modeling user transition behaviors between search and recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1029–1039. ACM.

Zihua Si, Zhongxiang Sun, Xiao Zhang, Jun Xu, Xiaoxue Zang, Yang Song, Kun Gai, and Ji-Rong Wen. 2023. When search meets recommendation: Learning disentangled search representation for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 1313–1323, New York, NY, USA. Association for Computing Machinery.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, Akshay Nathan, Alan Luo, Alec Helyar, Aleksander Madry, Aleksandr Efremov, Aleksandra Spyra, Alex Baker-Whitcomb, Alex Beutel, Alex Karpenko, and 465 others. 2025. Openai gpt-5 system card. *Preprint*, arXiv:2601.03267.

Chenkai Sun, Ke Yang, Revanth Gangi Reddy, Yi R. Fung, Hou Pong Chan, Kevin Small, ChengXiang Zhai, and Heng Ji. 2025. Persona-db: Efficient large language model personalization for response prediction with collaborative data refinement. *Preprint*, arXiv:2402.11060.

Zhongxiang Sun, Zihua Si, Xiaoxue Zang, Dewei Leng, Yanan Niu, Yang Song, Xiao Zhang, and Jun Xu. 2023. Kuaisar: A unified search and recommendation dataset. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, page 5407–5411. ACM.

Huaixiao Tou, Ying Zeng, Cong Ma, Muzhi Li, Minghao Li, Weijie Yuan, He Zhang, and Kai Jia. 2025. Shoppingcomp: Are llms really ready for your shopping cart? *Preprint*, arXiv:2511.22978.

Jiangyuan Wang, Kejun Xiao, Qi Sun, Huaipeng Zhao, Tao Luo, Jiandong Zhang, and Xiaoyi Zeng. 2025a. Shoppingbench: A real-world intent-grounded shopping benchmark for llm-based agents. *CoRR*, abs/2508.04266.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6).

Piaohong Wang, Motong Tian, Jiaxian Li, Yuan Liang, Yuqing Wang, Qianben Chen, Tiannan Wang, Zhicong Lu, Jiawei Ma, Yuchen Eleanor Jiang, and Wangchunshu Zhou. 2025b. O-mem: Omni memory system for personalized, long horizon, self-evolving agents. *Preprint*, arXiv:2511.13593.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. 2023. Augmenting language models with long-term memory. *Preprint*, arXiv:2306.07174.

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, Yu Su, Upol Ehsan, Malihe Alikhani, Toby Jia-Jun Li, Lydia B. Chilton, and Dakuo Wang. 2025c. Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation. *CoRR*, abs/2506.05606.

Shuyi Xie, Ziqin Liew, Hailing Zhang, Haibo Zhang, Ling Hu, Zhiqiang Zhou, Shuman Liu, and Anxiang Zeng. 2025. Towards reliable evaluation of large language models for multilingual and multimodal e-commerce applications. *Preprint*, arXiv:2510.20632.

Yuwei Yan, Yu Shang, Qingbin Zeng, Yu Li, Keyu Zhao, Zhiheng Zheng, Xuefei Ning, Tianji Wu, Shengen Yan, Yu Wang, Fengli Xu, and Yong Li. 2025. Agentsociety challenge: Designing llm agents for user modeling and recommendation on web platforms. *Preprint*, arXiv:2502.18754.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Yuqi Yang, Weiqi Wang, Baixuan Xu, Wei Fan, Qing Zong, Chunkit Chan, Zheye Deng, Xin Liu, Yifan Gao, Changlong Yu, Chen Luo, Yang Li, Zheng Li, Qingyu Yin, Bing Yin, and Yangqiu Song. 2025b. Sessionintentbench: A multi-task inter-session intention-shift modeling benchmark for e-commerce customer behavior understanding. *Preprint*, arXiv:2507.20185.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and Hao Zhou. 2025. Memagent: Reshaping long-context LLM with multi-conv rl-based memory agent. *CoRR*, abs/2507.02259.

Xiang Yue, Boshi Wang, Ziru Chen, Kai Zhang, Yu Su, and Huan Sun. 2023. Automatic evaluation of attribution by large language models. *Preprint*, arXiv:2305.06311.

Beibei Zhang, Yanan Lu, Ruobing Xie, Zongyi Li, Siyuan Xing, Tongwei Ren, and Fen Lin. 2025a. Harnessing multimodal large language models for personalized product search with query-aware refinement. *Preprint*, arXiv:2509.18682.

Junjie Zhang, Ruobing Xie, Yupeng Hou, Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2025b. Recommendation as instruction following: A large language model empowered recommendation approach. *ACM Trans. Inf. Syst.*, 43(5).

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025c. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

Yu Zhang, Shutong Qiao, Jiaqi Zhang, Tzu-Heng Lin, Chen Gao, and Yong Li. 2025d. A survey of large language model empowered agents for recommendation and search: Towards next-generation information retrieval. *Preprint*, arXiv:2503.05659.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1079–1088. ACM.

Deyu Zou, Yongqiang Chen, Fan Feng, Mufei Li, Pan Li, Yu Gong, and James Cheng. 2026a. On information self-locking in reinforcement learning for active reasoning of LLM agents. *CoRR*, abs/2603.12109.

Deyu Zou, Yongqiang Chen, Jianxiang Wang, Garry YANG, Mufei Li, Qing Da, James Cheng, Pan Li, and Yu Gong. 2026b. Reducing belief deviation in reinforcement learning for active reasoning of llm agents. In *International Conference on Learning Representations*, volume 2026, pages 21511–21545.

Simiao Zuo, Qingyu Yin, Haoming Jiang, Shaohui Xi, Bing Yin, Chao Zhang, and Tuo Zhao. 2022. Context-aware query rewriting for improving users' search experience on e-commerce websites. *Preprint*, arXiv:2209.07584.

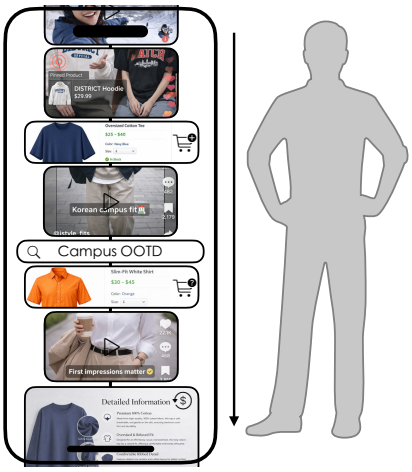


Figure 4: User interaction timeline on the base platform.

A Benchmark Collection

A.1 Holistic User Action Platform

The base platform underlying **APeB** hosts billions of short-form videos and livestreams and serves as a major venue for commerce. Users primarily consume video and livestream content via continuous scrolling, but can seamlessly transition to search for videos, livestreams, or products, as well as navigate directly to product detail pages embedded within promotional content. The queries collected in **APeB** are restricted to those exhibiting at least weak e-commerce intent. We further illustrate this user interaction timeline in Figure 4. All user data in **APeB** are collected in compliance with the platform's privacy policies and contractual data-use requirements.

A.2 Daily Active-User Filtering and Session Sampling

We process each day in the 30-day session-sampling window separately. For each sampling day, we retain users who have at least five product or media interactions on every day of the preceding 30 days. We then identify qualifying search-to-order sessions occurring on that sampling day within this daily active-user pool (Section 3.1). Across the 30 sampling days, if a user has multiple qualifying sessions, we retain the earliest one. For each retained session, we collect up to 60 days of historical interactions preceding the timestamp of its first search query, including e-commerce-related search queries, livestreams, videos, and product interactions.

A.3 Session Filtering

We detail the implementation of session filtering rules introduced in Section 3.1.2.

A.3.1 Rule-based Session Filtering

We first apply deterministic, rule-based filtering using SQL over raw interaction logs. SQL retains sessions satisfying: (i) at least ten product or media interactions (Eq. 4); (ii) at least two distinct queries (Eq. 3); and (iii) at least nine non-purchased viewed products sharing the purchased item's category. This category count is a session-level floor; after a session passes, its other viewed products remain in the final candidate set. These criteria remove shallow or accidental purchases and ensure that retained sessions reflect non-trivial exploration and intent evolution.

A.3.2 Intent–purchase alignment verification

For sessions passing the SQL filters, GPT-4.1 verifies that the purchased item is semantically consistent with at least one issued query. Human review validates this query–purchase decision. We use a shared audit sample of 100 de-identified sessions stratified by product category and query-sequence length. Two blinded annotators inspect the query trajectory, viewed-item metadata, and purchased-item metadata; a third annotator resolves disagreements. For query–purchase alignment, GPT-4.1 agrees with the adjudicated human decision on 97% of sessions, raw human–human agreement is 96%, and GPT-5.2 agreement is 97% under the same inputs. Sessions failing this semantic consistency check are discarded; if multiple purchased items satisfy the criterion, we retain only one to ensure a unique target per session.

A.3.3 Distinct intent and refined

Given a session u with query sequence $\mathcal{Q}_u^{\text{search}} = \{q_u^1, \ldots, q_u^{T_u}\}$ and purchased product p_u^o, we locate two aligned queries using GPT-4.1 by prompting it to *return indices* in $\mathcal{Q}_u^{\text{search}}$:

$$t_{\text{intent}} = \min\{t : q_u^t \text{ is semantically aligned with } p_u^o\},$$

the earliest aligned query, i.e., intent, and

$$t_{\text{refined}} = \max\{t : q_u^t \text{ is aligned and explicit in product type/attributes}\},$$

the last explicit aligned query before purchase, i.e., refined. We then set $q_u^{\text{intent}} = q_u^{t_{\text{intent}}}$ and $q_u^{\text{refined}} = q_u^{t_{\text{refined}}}$.

To ensure *distinct* intents, we apply a minimal rule-based check after GPT-4.1 localization: we discard session u if q_u^{intent} and q_u^{refined} are identical after normalization (lowercasing and whitespace/punctuation stripping), i.e., $\text{norm}(q_u^{\text{intent}}) = \text{norm}(q_u^{\text{refined}})$. This removes cases where the session contains only a single aligned query (or duplicated re-issues) and thus lacks an observable intent refinement trajectory.

For the same 100 stratified sessions, annotators identify the earliest aligned ambiguous query and the latest aligned explicit query from the de-identified query sequence, relevant product metadata, and purchased item. GPT-4.1 agrees with the adjudicated exact index pair on 93% of sessions, raw human–human agreement is 91%, and GPT-5.2 exact-pair agreement is 93%.

A.4 Heterogeneous History Curation

We curate heterogeneous user histories by unifying product-centric and media-centric interaction records into a structured sequence, enabling fine-grained modeling of user intent and preference evolution.

A.4.1 Dwell-time filtering

We discard video and product view interactions with dwell time shorter than 10 seconds, retaining only interactions that reflect meaningful user engagement.

A.4.2 Heterogeneous history content

Product interactions. Each product interaction $x_{u,\text{product}}^k$ (Eq. 5) corresponds to an explicit user action on an item. The product metadata meta_p^k consist of an *encrypted* product identifier id, a categorical taxonomy label category, and textual fields title and detail, where detail is parsed from the product's structured HTML page. The feature field contains a set of salient product attributes extracted by platform-level product understanding pipelines. The interaction type $\text{type}^k \in \{\text{view}, \text{like}, \text{cart}, \text{order}\}$ reflects increasing engagement intensity, and Δt^k denotes the dwell time associated with the interaction event.

Media interactions. Each media interaction $x_{u,\text{media}}^k$ (Eq. 6) represents user engagement with a short video or livestream. The media metadata meta_m^k include an *encrypted* media identifier id, content tags, and a title. The detail field is constructed from platform-generated multimodal summaries, including captions and anonymized OCR/ASR transcripts, to capture the semantic content of the media. The referenced product set $\mathcal{P}_{\text{refer}}^k$ consists of products explicitly linked or automatically annotated as mentioned within the media content. The engagement duration Δt^k measures the user's viewing or participation time. The contents of interacted media are represented by textual surrogates, such as summaries, captions, anonymized OCR/ASR text, and referenced products. The media summaries retain semantic and stylistic cues that can support personalized candidate selection. A lexical audit finds that 98.2% of audited media descriptions contain at least one camera/editing, temporal, presenter-style, or atmosphere marker. Moreover, media-summary histories achieve Hit@1 of 24.1 versus 24.2 with full histories for R1 and 22.3 versus 22.7 for Qwen3 (Table 21), indicating that the summaries preserve sufficient user-specific evidence for personalization in this setting.

A.4.3 Release Scope and Governance

APeB will be released through the project website as an anonymized benchmark package after privacy and compliance review. The package is designed for research evaluation and excludes fields that create avoidable re-identification risk. Table 5 summarizes the planned release boundary.

Category	Content and handling
Task records	Postprocessed queries, candidate sets, task labels, and purchased-item labels.
Preference context	Product and media metadata, summaries, referenced products, and key attributes.
Linkable fields	Entity expressions are substituted consistently across linked fields; identifiers are replaced with surrogates and timestamps are coarsened.
Raw/private data	Raw logs, media, and unredacted text; direct identifiers; device/location/profile data; and platform internals.

Table 5: Planned website release boundary for **APeB**. The package will include postprocessed task records and preference context, transform linkable fields, and exclude raw or private data.

Semantic-preserving entity substitution. Brand names, product names, and related source-specific expressions are replaced with semantically matched substitutes. A shared within-instance mapping is applied consistently across queries, histories, and candidate descriptions, retaining their relations and task-critical attributes such as product

type, size, material, and ingredients.

Privacy and linkage checks. Before the website release, each benchmark instance will be screened for raw user/account identifiers, cross-split persistent user identifiers, fine-grained timestamps, raw event-log payloads, and device, IP, GPS, or account-profile attributes.

Record and text checks. Exact duplicate product exposures are collapsed where needed for candidate construction. Free-form product and media text is scanned for phone numbers, emails, addresses, usernames, URLs, and other unique identifiers; sampled records containing OCR/ASR-derived text receive additional spot checks.

Data minimization. Data minimization is applied at both the user and field levels. We retain only the history window and shopping-session evidence needed to define the benchmark task, cap the number of retained interactions, and omit raw events outside the benchmark construction window. In the retained benchmark, users average 187 historical interactions; under the source-log accounting used for governance, the retained records correspond to about 200 interactions per user per month, approximately 10% of shopping-related activity.

Intended use and restrictions. The benchmark is intended for research evaluation of personalized product search, preference reasoning, history-aware ranking, and agentic shopping workflows. It is not intended for identifying individuals, reconstructing behavioral timelines, building user profiles outside the benchmark task, joining records with external datasets, production ad targeting, or optimizing manipulative personalization strategies. Dataset users should agree not to attempt re-identification, linkage attacks, demographic or sensitive-attribute inference about individuals, or redistribution outside the approved access terms.

A.5 Benchmark Validation and Scope

We characterize the retained benchmark records, validate hard-candidate construction and purchase recoverability, analyze activity and temporal properties, and illustrate the shared task structure across personalization settings. APeB covers 5,648 cases with the category distribution shown in Table 6. As Section 4 shows, retained users average 187 historical interactions, including 82 product interactions and 105 media interactions, with 18 historical orders on average; candidate sets contain 14.1 items on average.

Category	Percentage (%)
Personal Care	28.0
Clothing	20.2
Food & Beverages	15.2
Home Supplies	11.8
Consumer Electronics	8.9
Sports & Outdoor	5.1
Baby & Maternity Products	3.7
Others	7.1

Table 6: Category distribution of purchased products in the benchmark; lower-frequency categories are grouped under Others.

A.5.1 Hard-Candidate Controls and Purchase Recoverability

We first verify that hard-candidate sets are valid, size-controlled, de-duplicated, and drawn from sessions satisfying the same-category floor (Table 7). We then audit whether user histories provide evidence for the observed purchase and whether candidate decisions use metadata beyond commercial cues, including fine-grained attributes such as size, material, and ingredients. Human evaluation calibrates recoverability among these viewed alternatives.

Control	Retained setting
Set size	14.1 ± 2.8 items; discard sessions with > 25 candidates.
Category floor	≥ 9 non-purchased viewed products in the purchase category.
Validity	$< 1\%$ invalid, unavailable, or incomplete items.
Deduplication	Collapse duplicate encrypted product IDs.
Target label	One purchased-item label per session.

Table 7: Hard-candidate construction controls in **APeB**

Purchase-target validity audit. Tables 8 and 9 test whether historical preferences distinguish the purchase from viewed hard negatives through history augmentation and explicit preference extraction. The results favor history-based evidence, with the gain preserved when candidate price is removed, showing that prior interactions contain evidence for the observed choice.

Commercial-evidence audit. With queries, histories, and purchase–negative pairs fixed, Table 10 contrasts full metadata, metadata without commercial attributes, and commercial-only evidence. The

Input evidence	$> 60\%$	$> 80\%$
Query only	79	51
Query + history	93	63
Query + history, no price	91	59

Table 8: Percentage of sessions above each pairwise purchase win-rate threshold ($N = 500$); "no price" removes candidate price.

Decision context	Accuracy (%)
Query only	67
Single-round extracted preference	81
Multi-round extracted preference	83

Table 9: Effect of preference extraction on purchase identification among hard candidates.

pattern shows that commercial cues are auxiliary, while broader product attributes provide the larger discriminative signal.

Human target-recoverability audit. On the same 100 stratified sessions, two humans and GPT-5.2 assess randomized purchase–hard-negative pairs with identifiers, purchase status, and model outputs masked; GPT-4.1 supplies candidate-blind history summaries. Table 11 reports decisive-pair recovery (92% human agreement), while Table 12 provides Hit@1/5 calibration under intent and refined queries, with refinement benefiting both evaluators.

Query	GPT-5.2 Hit@1/5	Human Hit@1/5
Intent	25 / 79	34 / 83
Refined	38 / 84	40 / 86

Table 12: GPT-5.2 and human Hit@1/5 on the same 100 stratified sessions under intent and refined queries. Values are percentages.

A.5.2 Activity and Temporal Characteristics

History-activity profile. **APeB** targets active, multi-step shopping sessions with observable query refinement. Table 13 compares the lowest and highest deciles of retained history activity; their similar performance indicates that benchmark outcomes are stable across history density.

Evidence	GPT-5.2	Gemini-2.5-Pro
Full metadata	83.1	79.5
Without commercial	81.8	78.2
Commercial only	58.2	56.5

Table 10: Pairwise purchase accuracy (%) under candidate-side evidence ablations ($N = 500$); commercial denotes price, promotion, and delivery.

Metric	Human	GPT-5.2
Pairwise purchase accuracy	87.6%	83.1%
Sessions with win rate above 60%	95%	93%
Sessions with win rate above 80%	71%	63%

Table 11: Decisive-pair purchase recoverability on the 100-session stratified human-audit sample.

History activity	Avg. Hit@1 (%)
Bottom decile	23.4
Top decile	23.6

Table 13: Average Hit@1 across evaluated models for the lowest and highest deciles of retained historical interaction count on intent hard-candidate cases.

Search–purchase horizon. **APeB** uses a 30-minute search–purchase window to associate each retained session with its purchase outcome. Table 14 compares this benchmark horizon with longer, disjoint lag bins under the same alignment and candidate-construction protocol. Alignment becomes sparser beyond 30 minutes while refined queries remain stronger throughout, characterizing the retained data as exploration-heavy, short-horizon sessions with attributable purchase outcomes.

Lag	Align. (%)	Hit@1 I/R (%)
0–30 min	21.4	25.6 / 37.4
30–120 min	9.2	24.7 / 35.9
2–24 h	4.1	23.4 / 33.8

Table 14: Alignment and GPT-5.2 Hit@1 across search–purchase lags using 1,000 raw pairs per bin; I/R denotes intent/refined queries.

A.5.3 Scope and Transferability

APeB instantiates a recurring personalization structure: recover task-relevant preferences from noisy history to resolve an underspecified request and select among plausible candidates or actions. Table 15 gives representative examples of the same structure beyond product search.

Setting	Example history-conditioned decision
Mobility	Past routes and pickup habits help resolve a brief trip request and select a route or pickup point.
Outfit recommendation	Prior views, purchases, and occasions help resolve a vague style request and select garments or outfits.
News/content	Reading, viewing, and skip history helps resolve a broad information need and rank articles or clips.

Table 15: Representative examples of APeB's core task structure beyond product search: interaction history resolves an underspecified request and guides selection among plausible options.

B Experiment Setting

Due to architectural differences across *LLM-only*, *ReAct agent*, and *DeerFlow agent* settings, we adopt model-specific configurations. Unless otherwise stated, all models use their default temperature. The LLM/ReAct Hit@1/5 values in Table 2 are averaged over three runs. The detailed settings are as follows:

1. **Gemini-2.5-Pro** (`GEMINI` in Table 2): When used as a standalone LLM or within DeerFlow, we adopt the default reasoning budget. When deployed as a ReAct agent, we reduce the reasoning budget to 1024 tokens to ensure practical efficiency.

2. **GPT-4.1-Mini** (`4.1MINI`): All experiments use the default configuration.

3. **GPT-4o**: All experiments use the default configuration.

4. **Qwen3-235B** (`Qwen3-235B`): Although the model supports internal reasoning, we disable its thinking mode in all experiments due to interface limitations. Other settings remain default.

5. **DeepSeek-R1** (`R1`): Thinking mode is enabled with the default reasoning budget.

6. **GPT-4.1**: All experiments use the default configuration.

7. **GPT-5.2**: Unless specified otherwise, `GPT5.2` refers to the model configured with a minimal reasoning budget, with all other settings left unchanged.

8. **GPT-5.2Think**: Unless specified otherwise, `GPT5.2Think` refers to the GPT-5.2 condition with the default reasoning budget, while `GPT5.2` uses the minimal reasoning-budget condition; all remaining settings follow the defaults.

9. **Additional model-family runs**: Qwen3.5, Gemini3Pro, GPT5.5, and GPT5.5Think follow the same prompt, history-selection, retrieval-tool, and parsing protocols as the corresponding main-table LLM or reasoning-model conditions.

B.1 Additional Model-Family Results

Table 16 reports additional model-family results using the same query, method, and metric structure as Table 2.

B.1.1 Controlled Qwen Scale Study

To isolate model scale, we evaluate five Qwen models on the same stratified sample of 500 cases drawn from the full 5,648-case benchmark, fixing hard candidates and 60-record category-prior histories across models. All results use Single-Prompt inference and three runs. Table 17 provides the controlled comparison behind the main-text finding: scaling improves absolute Hit@1 without eliminating the intent–refined gap.

Model	Hit@1		Rubric mean	
	I	R	I	R
Qwen3.5-4B	18.2	26.1	52.0	67.3
Qwen3.5-9B	18.9	28.8	53.7	73.0
Qwen3.5-27B	20.6	31.9	57.3	79.7
Qwen3-32B	20.4	33.9	56.7	76.3
Qwen3-235B	22.8	35.9	58.3	78.3

Table 17: Controlled Qwen scaling on the shared 500-case sample from APeB. Rubric mean averages QI, PQ, and RQ; all values are three-run percentages, with I/R denoting intent/refined queries.

B.2 Statistical Reliability

Table 18 quantifies the marginal intent-query gains reported in the main text. After averaging three runs per session–method–backbone cell, we resample session identifiers 10,000 times with method and backbone pairing preserved, then macro-average differences from Single-Prompt equally across eight backbones. ReAct's 95% interval excludes zero only at Hit@1, whereas VQRA's intervals exclude zero at both Hit@1 and Hit@5.

Query	Method	Metric (%)	Qwen3.5	Gemini3Pro	GPT5.5	GPT5.5Think
Intent	**Single-Prompt**	Hit@1	22.9	24.1	25.4	25.0
		Hit@5	72.4	76.7	78.1	78.2
	ReAct	Hit@1	24.4	24.8	26.6	26.5
		Hit@5	71.3	76.9	78.6	78.1
Refined	**Single-Prompt**	Hit@1	36.1	35.1	37.2	37.3
		Hit@5	81.9	83.2	84.2	84.4
	ReAct	Hit@1	37.8	37.1	43.2	43.0
		Hit@5	82.1	83.5	84.8	84.5

Table 16: Additional model-family Hit@1/5 results under the same intent/refined hard-candidate protocol as Table 2. Values are percentages.

Method	Difference from Single-Prompt (pp)	
	Hit@1 [95% CI]	Hit@5 [95% CI]
ReAct	+1.64 [+0.7, +2.6]	+0.36 [-0.4, +1.1]
VQRA	+1.70 [+0.8, +2.6]	+1.08 [+0.3, +1.8]

Table 18: Paired session-level bootstrap macro differences for Intent hard candidates, using three-run averages, equal weighting over eight backbones, and 10,000 resamples.

Baseline	Hit@1 (%)	
	No history	With history
OpenAI embedding	15.1	13.9
Qwen3-Embedding-8B	17.7	18.0
Qwen3-Reranker-4B	9.0	14.3

Table 19: Additional embedding and reranking checks on intent hard candidates under the shared candidate protocol.

B.3 Additional Baseline Settings

We organize the additional comparisons into embedding/reranking baselines and memory-agent workflows.

B.3.1 Embedding and Reranking Baselines

We choose UniSAR (Shi et al., 2024) as the primary non-LLM PPS/SAR baseline because it is publicly available and directly models user transition behavior between search and recommendation. Table 19 reports reproducible OpenAI and Qwen3 embedding/reranking checks (Zhang et al., 2025c) under the shared intent hard-candidate protocol.

For embedding baselines, query and candidate text are encoded and ranked by similarity; the history-conditioned variant appends selected user-history evidence before scoring. For trained embedding and reranker baselines, training uses the retained search-purchase supervision available to the baseline protocol, and evaluation uses the APeB hard-candidate set. The baselines slightly differ in signal access. UniSAR and the trained embedding/reranker variants learn from historical search-purchase supervision, whereas LLM prompting and agent workflows operate primarily through in-context history evidence. UniSAR also uses cross-session negatives, which is standard in SAR/PPS training (Liu et al., 2025).

B.3.2 Memory-Agent Baselines

Mem0 and MemAgent augment an evaluated LLM backbone with an explicit memory component (Chhikara et al., 2025; Yu et al., 2025). Table 20 reports their results using an LLM mix comparable to Table 2.

B.4 Single-Prompt LLM

This subsection describes the shared direct-prompt protocol; agent-specific extensions are described in Appendix B.5 and Appendix B.6.

B.4.1 LLM Prompt

For all LLMs evaluated under the same setting, we use an identical system prompt to ensure fair comparison. The prompt is instantiated according to the scenario configuration. For example, when the input consists of a intent query with hard negative candidates, the system prompt takes the following form (details omitted for brevity):

```
TASK: Structured Product
Recommendation (Exactly 5 Items)
Your goal is to recommend
exactly five products that best
```

Query	Workflow	Metric (%)	Gemini3Pro	4.1mini	GPT4o	Qwen3.5	GPT4.1	GPT5.5
Intent	**Mem0**	Hit@1	26.1	25.5	25.9	25.1	25.8	24.1
		Hit@5	78.5	73.8	76.5	74.9	77.1	76.9
	MemAgent	Hit@1	23.8	23.7	24.2	22.5	25.0	24.0
		Hit@5	76.5	71.5	75.4	70.5	76.1	77.1
Refined	**Mem0**	Hit@1	34.3	35.4	37.5	37.9	38.2	38.1
		Hit@5	81.5	78.9	83.9	82.1	82.9	82.1
	MemAgent	Hit@1	34.2	35.8	37.3	34.2	38.6	37.9
		Hit@5	82.0	79.1	80.2	79.9	82.8	82.5

Table 20: Mem0 and MemAgent Hit@1/5 results across six backbones under intent and refined hard-candidate evaluation. Values are percentages.

```
match the user's explicit needs,
using only data from:
(1) <history> -- chronological
user interactions with indices
(action type, product in this
history, price)
(2) <query> -- the user's
current query (may be ambiguous
or intentional)
(3) <candidates> -- product
candidates (index, title, price)
____________________________________
STEP 0 -- Determine the User's
Explicit Target (Ambiguous Query
Handling)
____________________________________
(...)
____________________________________
STEP 1 -- Identify Important
History Entries Based on the
Query
____________________________________
(...)
____________________________________
STEP 2 -- Extract Explicit
Features (Summarized) & Rank
Relevance
____________________________________
(...)
____________________________________
STEP 3 -- Evaluate All Candidate
Products
____________________________________
(...)
____________________________________
STEP 4 -- Produce Exactly 5
Final Recommendations
____________________________________
(...)
____________________________________
OUTPUT SPECIFICATION (STRICT
JSON ONLY)
____________________________________
Output raw JSON conforming to
the Recommendation interface
without wrapping it in triple
back-ticks.
Type schema:
interface Product {
rank: int; // 1-5, unique
ascending
index: int; // candidate index
product_title: string; //
rewritten title (no new facts)
product_description: string; //
rewritten description
product_recommendation_reason:
List[string]; // grounded in
query + Step 0 +
summarized_features
}
interface
SupportingHistoryFeature {
summarized_feature: string;
support_history_index:
List[int];
relevance_rank: int; // from
Step 2
support_product: List[int]; //
candidate indices supported
}
interface Recommendation {
target_product: string; //
derived in Step 0
concerned_features: string; //
derived in Step 2
products: List[Product]; //
exactly 5 items
histories:
List[SupportingHistoryFeature];
// >5 entries, sorted by
relevance_rank
} End schema.
```

In other scenarios, the system prompt may undergo minor specification changes, while its overall structure remains unchanged.

The <history>, <query>, and <candidates> are supplied in the user prompt using a fixed template. For each interaction, to avoid introducing irrelevant context, only essential fields—interaction type, title, and price—are included; all identifiers and sensitive attributes are encrypted. For presentation, entity names are replaced with placeholders, and prices are slightly perturbed while preserving their scale.

```
<history>
1. Action type: order product;
Product name: [APPROVED BY A
SKINCARE LECTURER] [BRAND-A]
Water Gel Moisturizer 30g |
Brightens & Reduces Dark Spots |
Maintains Moisture for 48 Hours,
Strengthens the Skin Barrier |
Helps Reduce Acne and Soothe
Redness for Glowing Skin |
Contains Niacinamide and Vitamin
C; Price: 3.9 usd
(...)
</history>
<query>
dd cream [BRAND-B]
</query>
<candidates>
1. Product name: (DRY SKIN)
[BRAND-C] Emulsion for DRY SKIN
- Moisturizes and firms dry skin;
Price: 15.2 usd
(...)
</candidates>
```

Finally, we parse model outputs using a Pydantic-based parser and compute standard retrieval metrics, including HR@1 and HR@5.

B.4.2 History Record Selection

To control prompt length while preserving essential interaction semantics (e.g., action type, product title, and price), we limit the history to at most 60 records using an importance-aware selection mechanism. Results in Table 2 primarily adopt the category-prior strategy using the category label associated with the provided intent query.

Importance scoring. Each interaction is assigned an importance score determined by interaction type and dwell time, with stronger behavioral signals ranked as `order` > `cart` > `like` > `view`. For interactions of the same type and category, longer dwell time yields higher importance.

Time-prior selection. From the full interaction history, we first retain records whose importance exceeds a predefined ratio threshold. The final 60 records are then selected by recency within this importance-filtered subset, ensuring that retained histories emphasize both behavioral salience and temporal relevance.

Test	History input	R1	Qwen3
Assignment	True history	24.2	22.7
	Shuffled history	21.9	20.5
Modality	Product only	24.6	22.7
	Media/video only	24.1	22.3
Length	60 records	24.2	22.7
	240 records	23.8	21.9

Table 21: History-context stress tests on 500 sampled Intent hard-candidate sessions. Values are Hit@1 (%) for DeepSeek-R1 and Qwen3-235B; only the history input varies.

Category-prior selection. We first assign the provided intent query to a coarse shopping category label, such as skin care, toys, or clothing, following the category taxonomy summarized in Appendix A.5. We then include interactions whose category matches this intent-query category, up to 60 records, ranked by importance. If fewer than 60 same-category records are available, the remaining slots are filled using the time-prior selection strategy above. This approach maximizes query-relevant category evidence while retaining salient and recent interactions. This schema does not use the purchased item or reveal target-product information. We include this variant to test whether query-relevant category histories help models extract user preferences from noisy behavior records.

B.4.3 History Context Stress Tests

In Table 21, we run additional stress tests on 500 randomly selected intent hard-candidate sessions to isolate whether history gains come from useful user evidence. Shuffling histories reduces Hit@1 for both R1 and Qwen3, indicating that performance depends on coherent user-specific evidence. Modality-specific results show that product and media/video histories both provide preference signals, while increasing the context from 60 to 240 records does not improve performance. These results demonstrate that useful history selection and grounding matter more than simply exposing longer histories.

Explicit preference representation. On 500 randomly selected Intent hard-candidate sessions, a

Backbone	Hit@1 (%)	
	Raw history	Extracted prefs.
GPT-4.1-mini	23.2	24.4
Qwen3-235B	22.9	24.8
GPT-5.2	25.4	25.3

Table 22: Hit@1 (%) with raw histories versus extracted preferences across three backbones on 500 sampled Intent hard-candidate sessions.

Model	Pairwise accuracy (%)		
	Query only	Ctf.	Real
GPT-4.1	68.3	67.7	79.3
GPT-5.2	70.8	68.1	83.1
Gemini-2.5-Pro	69.1	68.5	79.5

Table 23: Pairwise preference accuracy for selecting the observed purchase over hard negatives on 500 sampled sessions. Query and candidates are fixed; Ctf. uses a length-matched history from another user.

GPT-5.2 extractor converts the query and raw pre-session history into explicit preferences with supporting history items. Candidate items and purchase status are masked from the extractor, and the resulting preferences replace the raw history in the Single-Prompt decision input. Explicit preferences increase Hit@1 performances for GPT-4.1-mini and Qwen3, while GPT-5.2 barely changes. This backbone-controlled comparison complements the staged extraction diagnostic in Table 9.

B.4.4 Contamination and Counterfactual History Check

We also probe *data contamination* through a memorization hypothesis: if decisions mainly rely on memorized query-item associations, then replacing the user's real history with unrelated counterfactual histories should have limited effect. We therefore hold the query and candidate set fixed and compare query-only, counterfactual-history, and real-history contexts on 500 sampled sessions. For each sampled session, the query and candidate set remain unchanged. In the query-only condition, the model receives no historical records. In the counterfactual condition, the true user's history is replaced with a length-matched history sampled from a different user, excluding records from the target user and session. In the real-history condition, the model receives the original user history under the same formatting and parsing protocol. We report pairwise preference accuracy in Table 23: the percentage of purchase-vs-hard-negative comparisons in which the model selects the observed purchased item.

Counterfactual histories stay close to query-only performance for all three models, while real histories improve over counterfactual histories substantially. This pattern suggests that useful performance depends on the matching user's context rather than arbitrary additional history text. The check provides evidence that memorized query-item associations are unlikely to be the dominant source of the observed history-conditioned gains. To address possible leakage concerns during data collection and evaluation, Appendix B.9 separately audits GPT-4.1's construction and diagnostic roles; GPT-4.1 does not determine Hit@K labels, and its evaluated task performance is not superior to the other evaluated models.

B.5 ReAct Agent

Our ReAct agent is implemented using the open-source LangChain framework. Compared to the single-prompt LLM setting, the system prompt follows the same high-level structure but additionally specifies the available tools and their usage constraints.

```
AVAILABLE TOOL 1 --
structured_search_tool

Purpose:
Use this tool only to retrieve
literal factual clarifications
about products or history
entries.
Permitted data sources:
- Candidate product detailed
descriptions
- History product detailed
descriptions
- Candidate product detailed
features
- History product detailed
features
- History video descriptions
(OCR/ASR)
Tool Query Rules:
1.  You must generate your own
search queries from:
- literal terms in the user
query
- literal terms relevant to
target_product or
concerned_features
- literal candidate titles or
features
```

```
- literal history text
- interpretation-critical terms
ONLY when clarifying explicit
text
4.  You may only rely on:
- explicit user inputs
- explicit tool results
```

```
AVAILABLE TOOL 2 --
web_search_tool
```

```
(...)
```

Beyond the explicit system-prompt specification, model providers (e.g., GPT and Gemini) impose additional constraints and instructions on tool invocation through their respective interfaces. We strictly follow the official usage guidelines of each provider. The user prompt is identical to that used in the single-prompt LLM setting.

Available Tools.

1. **Structured Search Tool.** In addition to item titles, **APeB** provides rich auxiliary information, including product descriptions, video OCR/ASR transcripts, and structured attributes. As these contents exceed prompt-length limits, they are organized in a fixed schema and stored in a retrieval database, which the agent queries via this tool (see Appendix B.7).

2. **Web Search Tool.** We wrap the web search API provided by YOU[2] as a callable tool. Each query returns the top-3 relevant text chunks.

3. **Crawl Tool.** We use the Jina AI crawling API[3] to fetch web pages and convert them into agent-friendly Markdown representations.

4. **Code Execution Tool.** We employ the Python REPL tool from LangChain[4], enabling the agent to execute code and observe intermediate results.

B.5.1 ReAct Failure-Source Analysis

We analyze logged ReAct traces under the same hard-candidate evaluation setting used in the main tables. In Table 24, for each trace, we report the number of reasoning rounds reached, and the AC/QC/QH/QQ diagnostic scores defined in Appendix B.8.2. Retrieval quality is measured by rubric-scored relevance to the issued query and utility to the user intent of the retrieved evidence.

Metric	Intent	Refined
Candidate retrieval relevance (QC-1)	88	92
History retrieval relevance (QH-1)	81	86
Candidate retrieval utility (QC-2)	68	80
History retrieval utility (QH-2)	49	55
Query total quantity (QQ)	8.1	8.5

Table 24: ReAct retrieval diagnostics: relevance and utility of retrieved candidate/history evidence, plus total query count.

Round	Trace ratio (%)	AC
First round	100.0	63
Second round	13.6	61
Third round	1.7	65

Table 25: Round-depth and coherence for ReAct traces. Ratio is the percentage of traces reaching each round; AC is the mean agent-coherence score for that round.

The reasonable query quantity indicates that the ReAct degradation is not primarily due to agents failing to call the available tools. The lower QH scores indicate that the harder bottleneck is formulating history-relevant queries under vague goals: agents often produce locally plausible search terms, but these terms retrieve weaker personalization evidence than candidate metadata.

Table 25 shows that most traces stop after one round, and the small subset that continues does not show improved coherence. This pattern argues against over-exploration as the main cause of degradation and indicates limited spontaneous self-correction.

We additionally report ReAct results with the explicit self-correction instruction prompt in Table 26. This produces only small, model-dependent changes across models. In Table 27, we report Hit@1 results when varying the retrieval backend. The minimal differences suggest that backend choice alone does not explain the ReAct gap. These checks indicate that a redesigned agent architecture or task-specific planner may be needed to improve personalization performance.

B.6 DeerFlow Agent System

As the original DeerFlow agent system includes components unsuitable for **APeB** evaluation (e.g., coordinators and human feedback loops), we sim-

[2]https://you.com/home
[3]https://jina.ai/
[4]https://www.langchain.com/

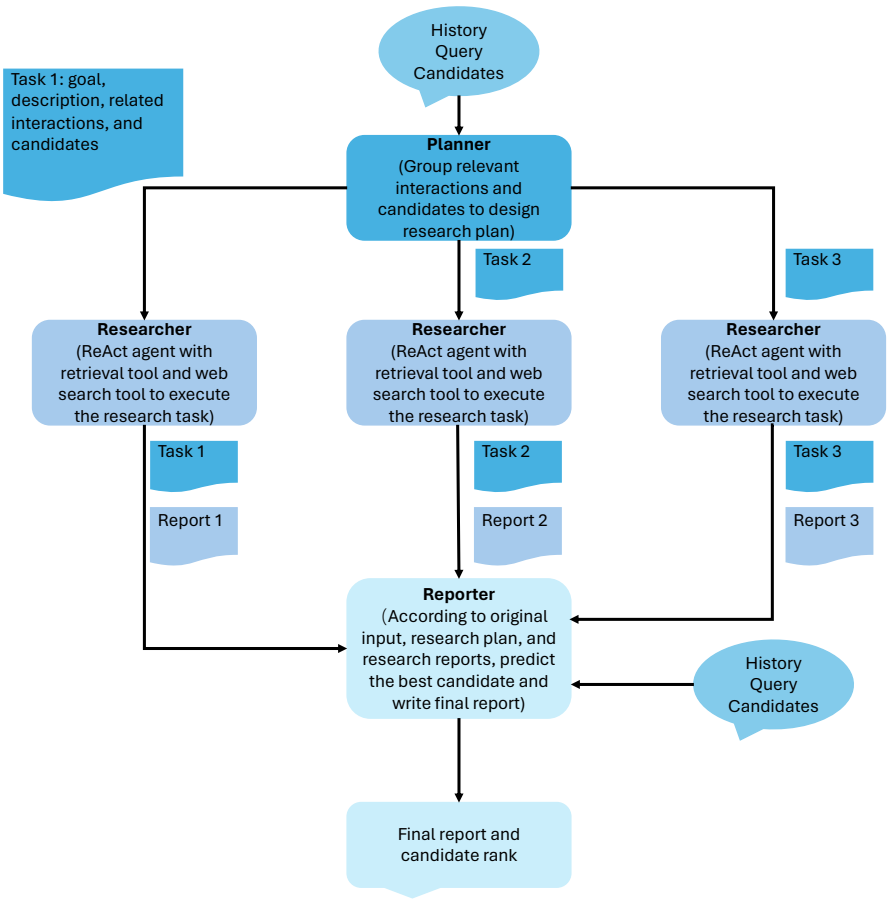


Figure 5: Overview of our modified DeerFlow agent system.

Model	Vanilla ReAct	Self-correction prompt
Qwen3	24.6	24.7
GPT-5.2	26.2	26.5
R1	24.8	25.3
Gemini	24.2	24.6
GPT-4.1	25.8	25.2

Table 26: Self-correction prompt ablation for ReAct on intent hard-candidate cases. Values are Hit@1 (%).

Backend	ReAct Hit@1 (%)		
	Qwen3	R1	GPT-4.1
BM25	24.6	24.8	25.8
OpenAI embedding	24.7	25.3	25.2

Table 27: ReAct retrieval-backend sensitivity on Intent hard-candidate cases. Backbones are Qwen3-235B, DeepSeek-R1, and GPT-4.1.

plify its architecture to the workflow illustrated in Figure 5. For all DeerFlow-based experiments, the evaluated model simultaneously assumes the roles of *Planner*, *Researcher*, and *Reporter*.

B.6.1 Planner

The Planner receives the user input—consisting of history, query, and candidates—directly as input. It is instructed to group relevant historical interactions and candidate items, and to formulate a set of research tasks aimed at information gathering and user preference investigation. The system prompt follows the form:

```
(...)
Follow these steps to generate a
research plan:
1. Understand query and
products - Understand the action
content. Reason about the
hidden pattern. Catch the
relation between the action
content, search query, and
candidates. Based on your
thinking result, infer a
grouping strategy that is
helpful for researcher to dig
out the important and detailed
features.
2. Group products - Based on
```

the grouping strategy derived from step 1, break the history interactions and candidates into different groups. In each groups, please clearly list the index of the interactions and products in this group, but you don't need to include the interaction content or product title in the task description because it will be automatically passed to researcher based on your index output.

3. **Check group size** - Consider whether the size of each group is reasonable to form a research task with medium difficulty and clear scope (a desired size will be between 6 and 10 interactions and candidates related with each other). If not, adjust the grouping strategy.

4. **Formulate research goal** - Formulate a clear and concise research goal for each group. The goal should be suitable enough to utilize the ability of researcher (they are strong models with retrieval tools).

(...)

Output raw JSON conforming to the Plan interface **without** wrapping it in triple back-ticks.

Type schema:

interface Step {

need_search: bool; // Must be explicitly set for each step

title: str;

description: str; // Specify exactly the product link and title belonging to this group, and discuss what features to collect from these products.

interactions_index_list: List[int]; // The index of the products in this group

candidates_index_list: List[int]; // The index of the candidates in this group

}

interface Plan {

has_enough_context: bool;

thought: str;

title: str;

steps: List[Step]; // Research & Processing steps to get more context

}

End schema.

(...)

B.6.2 Researcher

The Researcher agent shares the same implementation and toolset as the ReAct agent described in Section B.5. The only differences lie in the input and system prompt: the Researcher receives the research tasks generated by the Planner, and operates under the following system prompt:

Task Background

You are a **research agent** in an e-commerce recommendation system. This recommendation system is responsible for predicting what user will order for current search query based on the history interactions list and the candidate list of products. It made up of three agents:

1. **Planning agent** - Understand the user's search query, group the history interactions and candidates by feature or category, and break the overall task into sub-tasks on different group of products for the research agent.

2. **Research agent (you)** - Uses various retrieval tools to explore the product information in depth for the allocated group and analyze the feature of this group that user might be interested in or not interested in.

3. **Reporter agent** - Compile the results from the research agent and combine it with user's search query to generate a final report on how much is a candidate suitable for user and what is the most suitable candidate for user.

You are dedicated to conducting in-depth investigations about part of user history interactions and product candidates. You will be provided with a research topic, and a research description, together with a carefully selected and grouped set of history interactions and candidates on which this research topic focuses.

(...)

Output Format

- Provide a structured response in markdown format.
- Include the following sections:
- **Problem Statement**: Restate the problem for clarity.
- **Findings**: Organize your findings by topic rather than by tool used. For each major

```
finding:
- Summarize the key information
- Track the sources of
information but DO NOT include
inline citations in the text
- Conclusion: Provide a
synthesized response to the
problem based on the gathered
information.
(...)
```

B.6.3 Reporter

The Reporter agent receives the original user input, the research plan, and all research reports produced by the Researchers. It is instructed to synthesize the collected evidence into a final report and produce a ranked list of candidates. The system prompt takes the following form:

```
(...)
You should analyze the research
result, user persona, and user
history following these steps:
1. List: In the research
result of different groups,
there are lots of features that
are unique or similar to each
other. Please distinguish them
and list them out clearly. For
those similar features, you can
combine them if they share
exactly the same meaning, but
**NEVER** combine them into one
general feature because it will
make the report too general and
not able to meet user's needs.
2. Ranking: Based on user
persona and research results,
rank the primary features to
consider first when the user
issues such query content.
Notice that common features are
not necessarily to be ranked
higher, because details always
matter to the user.
3. Choosing: Based on the user
persona and ranking result,
discuss why each candidate is or
is not suitable for the user,
and present the most suitable
candidate for the user.
Finally, you should output a
rank for all candidates based on
how much it is suitable for the
user.
(...)
Output raw JSON conforming to
the Recommendation interface
without wrapping it in triple
back-ticks.
Type schema:
interface Product {
product_title: str; // The
title of the product
product_description: str; // A
detailed description of the
product to make it more
understandable
product_recommendation_reason:
List[str]; // The reasons that
this product is recommended to
the user
rank: int; // The rank of the
product in the recommendation, 1
being the most important
index: int; // The original
index of the product in the
candidates list
}
interface Recommendation {
title: str; // Main topic of
the recommendation
summary: str; // A summary that
previews the most important
points of the recommendation
products: List[Product]; // The
products that are recommended to
the user, ranked by their
importance
}
End schema.
```

B.7 Retrieval Database

As discussed earlier, each interaction or candidate in **APeB** contains rich auxiliary information. We organize this information into a retrieval database, which agents access via a retrieval tool.

Unlike conventional Retrieval-Augmented Generation (RAG) setups that partition content into fixed-length chunks and retrieve them via embedding similarity, such chunking would fragment individual interactions and degrade semantic coherence in our setting. We therefore adopt a document-level retrieval strategy, where the atomic retrieval unit corresponds to a single interaction. Each document aggregates all information associated with one interaction and is organized using the following format: The example follows the same semantic-translation, entity-placeholder, and price-perturbation convention.

```
# Detail information of one
candidate product
Product name:
[BRAND-B] [SERIES-B] Complete
Vitamin C Series - Panthenol
Anti-Pollution Radiant Creamy
Wash, Energizing Whip Foam,
Mousse Moisturizer, DD Cream
Light/Natural, Face Mist, 10%
Ferulic Acid Radiant Charge
Serum - Skincare
Price:
2.3 usd
```

```
Product description:
<p>[BRAND-B] [SERIES-B] Vitamin
C Series</p>
<p>1. [BRAND-B] [SERIES-B]
Vitamin C + Panthenol
Anti-Pollution Radiant Creamy
Wash (50 mL/100 mL)</p>
<p>Helps maintain the acid
mantle as the skin barrier's
first line of defense. Contains
vitamin C and panthenol to
brighten the skin and protect it
from pollution. (...)</p>
Categories:
Beauty & Personal Care,
Skincare, Serums & Essences,
Serums & Essences
Brand name:
[BRAND-B]
Shop name: [SHOP-B]
Attributes:
{'Volume': ['55 ml'],
'Manufacturing Country': ['ID'],
'Contains Dangerous Goods?':
['No'], 'Variant': ['Creamy
Wash 100', 'Creamy Wash 50',
'DDCream Light', 'DDCream
Natural', 'Face Mist', 'Gel
Cleanser', 'Gel Moisturizer',
'Mousse Moist', 'Serum', 'Whip
Foam 100', 'Whip Foam 50'],
'Edition': ['Regular Edition'],
'Region of Origin': ['ID'],
'Contains Alcohol or Aerosol':
['Contains Neither'], 'Quantity
Per Pack': ['1'], 'Is Suit':
['is_suit']}
```

The document format is adapted to its source type (product interaction, media interaction, or candidate). For example, interaction documents additionally include the corresponding action type.

We also consider the suitability of standard embedding-based retrieval for this setting. Empirically, we observe that agents primarily issue title-based queries when invoking the retrieval tool, using item titles to locate detailed information. Consequently, we adopt BM25—a robust lexical retrieval method—as the backend of the retrieval tool, rather than embedding-based similarity search.

B.8 Rubric Details

B.8.1 Diagnostic goal proxy summarization

We infer a behavior-grounded diagnostic goal proxy g at the session level from all interactions in Eq. 2, including issued queries, long-viewed products ($\geq$15s), and the purchased item, using GPT-4.1. The model is prompted to reconstruct the latent user intent that best explains the complete search–purchase trajectory. Specifically, GPT-4.1 outputs a concise goal proxy g that minimally and coherently aligns the observed queries with the final purchase. This proxy is used only as a reference for rubric diagnosis and does not determine Hit@K labels, which are computed from observed purchases in the ranked candidate lists.

B.8.2 Rubric perspectives

Table 28 follows the four-stage organization in Section 5.3 and summarizes the evaluated output and criterion for each stage; detailed scoring rules follow below. Diagnostic scores reported in the main analysis are averaged over Gemini-2.5-Pro, GPT-5.2, and GPT-4.1 judges; judge-sensitivity details are provided in Appendix B.8.3.

Stage	Evaluated output	Criterion and metric
Query inference	Inferred target τ	QI: query understanding and identification of the intended candidate category.
Historical preference reasoning	Preferences $\mathcal{F}$	PQ: history-grounded summary of the query-specific aspects the user cares about; PR↓/PC: first rank/count supporting the purchase.
Recommendation quality	Items and rationales $\mathcal{R}^\star, \rho$	RQ: whether the recommendations address the query and align with the user's preferences.
Agent reasoning	ReAct steps, queries, and evidence	AC: step coherence; QC/QH: candidate/history retrieval utility; QQ: query count.

Table 28: Four-stage diagnostic rubric for APeB outputs. QI/PQ/RQ/AC/QC/QH are judged scores; PR↓ is the rank of the first purchase-supporting preference, while PC/QQ count supporting preferences/search queries.

- *Query Inference (QI)* evaluates whether the inferred target τ correctly interprets the query and identifies its intended candidate category. The judge scores how well the candidate category expressed by `target_product` matches $(\texttt{user_query}, \texttt{user_goal})$ on a $[0, 100]$ scale from contradiction to direct satisfaction; the resulting `query_intent_to_target_relation_score` is the QI metric.
- *Historical Preference Reasoning (PQ/PR/PC)*
  - *Preference Quality (PQ).* PQ assesses whether the extracted preferences $\mathcal{F}$

faithfully and concisely summarize the query-specific aspects the user cares about, with each aspect supported by relevant history.

  - *Preference Salience Statistics.* We quantify the strength of historical grounding using two statistics over validated preferences: *Preference Minimum Rank (PR)*, defined as the rank π_i of the first preference supporting the purchased item (lower is better), and *preference count (PC)*, defined as the total number of distinct preferences that plausibly support the purchased item. Both are computed under strict gating rules that heavily penalize irrelevant, contradictory, or ungrounded preferences.

- *Recommendation Quality (RQ)* evaluates whether the recommended items $\mathcal{R}^\star$ and their rationales ρ address the user's query and align with the extracted preferences $\mathcal{F}$. The judge scores item relevance and rationale support, with factual consistency checked against candidate metadata; these components are aggregated into RQ.

- *Agent Reasoning (AR)*

  - *Agent Coherence (AC)* evaluates the *coherence* of each reasoning step Re_i by assessing whether the agent's explicit reasoning logically justifies its stop-or-continue decision under a ReAct-style retrieval framework. The evaluator scores the reasoning on intent understanding, task relevance, alignment with the purchased candidate, decision correctness, and logical completeness, penalizing unjustified leaps, hallucinated assumptions, or premature termination. The resulting AC score reflects the decision validity of the agent's stepwise reasoning.
  - *Candidate Query (QC)* evaluates the *quality* of each retrieved candidate by scoring the alignment between the issued search keywords, the agent's reasoning, and the retrieved item. We measure query relevance by scoring the retrieved evidence against the issued query. Query utility measures whether the keywords accurately reflect the user query and diagnostic goal proxy, are logically derived from the ReAct reasoning, and effectively retrieve candidates that are consistent with the observed purchased product. The resulting QC score captures how informative and decision-relevant the retrieval step is, penalizing generic, accidental, or misaligned candidate retrievals.
  - *History Query (QH)* evaluates the *quality* of retrieval history queries by scoring how effectively the issued search keywords retrieve decision-relevant user history. We measure query relevance by scoring the retrieved evidence against the issued query. Query utility measures whether the issued keywords retrieve history entries that provide meaningful signals for identifying the observed purchased candidate. The resulting QH score reflects the usefulness of retrieved history for personalization, penalizing generic, accidental, or weakly informative history retrievals.
  - *Query Quantity (QQ)* measures the total number of retrieval queries issued by the agent, defined as the sum of candidate queries and history queries, capturing the overall extent of retrieval tool usage during reasoning.

For the ReAct failure-source analysis in Appendix B.5.1, AC is averaged at the reasoning-step or round level, QC/QH summarize the quality of retrieved candidate/history evidence, and QQ is summarized through total query count across candidate/history across multiple steps.

B.8.3 Judge Robustness

Model	Gemini	GPT-5.2	GPT-4.1	Mean
GPT-4o	68.9	69.1	69.2	69.1
GPT-4.1-mini	64.1	64.1	64.2	64.1
Qwen3	67.3	67.2	67.3	67.3

Table 29: Judge sensitivity of mean diagnostic-rubric scores on 500 randomly sampled Intent sessions. Columns denote Gemini-2.5-Pro, GPT-5.2, and GPT-4.1 judges.

Table 29 shows nearly identical relative rankings across the three judges; human pairwise preference checks agree with the rubric preference in 96 of 100 cases. This supports using rubric scores as

diagnostic evidence while preserving Hit@K as the task-native evaluation metric.

B.8.4 Rubric Validity via Correlation with Task Metrics

The complementary judge-sensitivity evidence is summarized in Appendix B.8.3; this subsection focuses on rubric–Hit@K alignment. To evaluate output quality at the component level, we adopt an *LLM-as-a-Judge* paradigm to score model behaviors along multiple rubric dimensions (Appendix B.8). Prior work on LLM-based evaluation typically establishes validity through agreement with human annotations or inter-judge consistency (Shen et al., 2023; Min et al., 2023). While such criteria assess annotator reliability, they do not directly verify whether rubric scores faithfully reflect task success.

Following recent work on metric validity (Maynez et al., 2020; Yue et al., 2023), we instead assess our rubrics by examining their alignment with *task-native success signals*. Specifically, we use Hit@K as the reference metric family and analyze the plotted Hit@5 alignment between rubric-based scores and task success across models and agentic workflows (Figure 6). Our key assumption is that, if subtask-level rubric scores produced by a Judger LLM exhibit strong correlation with Hit@K within each model or experimental setting, these scores can be treated as supporting diagnostic evidence for diagnosing component-level reasoning failures.

Beyond correlation strength, we further examine whether rubric scores preserve *relative performance differences* across models and workflows. Figure 7 compares model-level mean rubric scores against mean Hit@5 from multiple rubric perspectives, showing that models with higher task success consistently achieve stronger rubric scores along corresponding dimensions. This consistency supports the use of rubric-based evaluation not only as a diagnostic signal within a model, but also as a meaningful basis for cross-model and cross-workflow comparison.

Together, these results support using rubric degradation as diagnostic evidence for task-level failures, while preserving Hit@K as the primary task-native outcome.

B.9 Summary of LLM Roles and Audits

This subsection consolidates the LLM roles and audit evidence reported across the main text and appendix. Table 30 serves as a cross-reference: each row identifies one role, summarizes its principal check, and points to the section containing the corresponding setup and results.

LLM use	Role and principal check	Detailed in
Query–view–purchase alignment	Construction filter; human and cross-model alignment audit ($N = 100$).	App. A.3.2
Intent/refined-query localization	Selects intent/refined indices; human exact-pair audit ($N = 100$).	App. A.3.3
Purchase validation	Pairwise recoverability diagnostic with query, history, and metadata controls ($N = 500$).	App. A.5.1
Diagnostic goal proxy	Session-derived reference for the component-level diagnostic rubric.	App. B.8.1
Rubric judging	Scores diagnostic outputs; multi-judge sensitivity and human-preference audit.	App. B.8.3
VQRA rewriting	History-conditioned query refinement evaluated through Hit@K and rubric analysis.	Sec. 5.4

Table 30: Cross-reference summary of LLM roles in APeB construction, diagnostics, and evaluation. Each row gives the role, its principal check, and the section containing the full setup and evidence.

B.10 VQRA Detail

B.10.1 Self-Rewrite Experiment

Section 5.3 and Figure 3 attribute the performance drop on intent queries primarily to weak query inference. To empirically verify this hypothesis, we construct a query-refinement pipeline. Given an intent query, which is intentionally under-specified and weakly aligned with the final purchased product, we refine it into an explicit refined query context using selected user interaction history. Selected same-category context follows the intent-query category-prior convention in Appendix B.4.2. Concretely, in the main rewrite setting, we select $k = 20$ product and media interaction records from the selected same-category history context as evidence. A shared backbone (GPT-4.1) rewrites the original intent query into a precise and discriminative keyword query, and the rewritten query is then used as the user query for personalized product retrieval. More than 92% of evaluated sessions have enough same-category histories for this context. We additionally evaluate a self-rewrite variant, where the evaluated backbone rewrites its own

query; VQRA(GPT-4.1) uses GPT-4.1 as a shared rewrite module.

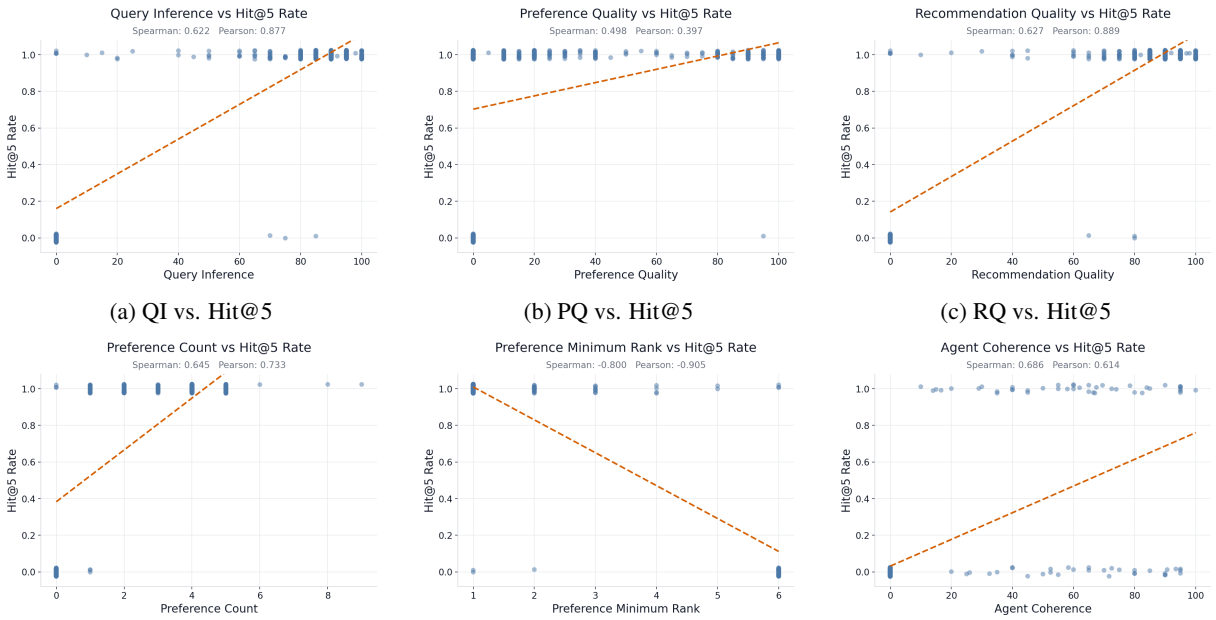
Setting	Gemini	GPT-5.2	GPT-5.2T
Vanilla	22.1	25.6	24.2
VQRA(self)	25.4	26.5	**26.6**
VQRA(GPT-4.1)	25.2	26.4	**26.9**

Table 31: Self-rewrite ablation for VQRA on intent hard-candidate cases. GPT-5.2T denotes GPT-5.2Think; both VQRA variants use the same intent-category history context.

Self-rewrite improves Hit@1 for all three reported backbones and is comparable to the GPT-4.1 rewrite variant, indicating that VQRA's gains mostly come from how the model utilizes history, not solely due to GPT-4.1 acting as an external assistant. By substituting the original intent with its refined counterpart while keeping the retrieval pipeline fixed, this design directly tests whether the model has potential to resolve query ambiguity through pipeline redesign and whether doing so can improve performance.

B.10.2 Qualitative Query-Refinement Example

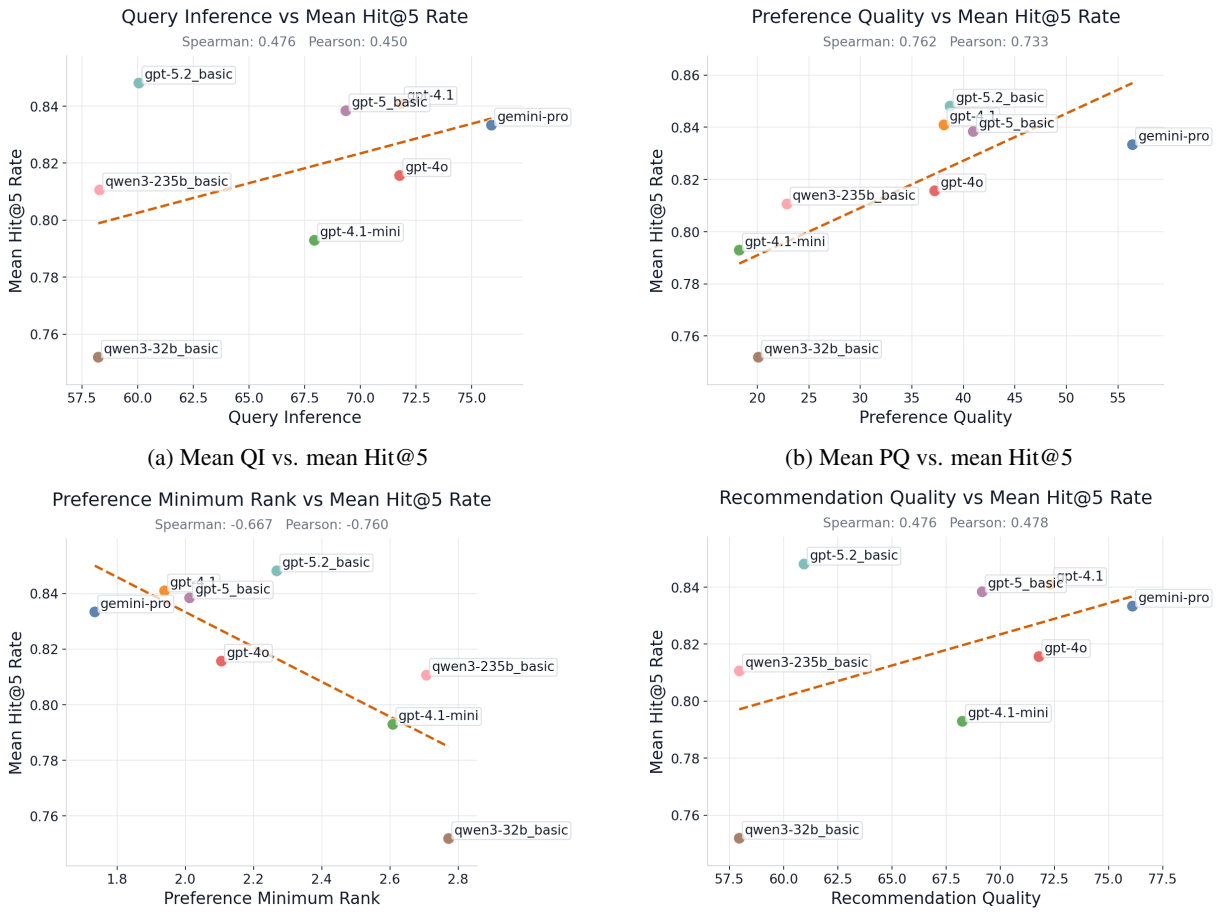
In an anonymized session, the initial query "cloves, rosemary for haircare" leaves the intended product type underspecified. VQRA uses the user's history to rewrite it as "rosemary hair growth oil" and selects the type of rosemary hair growth oil subsequently purchased by the user. Methods following the literal wording rank raw rosemary and clove ingredients.



(a) QI vs. Hit@5 (b) PQ vs. Hit@5 (c) RQ vs. Hit@5

(d) PC vs. Hit@5 (e) PR vs. Hit@5 (↓) (f) AC vs. Hit@5

Figure 6: Session-level alignment between diagnostic rubric scores and task-native Hit@5. Panels cover LLM diagnostics (QI, PQ, RQ, PC, PR) and ReAct agent coherence (AC); PR is a rank metric, so lower values indicate stronger preference salience.



(a) Mean QI vs. mean Hit@5 (b) Mean PQ vs. mean Hit@5

(c) Mean PR vs. mean Hit@5 (↓) (d) Mean RQ vs. mean Hit@5

Figure 7: Model-level rubric–Hit@5 alignment under the intent hard-candidate setting. Each point is one model backbone; panels aggregate the diagnostics used in Figure 3, showing whether cross-model task performance tracks rubric aspects.