LaPR: Latent Preference Reasoning for Generative Personalized Retrieval

Hanze Guo[1]*, Shunyu Zhang[2]*, Jiakai Tang[1], Jingwei Zhuo[5]†, Yi Zhang[2], Xuanping Li[2], Xiao Zhou[1,3,4]†

[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Kuaishou Technology
[3]Beijing Key Laboratory of Research on Large Models and Intelligent Governance
[4]Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE
[5] Unaffiliated
{ghz,xiaozhou}@ruc.edu.cn, zhuojw10@gmail.com

Abstract

Generative retrieval offers a unified paradigm for personalized search by autoregressively decoding items as Semantic IDs. However, personalized search hinges on reasoning about query-history relevance, in which explicit textual reasoning incurs heavy cost and high latency, whereas purely implicit reasoning does not provide a structured intermediate latent interface for retrieval. To resolve this tension, we propose LaPR, an adaptive latent retrieval reasoning framework that turns explicit retrieval reasoning into compact latent signals. Specifically, LaPR constructs verified step-wise retrieval traces, induces step-conditioned semantic slots via vector quantization, and distills textual reasoning into these slots through a text-latent to pure-latent curriculum. At inference, LaPR adaptively predicts the reasoning depth and corresponding slots from the query and user history, providing latent reasoning signals for efficient Semantic-ID generation. Extensive experiments show that LaPR consistently surpasses all baselines, achieving SOTA personal retrieval performance while substantially reducing reasoning-token cost and inference latency.

1 Introduction

In recent years, Semantic-ID-based generative retrieval has emerged as a promising paradigm for personalized search and recommendation (Tay et al., 2022; Wang et al., 2022b; Sun et al., 2023a; Rajput et al., 2023; Si et al., 2024; Shi et al., 2025; Chen et al., 2025a; Zhang et al., 2026a; Li et al., 2025; Zhou et al., 2025; Guo et al., 2025, 2026a; Zhang et al., 2026d; Zhu et al., 2026; Zhang et al., 2026b; Yong et al., 2026). These methods represent items as discrete Semantic-ID sequences and retrieve target items through autoregressive generation, enabling queries, user histories, and item content to be modeled within a unified generative framework.

Compared with general retrieval and recommendation, personalized search reconciles one factor: a query may correspond to a large set of potentially relevant items, while the user's historical behaviors provide only implicit preference signals that serve as a personalized filter under the current query (Bennett et al., 2012; Ai et al., 2017, 2019a; Bi et al., 2020; Bai et al., 2024; Zhou et al., 2025; Guo et al., 2025, 2026a,b; Zhang et al., 2026c). This makes query-history matching essential, requiring the model to identify relevant evidence from the user's history and transfer the matching signal to the subsequent Semantic-ID generation process. We define this process as **retrieval reasoning**, where the model identifies evidence-bearing behaviors from the user history and leverages them to infer the current retrieval intent before generating the target Semantic ID (Tang et al., 2025; Zhang et al., 2025; Jin et al., 2026; Yong et al., 2025b,a). Explicit textual rationales provide a natural way to supervise this process, as they can describe the current query intent, relevant historical behaviors, and the semantic connections among the query, history, and target item (Wei et al., 2022; Wang et al., 2022a). However, generating long textual rationales during inference is impractical for real-world industrial retrieval systems, since free-form textual reasoning substantially increases token cost and response latency (Cheng and Van Durme, 2024; Shen et al., 2025; Jin et al., 2026). On the other hand, purely implicit reasoning (Zhang et al., 2025) is computationally efficient but does not provide a structured intermediate latent interface for retrieval.

In generative retrieval, retrieval reasoning plays a central role: the model decodes Semantic IDs

*Equal contribution.
†Corresponding authors: Xiao Zhou and Jingwei Zhuo.

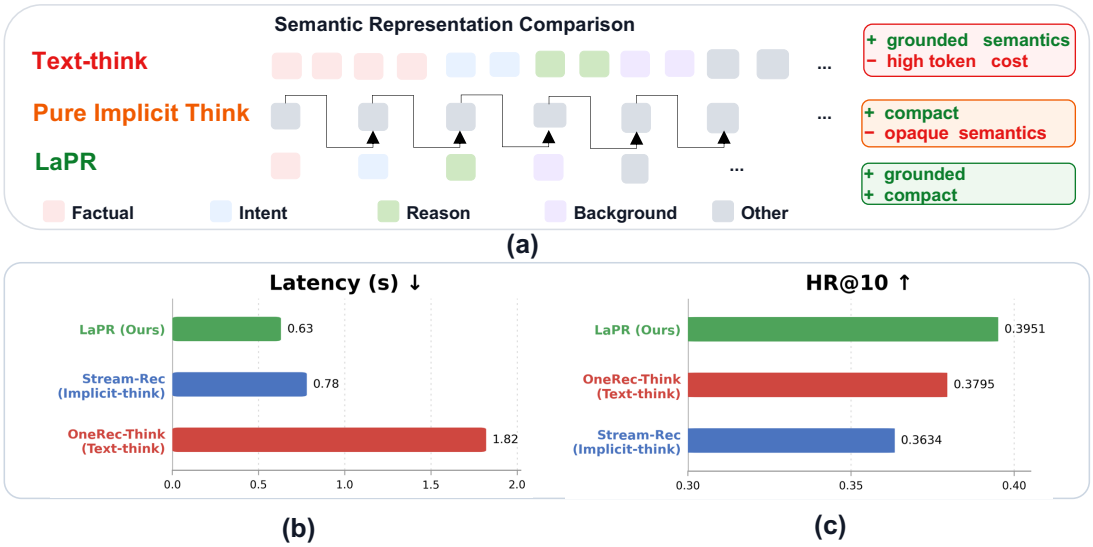

Figure 1: Overview and motivation of our latent personalized retrieval reasoning framework. Compared with explicit textual reasoning and purely implicit reasoning, LaPR uses step-conditioned semantic slots to support efficient retrieval reasoning with fewer reasoning tokens while preserving strong retrieval performance.

autoregressively without an explicit retrieval scoring step, so query-history matching evidence has to be integrated into the generation process itself. Explicit rationales provide such evidence but incur substantial token overhead at inference time, whereas purely implicit reasoning is efficient but does not expose a structured intermediate latent interface. Recent advances in latent reasoning distillation for LLM reasoning tasks offer inspiration for alleviating this tension: by compressing explicit rationales into latent states or tokens, it is possible to balance semantic information and inference efficiency to some extent (Deng et al., 2023; Pfau et al., 2024; Hao et al., 2024; Cheng and Van Durme, 2024; Shen et al., 2025; Su et al., 2025). However, in generative retrieval for personalized search, a rationale is not merely text to be compressed; it encodes the retrieval reasoning process that identifies how the current query is matched with relevant user behaviors to infer the target item. Thus, the key question is not only how to compress rationales, but how to transform such retrieval reasoning into structured latent signals that can guide Semantic-ID generation during inference

To address this challenge, we propose LaPR, a latent retrieval reasoning framework for personalized search that transforms verified step-wise retrieval reasoning traces into step-conditioned semantic slots, as shown in Figure 1. Rather than directly compressing teacher rationales as flat text sequences, LaPR first constructs verified step-wise retrieval reasoning traces that capture query intent, history-grounded evidence, and target alignment. Through vector quantization, LaPR then induces **step-conditioned semantic slots**: compact latent codes where each slot represents one reasoning step and preserves both its reasoning position and semantic content (Van Den Oord et al., 2017). Finally, LaPR adopts **text-latent to pure-latent curriculum distillation**, where textual reasoning is gradually removed during training and its logical-semantic signals are transferred into compact latent slots. During inference, the model predicts the required reasoning depth and the corresponding semantic slots solely from the current query and user history, and uses them as intermediate reasoning signals for rationale-free Semantic-ID generation, without relying on teacher rationale, or explicit textual reasoning.

The main contributions are as follows:

- We formulate **retrieval reasoning** in generative retrieval for personalized search, emphasizing how rationales link the query, user history, and the target item through evidence-grounded matching.
- We propose **LaPR**, an adaptive semantic-slot reasoning framework that induces step-conditioned semantic slots from verified reasoning traces and uses them for rationale-free Semantic-ID generation.
- Experiments show that LaPR improves retrieval effectiveness while reducing reasoning-

token cost and inference latency, with analyses validating the benefits of reasoning-depth prediction, adaptive slot prediction, and semantic slot induction.

2 Related Work

Personalized Retrieval Personalized retrieval adapts search results to individual users by leveraging historical behaviors and preferences. Early studies build user profiles from clicks, browsing history, or long-term interests to re-rank results for the current query (Teevan et al., 2005; Shen et al., 2005; Dou et al., 2007; Matthijs and Radlinski, 2011; Bennett et al., 2012). Later personalized product search methods jointly model query relevance, user preference, and item semantics through user-query-item representations, personalization-aware attention, neural user modeling, and attribute-aware matching (Ai et al., 2017, 2019a; Bi et al., 2020; Bai et al., 2024). More recently, generative retrieval represents items with discrete identifiers or Semantic IDs and formulates retrieval as sequence generation (Rajput et al., 2023), which has been extended to unified search and recommendation settings (Shi et al., 2025; Chen et al., 2025a).

Despite these advances, most existing methods do not explicitly transform query-history matching into structured intermediate signals for Semantic-ID generation. Reasoning has been explored in search and recommendation, including explainable product search and latent or slow-thinking recommendation (Ai et al., 2019b; Tang et al., 2025; Zhang et al., 2025). However, explicit textual reasoning introduces substantial token overhead, while existing implicit reasoning methods often do not expose a structured intermediate interface for retrieval. In contrast, LaPR distills the matching logic among the query, user history, and target item into structured latent slots for rationale-free SID generation.

Latent Reasoning Reasoning improves large language models on complex tasks, typically through explicit textual rationales such as chain-of-thought (Wei et al., 2022) and self-consistency decoding (Wang et al., 2022a). However, generating reasoning tokens increases inference cost and latency. To reduce this cost, recent studies explore implicit or latent reasoning, where reasoning is carried out through hidden states, special tokens, or continuous representations. Representative methods distill chain-of-thought into internal computation (Deng et al., 2023), introduce hidden or latent computation tokens (Pfau et al., 2024), or compress reasoning traces into dense or continuous representations (Hao et al., 2024; Cheng and Van Durme, 2024; Shen et al., 2025; Su et al., 2025).

Latent reasoning has also been explored in recommendation, such as ReaRec (Tang et al., 2025) and STREAM-Rec (Zhang et al., 2025). However, these methods mainly target sequential recommendation or slow-thinking recommendation, rather than query-conditioned personalized generative retrieval. LaPR differs by distilling the retrieval reasoning that links the query, user history, and target item into structured latent slots that support rationale-free Semantic-ID generation.

3 Methodology

3.1 Task Formulation

Personalized Generative Retrieval. Given a query q, a user behavior history $\mathcal{H}_u = (h_1, \ldots, h_n)$, and an item corpus $\mathcal{I}$, personalized retrieval aims to find the item i^* that matches both the current query and the user's preferences:

$$i^* = \arg\max_{i \in \mathcal{I}} p(i \mid q, \mathcal{H}_u). \tag{1}$$

In Semantic-ID-based generative retrieval, each item i is represented as a discrete sequence $\mathbf{s}_i = (s_{i,1}, \ldots, s_{i,L})$, and the target item is retrieved by autoregressively generating its Semantic ID:

$$p(\mathbf{s}_{i^*} \mid q, \mathcal{H}_u) = \prod_{t=1}^{L} p(s_{i^*,t} \mid q, \mathcal{H}_u, s_{i^*,<t}). \tag{2}$$

Retrieval Reasoning. We define *retrieval reasoning* as selecting query-relevant evidence from user history and transforming it into intermediate signals for Semantic-ID generation. Since textual rationales are costly at inference time and purely implicit reasoning does not explicitly expose structured intermediate retrieval signals, LaPR learns compact semantic slots to support rationale-free retrieval reasoning.

3.2 Overall Framework

Figure 2 illustrates the overall framework of LaPR. Given a training instance $(q, \mathcal{H}_u, i^*)$, a teacher LLM first constructs a verified step-wise retrieval reasoning trace as offline supervision. The trace captures how the current query is related to the user's historical behaviors and how such evidence supports the target item. LaPR then converts each

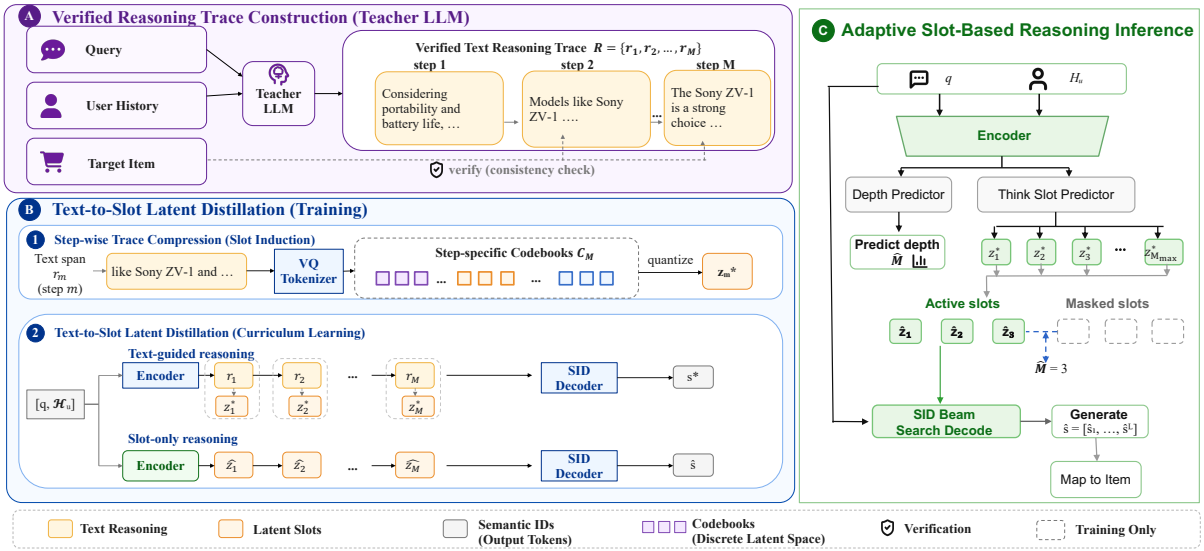

Figure 2: Overview of LaPR. A teacher LLM constructs verified retrieval-reasoning traces, which are compressed into latent semantic slots for training a parameter-shared student LLM. At inference, LaPR directly predicts adaptive slots from the query–history pair for rationale-free Semantic-ID generation.

reasoning step into a step-conditioned semantic slot, producing compact latent reasoning signals instead of using textual rationales during inference.

The student model is jointly trained from both the explicit view and the implicit latent view to perform personalized generative retrieval. It learns to estimate the required reasoning depth, predict the corresponding semantic slots from q and $\mathcal{H}_u$, and generate the target Semantic ID conditioned on the predicted slots. At inference time, the teacher and textual rationales are removed. The student only observes the query and user history, and performs rationale-free Semantic-ID generation with adaptive latent reasoning slots.

3.3 Reasoning Trace Construction

We first construct explicit reasoning traces as teacher supervision. For each training instance, the teacher receives the query q, the user behavior history $\mathcal{H}_u$, and the target item i^*. It is required to generate a structured step-wise rationale that explains how the target item can be inferred from the current query and the user's historical behaviors. The teacher is not asked to output the final item label, item title, or Semantic ID, which helps reduce direct label leakage.

To improve the quality and controllability of teacher supervision, we adopt a rubric-guided generate-then-verify procedure. The teacher is encouraged to reason over query intent, user preferences reflected by historical behaviors, attribute-level evidence, category-level evidence, and query-history-target consistency. A verifier then selects or filters the generated trace according to four constraints: *query consistency*, *history grounding*, *target alignment*, and *format validity*. Traces that fail verification are regenerated until they satisfy the constraints or reach the maximum number of attempts.

Formally, we split the verified reasoning trace into sentence-level reasoning steps, denoted as

$$\mathcal{R} = \{r_1, r_2, \ldots, r_{M^*}\}, \tag{3}$$

where r_m is the m-th sentence-level reasoning step and M^* is the teacher-derived reasoning depth. The target item and textual reasoning trace are only used for constructing training supervision. During inference, the student model only observes q and $\mathcal{H}_u$. Details of the full trace construction pipeline, including candidate generation, selection-based verification, and verification-loop prompting, are provided in Appendix A.1.

3.4 Semantic Latent Reasoning

We next describe how LaPR induces semantic slots, distills textual rationales into these slots, and predicts them adaptively.

Step-based Semantic Slot Induction Given the verified reasoning trace, LaPR discretizes each reasoning step into a semantic-slot code using a VQ-VAE tokenizer. For the m-th reasoning step, the

tokenizer selects the nearest entry from a position-specific codebook:

$$c_m^* = \arg\min_{k\in\{1,\dots,K\}} \left\| \mathrm{Enc}_\eta^{(m)}(x_m) - \mathbf{e}_{m,k} \right\|_2^2,$$
$$\mathbf{z}_m^* = \mathbf{e}_{m,c_m^*}, \quad \mathcal{C}_m = \{\mathbf{e}_{m,1},\dots,\mathbf{e}_{m,K}\}. \tag{4}$$

Here, $x_m \in R^D$ denotes the contextual semantic embedding of the m-th reasoning step, $\mathrm{Enc}_\eta^{(m)}(\cdot)$ is implemented by a shared-to-position MLP encoder, and $\mathcal{C}_m$ is the codebook for the m-th reasoning position. For an instance with M^* teacher steps, only the first M^* positions are activated, while the remaining positions are masked out. More details of the tokenizer architecture and training objective are provided in Appendix A.2.

Curriculum Latent Distillation To transfer textual reasoning into latent slots, we train the student with a dual-path distillation strategy. The first path uses a text-latent sequence, where each textual reasoning step is paired with its corresponding slot token:

$$\mathcal{Y}^{\mathrm{TL}} = \langle\mathrm{think}\rangle r_1, z_1^*, \dots, r_{M^*}, z_{M^*}^* \langle/\mathrm{think}\rangle. \tag{5}$$

The second path uses a latent-only sequence:

$$\mathcal{Y}^{\mathrm{L}} = \langle\mathrm{think}\rangle z_1^*, \dots, z_{M^*}^* \langle/\mathrm{think}\rangle. \tag{6}$$

During training, we progressively transition from text-latent reasoning to latent-only reasoning by annealing the text-latent supervision ratio with a stagewise schedule $(1.0 \to 0.5 \to 0.0)$. To preserve the semantic guidance from textual rationales, we further regularize the latent-only path with a KL loss:

$$\mathcal{L}_{\mathrm{KL}} = \mathrm{KL}\Big(p_\theta^{\mathrm{TL}}(\mathbf{z} \mid q, \mathcal{H}_u, \mathcal{Y}^{\mathrm{TL}}) \,\Big\|\, p_\theta^{\mathrm{L}}(\mathbf{z} \mid q, \mathcal{H}_u, \mathcal{Y}^{\mathrm{L}})\Big). \tag{7}$$

This encourages the latent-only path to inherit the reasoning behavior of the text-latent path, enabling the model to rely on compact latent slots during inference.

Adaptive Slot Prediction After inducing semantic slots from teacher traces, LaPR trains the student to predict latent reasoning signals from only the query q and user history $\mathcal{H}_u$. Since different instances may require different reasoning depths, the student adopts a two-level prediction strategy: it first predicts the number of reasoning steps and then predicts one semantic slot for each active step.

Specifically, the step predictor estimates:

$$p_\psi(M \mid q, \mathcal{H}_u), \quad M \in \{1,\dots,M_{\max}\}. \tag{8}$$

During training, the target depth M^* is obtained from the verified teacher trace. The depth predictor is trained with

$$\mathcal{L}_{\mathrm{step}} = -\log p_\psi\left(M^* \mid q, \mathcal{H}_u\right). \tag{9}$$

Given the target depth M^*, LaPR predicts the teacher-induced semantic slot sequence in an autoregressive manner. During training, we use teacher forcing and optimize the slot-level negative log-likelihood:

$$\mathcal{L}_{\mathrm{slot}} = -\sum_{m=1}^{M^*} \log p_\theta\left(z_m^* \mid q, \mathcal{H}_u, \mathbf{z}_{<m}^*\right). \tag{10}$$

For the text–latent path, $\mathcal{L}_{\mathrm{slot}}^{\mathrm{TL}}$ denotes the NLL over the slot-token positions in the text–latent sequence $\mathcal{Y}^{\mathrm{TL}}$. Positions beyond M^* are masked out, enabling variable-length latent reasoning.

3.5 Two-stage Optimization and Inference

LaPR is optimized in two stages. Stage 1 warms up the Semantic-ID space through SID-title alignment and title-to-SID pretraining, while Stage 2 trains latent reasoning for personalized generative retrieval.

Stage 1: SID decoder warm-up. Before latent reasoning training, we warm up the Semantic-ID decoder with both item-side and retrieval-side supervision. Specifically, we use SID-to-title reconstruction, title-to-SID generation, and query-history-to-next-item prediction to align the SID space with item semantics and initialize the decoder for personalized retrieval. The VQ-VAE tokenizer is trained offline after reasoning trace construction. After training, the tokenizer is frozen and used to assign fixed slot labels $\mathbf{z}^*$ for student optimization.

Stage 2: Latent reasoning training. The Semantic-ID generator is initialized from Stage 1 and trained for personalized retrieval. Let $P \in \{\mathrm{TL}, \mathrm{L}\}$ denote the selected training path. For the text–latent path, the rationale-token loss is

$$\mathcal{L}_{\mathrm{rat}} = -\sum_{j\in\mathcal{I}_{\mathrm{rat}}} \log p_\theta\left(y_j^{\mathrm{TL}} \mid q, \mathcal{H}_u, \mathcal{Y}_{<j}^{\mathrm{TL}}\right), \tag{11}$$

where $\mathcal{I}_{\mathrm{rat}}$ indexes rationale positions. After either path, the shared decoder predicts the target SID:

$$\mathcal{L}_{\mathrm{SID}}^P = -\sum_{t=1}^{L} \log p_\theta\left(s_{i^*,t} \mid q, \mathcal{H}_u, \mathcal{Y}^P, s_{i^*,<t}\right). \tag{12}$$

Table 1: Statistics of the datasets used in experiments.

Dataset	#User	#Item	#Query	#Inter.	AvgLen
Qilin	8.95K	183.5K	44.3K	195.0K	18.49
Amazon	192.4K	62.8K	986	1.09M	11.47

Let $\rho_\tau \in \{1.0, 0.5, 0.0\}$ be the text–latent supervision ratio at curriculum stage τ. The complete objective is

$$\begin{aligned}\mathcal{L}_{\text{stage2}} = \rho_\tau \left(\mathcal{L}_{\text{rat}} + \lambda_{\text{slot}}\mathcal{L}^{\text{TL}}_{\text{slot}} + \mathcal{L}^{\text{TL}}_{\text{SID}}\right) \\ + (1-\rho_\tau)\left(\lambda_{\text{slot}}\mathcal{L}^{\text{L}}_{\text{slot}} + \mathcal{L}^{\text{L}}_{\text{SID}}\right) \\ + \lambda_{\text{step}}\mathcal{L}_{\text{step}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}}.\end{aligned} \tag{13}$$

These terms separately supervise rationale tokens, slots, depth, and SIDs, while $\mathcal{L}_{\text{KL}}$ aligns paired text–latent and latent-only slot distributions. The objective components are detailed in Appendix A.4.

3.6 Rationale-free Inference

During inference, LaPR only observes the query q and user history $\mathcal{H}_u$. It first estimates the required reasoning depth:

$$\hat{M} = \arg\max_{M} p_\psi(M \mid q, \mathcal{H}_u). \tag{14}$$

Given $\hat{M}$, LaPR performs a two-stage beam search. It first searches over latent semantic-slot sequences $\hat{\mathbf{z}}_{1:\hat{M}}$ and then decodes Semantic-ID candidates conditioned on the selected slot beams. The top-ranked Semantic ID is mapped back to the corresponding item, avoiding the token cost and latency of explicit rationale generation.

4 Experiments

4.1 Experiment Setup

Datasets. We evaluate LaPR on two personalized generative retrieval benchmarks: Qilin and Amazon[1]. Qilin (Chen et al., 2025b) is a real-world user-centered retrieval benchmark with user behavior signals and item-side textual information. Amazon (Ai et al., 2017) refers to the Amazon Electronics subset, a personalized product search benchmark constructed from e-commerce data and widely used for evaluating personalized product retrieval. For both datasets, we follow the BM25-based hard negative sampling setting commonly used in GenSAR (Shi et al., 2025). Specifically, each test instance is evaluated with the ground-truth item and 100 BM25-sampled negative items, and all methods are compared under the same candidate set for fair evaluation. Dataset statistics are shown in Table 1.

Baselines. We compare LaPR with three groups of baselines: generic retrieval methods, including BM25 (Robertson and Zaragoza, 2009), BGE (Xiao et al., 2024), QEM (Ai et al., 2017), DSI (Tay et al., 2022), and GenRet (Sun et al., 2023a); personalized retrieval methods, including TEM (Bi et al., 2020), GenSAR (Shi et al., 2025), and OneSearch (Chen et al., 2025a). For reasoning-based baselines, we compare explicit textual reasoning methods, i.e., OneRec-Think (Liu et al., 2025) and CRS (Liu et al., 2026), with latent reasoning methods, including the pure latent retrieval baseline STREAM-REC (Zhang et al., 2025) and adapted latent reasoning methods from other domains, i.e., LASER (Jin et al., 2026), Coconut (Hao et al., 2024), and Token Assorted (Su et al., 2025). To control for confounding factors, all generative models, including OneSearch, use the same initial Semantic IDs obtained from RQ-VAE.

Evaluation and Implementation. We evaluate retrieval performance with HR@K and NDCG@K for $K \in \{1, 5, 10\}$. For generative retrieval methods, candidate Semantic IDs are generated by deterministic beam search with the same generation budget across methods and ranked by generation scores. We use Qwen3-1.7B as the student backbone for LaPR and generative baselines, and use Qwen3.5-27B as the teacher model to provide reasoning supervision for LaPR. The teacher model is used only during training and is not involved in inference. All generative baselines share the same RQ-VAE-based Semantic IDs for fair comparison. For LaPR, we set $M_{\max} = 12$ and $|\mathcal{C}| = 128$. Appendix B.1 summarizes baseline adaptations and shared decoding settings, while Appendix B provides further implementation details.

4.2 Main Experiments

Table 2 compares LaPR with representative baselines on the Qilin and Amazon personalized generative retrieval benchmarks. The compared methods cover generic retrieval, personalized retrieval, explicit reasoning, and latent reasoning baselines. We

[1]JDSearch (Liu et al., 2023) and KuaiSAR (Sun et al., 2023b) are not included because their released versions are anonymized and do not provide usable semantic identifiers (SIDs), making them incompatible with our generative retrieval setting.

Table 2: Performance comparison on personalized generative retrieval benchmarks. Best results are in bold and second-best results are underlined. * indicates statistically significant improvement over the strongest baseline in the same column under paired bootstrap resampling over test queries ($p < 0.05$).

Model	Qilin					Amazon				
	HR@1	HR@5	HR@10	NDCG@5	NDCG@10	HR@1	HR@5	HR@10	NDCG@5	NDCG@10
Group 1: Generic Retrieval Baselines										
BM25	0.0500	0.1422	0.1999	0.0965	0.1147	0.0300	0.1100	0.1767	0.0712	0.0926
BGE	0.0662	0.1977	0.2892	0.1347	0.1643	0.4030	0.6264	0.7475	0.5219	0.5613
QEM	0.0368	0.1042	0.1619	0.0692	0.0880	0.1512	0.3101	0.4657	0.2311	0.2809
DSI	0.0174	0.0598	0.1174	0.0383	0.0566	0.3558	0.5848	0.6897	0.4764	0.5103
GenRet	0.0125	0.0744	0.1362	0.0437	0.0638	0.4173	0.6513	0.7339	0.5399	0.5667
Group 2: Personalized Retrieval Baselines										
TEM	0.0208	0.0618	0.1181	0.0408	0.0589	0.0839	0.3471	0.5181	0.2173	0.2722
GenSAR	0.1185	0.2580	0.3325	0.1932	0.2173	0.5262	0.7529	0.8217	0.6485	0.6710
OneSearch	0.1479	0.2938	0.3664	0.2268	0.2471	0.6569	0.7647	0.7843	0.7169	0.7310
Group 3: Reasoning-based Personalized Retrieval Baselines										
Explicit reasoning										
OneRec-Think	0.1594	0.3215	0.3795	0.2399	0.2585	0.7158	0.8086	0.8223	0.7625	0.7672
CRS	0.1541	0.3033	0.3751	0.2354	0.2585	0.6812	0.8319	0.8434	0.7654	0.7691
Latent reasoning										
STREAM-REC	0.1450	0.2904	0.3634	0.2232	0.2467	0.6748	0.7932	0.8107	0.7456	0.7564
LASER	0.1398	0.2792	0.3511	0.2150	0.2382	0.6630	0.7747	0.7952	0.7298	0.7409
CoConut	0.1486	0.2966	0.3679	0.2287	0.2518	0.6756	0.8004	0.8172	0.7547	0.7624
Token Assorted	0.1502	0.2998	0.3715	0.2314	0.2546	0.6789	0.8043	0.8214	0.7581	0.7665
Ours	**0.1693***	**0.3287***	**0.3951***	**0.2528***	**0.2743***	**0.7198***	**0.8654***	**0.8789***	**0.7978***	**0.8021***

evaluate all methods using HR@K and NDCG@K to measure both hit accuracy and ranking quality under the same generative retrieval setting.

Overall, LaPR consistently achieves the best performance across all metrics on both datasets. On Qilin, compared with the strongest baseline in each column, LaPR improves HR@1, HR@5, HR@10, NDCG@5, and NDCG@10 by 6.2%, 2.2%, 4.1%, 5.4%, and 6.1%, respectively. It also outperforms the strongest non-reasoning personalized baseline OneSearch by 7.8%–14.5% (using same Semantic ID), demonstrating the benefit of introducing intermediate retrieval reasoning beyond standard personalized retrieval. On Amazon, LaPR further improves over the strongest baseline by 0.6%, 4.0%, 4.2%, 4.2%, and 4.3% on the five metrics, respectively, showing that the gains generalize across different personalized retrieval scenarios.

These gains suggest that LaPR provides a more suitable intermediate representation for personalized generative retrieval. Explicit reasoning baselines expose query-history evidence through free-form rationales but incur high token and latency costs, while pure latent reasoning methods are efficient but lack retrieval-oriented step supervision. Meanwhile, LaPR's full-corpus results are reported in Appendix E.3.

4.3 Ablation Study

Table 3 reports a compact ablation study of LaPR, with full results provided in Appendix E.1. We include the base model as a reference and ablate the key components of LaPR, including thinking traces, inference-time semantic slots, adaptive reasoning depth, KL regularization, slot supervision, and the text-to-slot curriculum.

The ablation results reveal two main findings. First, retrieval-oriented latent reasoning is effective: LaPR substantially outperforms the base model on both datasets, while *w/o think* still lags behind LaPR, indicating that task supervision alone provides only limited query-history matching signals and that verified thinking traces offer useful reasoning supervision. Second, the slot-based design is necessary: removing inference-time slots or using fixed slots consistently degrades performance, showing that semantic slots act as effective intermediate variables and that adaptive reasoning depth is beneficial. The drops caused by removing KL regularization, slot supervision, or the text-to-slot curriculum further suggest that stable text-to-latent transfer requires explicit slot supervision and gradual distillation. Meanwhile, target-blind trace-generation ablation results are reported in Appendix E.4.

Table 3: Ablation results of the main components in LaPR. Full results are provided in Appendix E.1.

Model	Qilin		Amazon	
	HR@10	NDCG@10	HR@10	NDCG@10
Base	0.3642	0.2476	0.8089	0.7535
w/o think	0.3764	0.2545	0.8374	0.7771
w/o infer slots	0.3827	0.2645	0.8657	0.7942
fixed 4 slots	0.3910	0.2715	0.8751	0.7997
w/o KL reg.	0.3834	0.2643	0.8712	0.7977
w/o slot loss	0.3781	0.2617	0.8638	0.7927
w/o curriculum	0.3735	0.2585	0.8586	0.7892
LaPR	**0.3951**	**0.2743**	**0.8789**	**0.8021**

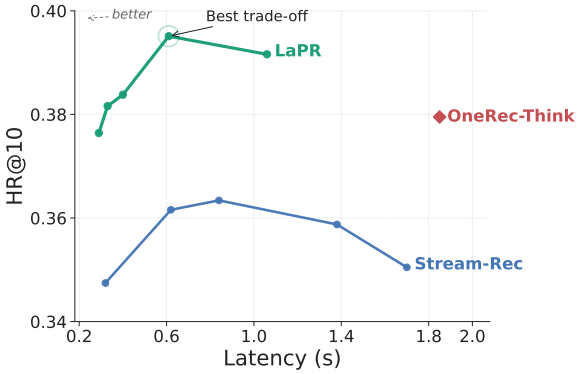

Figure 3: Effectiveness-efficiency trade-off across reasoning modes on Qilin. LaPR achieves a favorable Pareto frontier by using compact semantic slots, improving HR@10 with substantially lower latency than explicit textual reasoning and pure latent reasoning.

4.4 Effectiveness-Efficiency Trade-off

Adaptive Reasoning Budget. We analyze the effectiveness-efficiency trade-off of LaPR by varying the number of latent reasoning steps and comparing it with pure latent reasoning and explicit textual reasoning. As shown in Figure 3, the no-reasoning variant achieves an HR@10 of 0.3764, whereas the full LaPR with adaptive depth prediction reaches 0.3951. The full model sets $M_{\max} = 12$ and predicts 4.7 reasoning slots on average. Compared with STREAM-REC, LaPR achieves higher HR@10 with lower latency under comparable reasoning depths, while STREAM-REC incurs higher latency. Compared with OneRec-Think, LaPR avoids generating long textual rationales and reduces latency from 1.85s to 0.61s while achieving stronger HR@10. These results demonstrate that compact and adaptive latent reasoning enables a favorable Pareto frontier between effectiveness and efficiency. Sensitivity results for $M_{\max}$ and the codebook size are reported in Appendix E.5.

4.5 Latent Slot Structure and Post-hoc Diagnostics

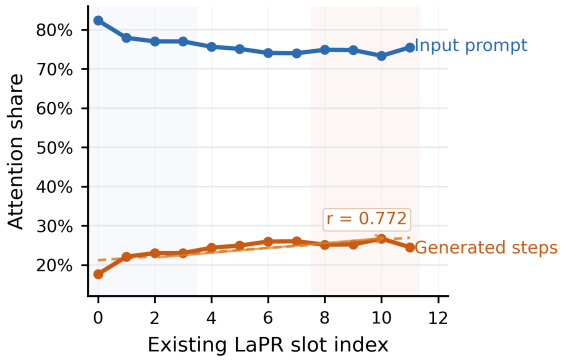

Figure 4: Attention source decomposition across LaPR slot positions. Early slots attend primarily to the input prompt, while later slots allocate an increasing share of attention to previously generated slots, indicating a progressive latent reasoning process.

Attention Analysis of Latent Thinking. To examine whether LaPR slots form a progressive reasoning process, we decompose each generated slot's causal attention into two sources: the input prompt before <think> and previously generated latent slots. As shown in Figure 4, slots consistently attend more to the input prompt, indicating that latent reasoning remains grounded in query-history information. Meanwhile, attention to previous slots increases with the slot index and is positively correlated with position (Pearson $r = 0.772$ on test samples), suggesting that later slots build on earlier latent states rather than independently re-encoding the input. These results indicate a progressive latent structure, where early slots extract input-grounded intent signals and later slots refine them through intermediate latent reasoning.

Post-hoc Slot Analysis. To examine the structure associated with different latent slots, we retrieve their nearest-neighbor text representations as a post-hoc diagnostic. Table 4 shows a representative personalized retrieval case from the Qilin dataset, where the query, user history, answer, and decoded slot semantics are translated into English for readability. We interpret each slot through a K-nearest-neighbor lookup over a vocabulary hidden-state lexicon and summarize the retrieved neighbors into semantic keywords, which serve as approximate projections of latent slots rather than explicit textual explanations. This representation reflects latent slots as compressed semantic states

Table 4: Post-hoc KNN projections of predicted latent slots in a personalized retrieval case.

Field / Slot	Text or KNN-projected Keywords
Query	User searches for outdoor campsites in Chongqing that support barbecue.
History	Guangyang Island one-day trip guide; Chongqing downtown Guangyang Island guide; Chongqing travel itinerary; winter outfit and travel posts.
z_1/015	{Chongqing, barbecue, campsite, outdoor, travel}
z_2/037	{Chongqing, free, campsite, Mutai, sand pit, tadpoles, outdoor, travel}
z_3/091	{2024, Chongqing, six sites, free, campsite, collection, Mutai, sand pit, tadpoles}
z_4/057	{six free campsites in Chongqing, barbecue facilities}
Answer	2024 collection of six free campsites in Chongqing: Mutai, Sand Pit, Tadpoles.

rather than explicit textual explanations. The case shows that LaPR slots capture role-aware retrieval semantics: earlier slots encode query- and history-related signals, while later slots focus on attribute- and target-level matching for Semantic-ID generation.

5 Conclusion

We proposed LaPR, a latent reasoning framework for generative personalized retrieval. Instead of relying on explicit textual rationales or a single opaque latent vector, LaPR distills offline teacher rationales into step-conditioned semantic slots and uses adaptive depth prediction to support compact, rationale-free SID generation. Experiments show that the resulting latent-slot representation improves retrieval effectiveness while reducing reasoning overhead.

Limitations

LaPR focuses on improving the effectiveness–efficiency trade-off of reasoning-enhanced personalized generative retrieval. Our main evaluation follows the hard-negative candidate protocol commonly used in prior personalized retrieval studies, enabling controlled comparisons across retrieval and generative baselines. Although we additionally reports full-corpus results on the two benchmark collections, these collections remain substantially smaller and less dynamic than production-scale search corpora. Evaluating LaPR on web-scale, continuously updated item collections under realistic latency and resource constraints remains an important direction for future work.

LaPR also uses teacher-generated reasoning traces as offline supervision. The teacher is removed at inference time, and the student only observes the query and user history, but the quality of slot induction can still benefit from stronger or more diverse teachers. Finally, our experiments are conducted on two representative personalized retrieval benchmarks. Future work can further study LaPR under broader domains, longer histories, multilingual queries, and dynamically updated item corpora.

Ethical Considerations

This work uses user behavior histories for personalized retrieval, which may involve privacy and bias risks. Any real-world deployment should follow data protection regulations, obtain proper user consent, and apply anonymization and access-control mechanisms. LaPR is designed to improve retrieval relevance, not to infer sensitive user attributes or make decisions about individuals. The teacher-generated reasoning traces are used only for offline training and are not exposed during inference.

Acknowledgments

This work was supported by the Beijing Nova Program (Grant No. 202604841294).

References

Qingyao Ai, Daniel N Hill, SVN Vishwanathan, and W Bruce Croft. 2019a. A zero attention model for personalized product search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 379–388.

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*, pages 645–654.

Qingyao Ai, Yongfeng Zhang, Keping Bi, and W Bruce Croft. 2019b. Explainable product search with a dynamic relation embedding model. *ACM Transactions on Information Systems (TOIS)*, 38(1):1–29.

Yutong Bai, Zhicheng Dou, and Ji-Rong Wen. 2024. Learning dynamic multi-attribute interest for personalized product search. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2984–2993.

Paul N Bennett, Ryen W White, Wei Chu, Susan T Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. 2012. Modeling the impact of short-and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 185–194.

Keping Bi, Qingyao Ai, and W Bruce Croft. 2020. A transformer-based embedding model for personalized product search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1521–1524.

Ben Chen, Xian Guo, Siyuan Wang, Zihan Liang, Yue Lv, Yufei Ma, Xinlong Xiao, Bowen Xue, Xuxin Zhang, Ying Yang, and 1 others. 2025a. Onesearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search. *arXiv preprint arXiv:2509.03236*.

Jia Chen, Qian Dong, Haitao Li, Xiaohui He, Yan Gao, Shaosheng Cao, Yi Wu, Ping Yang, Chen Xu, Yao Hu, and 1 others. 2025b. Qilin: A multimodal information retrieval dataset with app-level user sessions. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3670–3680.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*.

Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*, pages 581–590.

Hanze Guo, Jianxun Lian, and Xiao Zhou. 2026a. Why not collaborative filtering in dual view? bridging sparse and dense models. *ACM Transactions on Information Systems*, 44(3):1–24.

Hanze Guo, Yijun Ma, and Xiao Zhou. 2025. Sorex: Towards self-explainable social recommendation with relevant ego-path extraction. *ACM Transactions on Information Systems*, 44(2):1–27.

Hanze Guo, Jing Yao, Xiao Zhou, Xiaoyuan Yi, and Xing Xie. 2026b. Counterfactual reasoning for steerable pluralistic value alignment of large language models. *Advances in Neural Information Processing Systems*, 38:122128–122169.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.

Jiajie Jin, Yanzhao Zhang, Mingxin Li, Dingkun Long, Pengjun Xie, Yutao Zhu, and Zhicheng Dou. 2026. Laser: Internalizing explicit reasoning into latent space for dense retrieval. *arXiv preprint arXiv:2603.01425*.

Lei Li, Xiangxu Zhang, Xiao Zhou, and Zheng Liu. 2025. AutoMIR: Effective zero-shot medical information retrieval without relevance labels. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 24028–24047, Suzhou, China. Association for Computational Linguistics.

Jiongnan Liu, Zhicheng Dou, Guoyu Tang, and Sulong Xu. 2023. 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*, pages 2945–2952.

Zhanyu Liu, Shiyao Wang, Xingmei Wang, Rongzhou Zhang, Jiaxin Deng, Honghui Bao, Jinghao Zhang, Wuchao Li, Pengfei Zheng, Xiangyu Wu, and 1 others. 2025. Onerec-think: In-text reasoning for generative recommendation. *arXiv preprint arXiv:2510.11639*.

Zhiding Liu, Ben Chen, Mingyue Cheng, Enhong Chen, Li Li, Chenyi Lei, Wenwu Ou, Han Li, and Kun Gai. 2026. Towards context-aware reasoning-enhanced generative searching in e-commerce. In *Proceedings of the ACM Web Conference 2026*, pages 6551–6561.

Nicolaas Matthijs and Filip Radlinski. 2011. Personalizing web search using long term browsing history. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 25–34.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Jacob Pfau, William Merrill, and Samuel R Bowman. 2024. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, and 1 others. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc.

Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 824–831.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693.

Teng Shi, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Yang Song, and Enyun Yu. 2025. Gensar: Unifying balanced search and recommendation with generative retrieval. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, pages 124–134.

Zihua Si, Zhongxiang Sun, Jiale Chen, Guozhang Chen, Xiaoxue Zang, Kai Zheng, Yang Song, Xiao Zhang, Jun Xu, and Kun Gai. 2024. Generative retrieval with semantic tree-structured identifiers and contrastive learning. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 154–163.

DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. 2025. Token assorted: Mixing latent and text tokens for improved language model reasoning. *arXiv preprint arXiv:2502.03275*.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. 2023a. Learning to tokenize for generative retrieval. *Advances in Neural Information Processing Systems*, 36:46345–46361.

Zhongxiang Sun, Zihua Si, Xiaoxue Zang, Dewei Leng, Yanan Niu, Yang Song, Xiao Zhang, and Jun Xu. 2023b. Kuaisar: A unified search and recommendation dataset. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 5407–5411.

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

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, and 1 others. 2022. Transformer memory as a differentiable search index. *Advances in neural information processing systems*, 35:21831–21843.

Jaime Teevan, Susan T Dumais, and Eric Horvitz. 2005. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456.

Aaron Van Den Oord, Oriol Vinyals, and 1 others. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022a. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, and 1 others. 2022b. A neural corpus indexer for document retrieval. *Advances in Neural Information Processing Systems*, 35:25600–25614.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649.

Xixian Yong, Jianxun Lian, Xiaoyuan Yi, Xiao Zhou, and Xing Xie. 2025a. Motivebench: How far are we from human-like motivational reasoning in large language models? In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pages 20059–20089. Association for Computational Linguistics.

Xixian Yong, Peilin Sun, Zihe Wang, and Xiao Zhou. 2026. Intelli-planner: Towards customized urban planning via large language model empowered reinforcement learning. In *Proceedings of the ACM Web Conference 2026, WWW 2026, Dubai, United Arab Emirates, originally scheduled for April 13-17, 2026, rescheduled for June 29 - July 3, 2026*, pages 9385–9396. ACM.

Xixian Yong, Xiao Zhou, Yingying Zhang, Jinlin Li, Yefeng Zheng, and Xian Wu. 2025b. Think or not? exploring thinking efficiency in large reasoning models via an information-theoretic lens. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*.

Junjie Zhang, Beichen Zhang, Wenqi Sun, Hongyu Lu, Wayne Xin Zhao, Yu Chen, and Ji-Rong Wen. 2025. Slow thinking for sequential recommendation. *arXiv preprint arXiv:2504.09627*.

Xiangxu Zhang, Lei Li, Xiao Zhou, and Zheng Liu. 2026a. R2med: A benchmark for reasoning-driven medical retrieval. *Preprint*, arXiv:2505.14558.

Xiangxu Zhang, Lei Li, Yanyun Zhou, Xiao Zhou, Yingying Zhang, and Xian Wu. 2026b. Inflated excellence or true performance? rethinking medical diagnostic benchmarks with dynamic evaluation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), pages 26454–26493, San Diego, California, United States. Association for Computational Linguistics.

Xiangxu Zhang, Jiamin Wang, Qinlin Zhao, Hanze Guo, Linzhuo Li, Jing Yao, Xiao Zhou, Xiaoyuan Yi, and Xing Xie. 2026c. Human values matter: Investigating how misalignment shapes collective behaviors in llm agent communities. *Preprint*, arXiv:2604.05339.

Xiangxu Zhang, Xiao Zhou, Hongteng Xu, and Jianxun Lian. 2026d. Hypemed: Enhancing medication recommendations with hypergraph-based patient relationships. *ACM Trans. Inf. Syst.*, 44(4).

Xiao Zhou, Zhongxiang Zhao, and Hanze Guo. 2025. Tricolore: Multi-behavior user profiling for enhanced candidate generation in recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 37(7):4349–4360.

Yanxu Zhu, Shitong Duan, Xiangxu Zhang, Jitao Sang, Peng Zhang, Tun Lu, Xiao Zhou, Jing Yao, Xiaoyuan Yi, and Xing Xie. 2026. Mohobench: Assessing honesty of multimodal large language models via unanswerable visual questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 29205–29213.

A Methods Details

A.1 Reasoning Trace Construction

For each training instance, the teacher generates a structured reasoning trace enclosed by special thinking markers:

$$\langle \text{think} \rangle r_1, r_2, \ldots, r_{M^*} \langle / \text{think} \rangle. \tag{15}$$

Trace construction is an offline, candidate-aware supervision procedure. The teacher is given the query, user history, and a small candidate set containing the target item, and is instructed to write intermediate retrieval reasoning steps for selecting the target-compatible candidate. It must not output the final SID or use the answer as a generated label. The resulting textual trace is used only to induce semantic-slot supervision; during training and inference, the student predicts slots from the query and user history, and at inference time it does not observe the target item, candidate titles, or teacher rationale.

Candidate generation. For each instance, we sample five candidate traces. The candidate prompt asks the teacher to cover the following retrieval roles when applicable: query intent, relevant user-history evidence, preference abstraction, attribute-level matching, category-level matching, and target compatibility. The number of steps is allowed to vary up to $M_{\max}$.

Verification. A verifier checks whether the teacher-selected candidate matches the target index and whether the trace satisfies four quality rubrics:

- **Query consistency**: the trace should explain the current query rather than only summarizing the user's history.
- **History grounding**: preference claims should be supported by observed historical behaviors.
- **Target alignment**: the trace should connect the query and history to the target item without copying the answer as a label.
- **Format validity**: the output should follow the step-wise format and stay within the maximum depth.

If no candidate is valid, the teacher regenerates candidates until a valid trace is selected or the retry limit is reached. Instances without a valid trace are trained with SID supervision only. The verifier is used only for offline trace filtering and is not used during validation or test inference.

A.2 Details of Step-conditioned Semantic Slot Induction

For the m-th reasoning step r_m, we construct a contextual semantic string x_m and obtain its continuous representation with a frozen mean-pooling encoder:

$$\mathbf{h}_m = \text{MeanPool}\left(\text{Enc}_{\text{frozen}}(x_m)\right). \tag{16}$$

The encoder is kept fixed during tokenizer training, providing a stable semantic space for latent code induction.

The VQ-VAE tokenizer adopts a shared-to-individual MLP architecture:

$$\mathbf{u}_m = g_{\text{sh}}(\mathbf{h}_m), \quad \mathbf{g}_m = g_m(\mathbf{u}_m), \tag{17}$$

where g_{sh} denotes the shared lower MLP layers and g_m denotes the position-specific projection head for the m-th reasoning position. Each position has an independent codebook:

$$\mathcal{C}_m = \{\mathbf{e}_{m,1}, \ldots, \mathbf{e}_{m,K}\}. \tag{18}$$

In our implementation, we set $K = 128$.

The discrete slot is obtained by nearest-neighbor quantization:

$$z_m^* = \arg\min_{k \in \{1,\ldots,K\}} \|\mathbf{g}_m - \mathbf{e}_{m,k}\|_2^2, \tag{19}$$

and the corresponding quantized representation is:

$$\mathbf{v}_m = \mathbf{e}_{m,z_m^*}. \tag{20}$$

The tokenizer is trained with a token-level reconstruction objective:

$$\mathcal{L}_{\text{rec}} = -\sum_{m=1}^{M^*} \sum_{t=1}^{|x_m|} \log p_\omega(x_{m,t} \mid \mathbf{v}_m, x_{m,<t}). \tag{21}$$

We also use the standard VQ commitment objective:

$$\begin{aligned} \mathcal{L}_{\text{vq}} = \sum_{m=1}^{M^*} & \|\text{sg}[\mathbf{g}_m] - \mathbf{v}_m\|_2^2 \\ & + \beta \|\mathbf{g}_m - \text{sg}[\mathbf{v}_m]\|_2^2, \end{aligned} \tag{22}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operation. The overall tokenizer objective is:

$$\mathcal{L}_{\text{tok}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{vq}} \mathcal{L}_{\text{vq}}. \tag{23}$$

A.3 Details of SID Decoder Warm-up

Before latent reasoning training, we warm up the Semantic-ID decoder with item-side and retrieval-side supervision. For each item i, let x_i denote its title or description, and let $\mathbf{s}_i = (s_{i,1}, \ldots, s_{i,L})$ denote its Semantic ID sequence. We use three complementary objectives.

First, we use a SID-to-title objective to encourage Semantic IDs to preserve item-side textual semantics:

$$\mathcal{L}_{\text{SID}\to\text{title}} = -\sum_{t=1}^{|x_i|} \log p_\theta(x_{i,t} \mid \mathbf{s}_i, x_{i,<t}). \quad (24)$$

Second, we train the decoder to generate the Semantic ID from the item title or description:

$$\mathcal{L}_{\text{title}\to\text{SID}} = -\sum_{\ell=1}^{L} \log p_\theta(s_{i,\ell} \mid x_i, s_{i,<\ell}). \quad (25)$$

Third, to adapt the decoder to the personalized retrieval setting, we further train it to predict the next-item Semantic ID from the query and user history. Given a query q, user behavior history $\mathcal{H}_u$, and target item i^*, the retrieval warm-up loss is:

$$\mathcal{L}_{\text{next}} = -\sum_{\ell=1}^{L} \log p_\theta(s_{i^*,\ell} \mid q, \mathcal{H}_u, s_{i^*,<\ell}). \quad (26)$$

The overall warm-up objective is:

$$\mathcal{L}_{\text{stage1}} = \lambda_1 \mathcal{L}_{\text{SID}\to\text{title}} + \lambda_2 \mathcal{L}_{\text{title}\to\text{SID}} + \lambda_3 \mathcal{L}_{\text{next}}. \quad (27)$$

The first two objectives align item textual semantics with the discrete SID space, while the third objective initializes the decoder for query- and history-conditioned personalized retrieval.

A.4 Details of Text-to-Slot Curriculum Distillation

Given a verified reasoning trace $\mathcal{R} = \{r_1, \ldots, r_{M^*}\}$ and its corresponding slot sequence $\mathbf{z}^* = (z_1^*, \ldots, z_{M^*}^*)$, we construct two supervision paths. The text-latent path interleaves each textual reasoning step with its corresponding slot token:

$$\mathcal{Y}^{\text{TL}} = \langle\text{think}\rangle r_1, z_1^*, \ldots, r_{M^*}, z_{M^*}^* \langle/\text{think}\rangle. \quad (28)$$

The latent-only path removes textual rationales and retains only the induced slot tokens:

$$\mathcal{Y}^{\text{L}} = \langle\text{think}\rangle z_1^*, \ldots, z_{M^*}^* \langle/\text{think}\rangle. \quad (29)$$

Let $c = (q, \mathcal{H}_u)$ denote the query–history context, and let $\mathbf{s}^* = (s_1^*, \ldots, s_L^*)$ denote the target Semantic ID.

The text–latent path applies token-level NLL to the natural-language rationale positions:

$$\mathcal{L}_{\text{rat}} = -\sum_{j \in \mathcal{I}_{\text{rat}}} \log p_\theta \left(y_j^{\text{TL}} \mid c, \mathcal{Y}_{<j}^{\text{TL}} \right), \quad (30)$$

where $\mathcal{I}_{\text{rat}}$ indexes only rationale-token positions.

For either training path $P \in \{\text{TL}, \text{L}\}$, the slot-token loss is

$$\mathcal{L}_{\text{slot}}^{P} = -\sum_{m=1}^{M^*} \log p_\theta^{P} \left(z_m^* \mid c, \mathcal{Y}_{<z_m}^{P} \right). \quad (31)$$

The depth predictor is supervised by the teacher-derived trace length:

$$\mathcal{L}_{\text{step}} = -\log p_\psi \left(M^* \mid c \right). \quad (32)$$

After either reasoning path, the shared SID decoder is trained with

$$\mathcal{L}_{\text{SID}}^{P} = -\sum_{\ell=1}^{L} \log p_\theta \left(s_\ell^* \mid c, \mathcal{Y}^{P}, \mathbf{s}_{<\ell}^* \right). \quad (33)$$

When paired text–latent and latent-only slot logits are available, their distributions are aligned using

$$\mathcal{L}_{\text{KL}} = \sum_{m=1}^{M^*} \text{KL}\Big(p_\theta^{\text{TL}} \left(z_m \mid c, \mathcal{Y}_{<z_m}^{\text{TL}} \right) \| \, p_\theta^{\text{L}} \left(z_m \mid c, \mathcal{Y}_{<z_m}^{\text{L}} \right) \Big). \quad (34)$$

The curriculum uses the text–latent supervision ratios $\rho_\tau \in \{1.0, 0.5, 0.0\}$. Table 5 summarizes the corresponding training and inference roles.

Thus, rationale-token prediction, slot prediction, depth estimation, SID generation, and cross-path alignment constitute distinct supervision signals. At inference time, the slot tokenizer, teacher traces, text–latent path, and training-only losses are removed.

A.5 Details of Rationale-free Inference

At inference time, LaPR removes the teacher model and textual rationales. Given only the query q and user history $\mathcal{H}_u$, it first predicts the reasoning depth:

$$\hat{M} = \arg\max_{M} p_\psi(M \mid q, \mathcal{H}_u). \quad (35)$$

Table 5: Decomposition of the LaPR training objective. TL and L denote the text–latent and latent-only paths.

Component	Loss	Supervision	Parameter sharing	Inference status
Offline slot tokenizer	$\mathcal{L}_{\mathrm{tok}}$	Teacher steps and quantized slot targets	Separate tokenizer parameters	Removed
Rationale prediction	$\mathcal{L}_{\mathrm{rat}}$	Natural-language tokens in the TL path	Shared student backbone	Removed
Slot prediction	$\mathcal{L}_{\mathrm{slot}}^{P}$	Positioned slot tokens in the TL and L paths	Shared backbone and slot embeddings	L path retained
Depth prediction	$\mathcal{L}_{\mathrm{step}}$	Teacher-derived trace length M^*	Separate depth head over shared states	Retained
SID generation	$\mathcal{L}_{\mathrm{SID}}^{P}$	Target Semantic ID	Shared SID decoder	Used with L path
Cross-path alignment	$\mathcal{L}_{\mathrm{KL}}$	Paired TL and L slot distributions	No additional parameters	Removed

Conditioned on $\hat{M}$, LaPR performs two-stage beam search. The first stage searches over latent slot sequences:

$$\hat{\mathbf{z}}_{1:\hat{M}} = \arg\max_{\mathbf{z}_{1:\hat{M}}} p_\theta(\mathbf{z}_{1:\hat{M}} \mid q, \mathcal{H}_u, \hat{M}). \quad (36)$$

The second stage decodes Semantic-ID candidates conditioned on the selected slot beams:

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} p_\theta(\mathbf{s} \mid q, \mathcal{H}_u, \hat{\mathbf{z}}_{1:\hat{M}}). \quad (37)$$

The generated Semantic ID is then mapped back to the corresponding item.

B Implementation Details

Optimization. We optimize the student model with AdamW, using $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a weight decay of 10^{-7}. We adopt a constant learning-rate scheduler with a warmup ratio of 0.03 and train all models with bfloat16 precision. The maximum input length is set to 1024. For user history, we keep at most 5 recommendation-history behaviors and 5 search-session behaviors.

Training configuration. For the Amazon experiments, LaPR is trained for 5 epochs with a per-device batch size of 16 and no gradient accumulation. For the Qilin experiments, LaPR is trained for 4 epochs with a per-device batch size of 16 and gradient accumulation steps of 2. Thus, the effective batch size is 16 for Amazon and 32 for Qilin. The main training seed is set to 2024.

Semantic-ID generation. For Amazon, each item is represented by a Semantic ID sequence of length 4, while for Qilin, each item is represented by a Semantic ID sequence of length 3. All generative retrieval baselines use the same RQ-VAE-based Semantic IDs for fair comparison. For the final test evaluation, we increase the beam size to 20. We disable sampling during decoding by setting `do_sample=False`, and no temperature sampling is used.

System:
You are a user-intent reasoning model for a personalized retrieval scenario. You will be given the user's recent behaviors, the current search query, and five candidate items. Select the candidate that best matches the user's current retrieval intent. Then write a clear and self-consistent reasoning process with at most 12 steps.
Each reasoning line must start with "Step k:" where k is the step index. The reasoning must be grounded in the given history, query, and selected candidate metadata. Do not output the final item ID, Semantic ID, or any hidden label.

User:
User history:
- iPad outfit collection
- Coriander beef mixed noodles
- PDD product recommendations
- Useful home-cleaning products
- Search session: English speech training
- Result clicked: independent English teachers

Current query: English speech training in Ningbo
Candidate items:
1. Independent English teachers need lifelong learning
2. Primary-school math tutoring course
3. Ningbo home-cleaning service package
4. English picture books for children
5. Office chair with lumbar support

Please output the selected candidate index and the reasoning steps.

Figure 5: Teacher prompt used for offline candidate-aware trace construction. The teacher selects the target-compatible item from five candidates and generates a step-wise retrieval reasoning trace.

LaPR-specific settings. The maximum reasoning depth is set to $M_{\max} = 12$, and the predicted number of reasoning steps ranges from 1 to 12. The slot codebook size is set to $K = 128$, with 12 segment-level slot positions. The slot prediction loss weight is set to $\lambda_{\mathrm{slot}} = 0.1$, and the step estimator is trained with a standard cross-entropy objective, corresponding to $\lambda_{\mathrm{step}} = 1.0$.

Hardware and reproducibility. All experiments are conducted on NVIDIA RTX PRO 6000 Blackwell Server Edition GPUs. The step-estimator script follows the GenSAR setting and uses random seed 42.

B.1 Baseline Adaptation and Decoding Fairness

All generative methods are adapted to the same input–target format: the current query is followed by the same truncated recommendation and search histories, and the target is the corresponding RQ-VAE Semantic ID. They use Qwen3-1.7B, identical training, validation, and test instances, the same SID tokenizer, output vocabulary, and length limit, deterministic decoding, valid-SID prefix constraints, and an SID beam budget of 20.

Each method retains its original reasoning representation and inference procedure. Direct generative methods use one-stage SID decoding; explicit-reasoning methods retain their rationale-to-SID procedure; and latent-reasoning methods retain their original continuous or discrete latent representations. LaPR generates its slot sequence before constrained SID decoding. Non-generative methods retain their original scoring functions and are evaluated on the same instances and candidate sets. Target-specific metadata, teacher traces, and teacher outputs are not provided at inference.

C Prompt Templates

The following prompt templates are used only for training data construction. They are not required during inference.

C.1 Teacher Selection Prompt

Figure 5 shows the teacher prompt used for offline reasoning-trace construction. For each training instance, we construct a small candidate set containing the target item and four distractor items. The teacher is given the user history, current query, and the five candidate item titles or metadata. It is asked to select the target-compatible candidate and provide step-wise retrieval reasoning for the selection.

The teacher output is not directly trusted. We further apply the verifier in Figure 6 to check whether the selected candidate matches the target item and whether the reasoning trace is valid. Only traces that pass verification are used for semantic slot induction and text-to-slot curriculum distillation. The candidate set and target index are used only in this offline supervision stage and are not provided to the student during inference.

System:
You are a verifier for personalized retrieval reasoning traces. Given the user history, current query, five candidate items, the ground-truth target index, and the teacher output, determine whether the teacher output is valid.
Accept the output only if: (1) the selected candidate index matches the ground-truth target index; (2) the reasoning is consistent with the current query; (3) preference claims are grounded in the user history; (4) the reasoning explains why the selected candidate is more suitable than the alternatives; (5) the output follows the required step-wise format and does not reveal item IDs or Semantic IDs.

User:
User history: {user history}
Current query: {query}
Candidate items: 1. {candidate item 1}
2. {candidate item 2}
3. {candidate item 3}
4. {candidate item 4}
5. {candidate item 5}
Ground-truth target index: {target index}
Teacher output: {selected candidate index and reasoning trace}
Please output `Accept` or `Reject`, followed by a brief justification.

Figure 6: Verifier prompt used for offline filtering of teacher traces. It checks whether the selected candidate matches the target and whether the generated reasoning trace is valid.

C.2 Verifier Prompt

The verifier is used to filter noisy teacher outputs. Given the query, user history, five candidate items, the ground-truth target index, and the teacher output, the verifier checks whether the teacher selected the correct candidate and whether the accompanying reasoning trace is valid. A trace is accepted only if the selected candidate matches the target and the reasoning is query-consistent, history-grounded, target-aligned, and format-valid. This verification step is used only to construct training supervision and is not part of the retrieval model at inference time.

C.3 Student Prompt

Figure 7 shows the prompt used by the student model during inference. Unlike the teacher prompt, the student prompt does not contain the target title or any target-side metadata. The student is required to infer latent reasoning slots from the query-history context and then generate the Semantic ID sequence.

D Example Reasoning Trace

Figure 8 shows an example reasoning trace for a product-search query. The trace decomposes the user's query into intent understanding, requirement extraction, behavior-context analysis, and retrieval-target formulation.

Table 6: Full ablation study on Qilin and Amazon.

Model	Qilin					Amazon				
	HR@1	HR@5	HR@10	NDCG@5	NDCG@10	HR@1	HR@5	HR@10	NDCG@5	NDCG@10
Reference model										
Base	0.1458	0.2910	0.3642	0.2243	0.2476	0.6715	0.7906	0.8089	0.7436	0.7535
Reasoning component ablations										
LaPR w/o infer slots	0.1611	0.3192	0.3827	0.2431	0.2645	0.7099	0.8480	0.8657	0.7867	0.7942
LaPR w/o think	0.1497	0.3112	0.3764	0.2328	0.2545	0.6970	0.8241	0.8374	0.7694	0.7771
LaPR fixed 4 slots	0.1664	0.3248	0.3910	0.2506	0.2715	0.7173	0.8610	0.8751	0.7952	0.7997
Training objective ablations										
LaPR w/o KL reg.	0.1577	0.3184	0.3834	0.2429	0.2643	0.7136	0.8538	0.8712	0.7918	0.7977
LaPR w/o slot loss	0.1539	0.3127	0.3781	0.2383	0.2617	0.7061	0.8437	0.8638	0.7853	0.7927
LaPR w/o curriculum	0.1505	0.3091	0.3735	0.2349	0.2585	0.7016	0.8366	0.8586	0.7808	0.7892
LaPR	**0.1693**	**0.3287**	**0.3951**	**0.2528**	**0.2743**	**0.7198**	**0.8654**	**0.8789**	**0.7978**	**0.8021**

System:
You are a personalized generative retrieval model. Given the user's recent behaviors and the current search query, infer the latent retrieval intent and generate the Semantic ID sequence of the next relevant item. Use compact latent slot tokens inside the thinking markers. Do not output natural-language reasoning.

User:
User history:
- iPad outfit collection
- Coriander beef mixed noodles
- PDD product recommendations
- Useful home-cleaning products
- Search session: English speech training
- Result clicked: independent English teachers

Current query: English speech training in Ningbo
Please generate the latent reasoning slots and the Semantic ID.

Assistant:
`<think> <slot_1> <slot_2> ··· <slot_M> </think> <sid_1> <sid_2> ··· <sid_L>`

Figure 7: Student prompt used by LaPR during inference. The student observes only the user history and current query, predicts latent reasoning slots, and then generates the Semantic ID sequence.

Step 1: Extract the core intent. The user is seeking an ergonomic office chair.
Step 2: Isolate explicit requirements. The requested features are lumbar support and breathable material.
Step 3: Analyze behavioral context. The user recently interacted with work-from-home setup products.
Step 4: Profile priorities. The user values ergonomics and long-duration comfort.
Step 5: Determine purchasing stage. The chair is likely a key remaining item for a home-office setup.
Step 6: Establish must-have filters. Retrieve office or ergonomic chairs with adjustable support and breathable mesh.
Step 7: Predict nice-to-have attributes. Prefer a modern design that matches a technology-oriented workspace.
Step 8: Form retrieval target. Prioritize ergonomic design, adjustable support, breathable upholstery, and positive comfort evidence.

Figure 8: Example workflow for converting a personalized query into structured retrieval constraints.

Table 7: Slot prediction diagnostics.

Metric	Qilin	Amazon
Depth accuracy	74.6%	78.3%
Slot top-1 accuracy	46.8%	51.2%
Slot top-5 accuracy	79.5%	83.1%
Codebook perplexity	63.4	69.8
Avg. predicted depth	4.7	4.3

E Additional Experimental Results

E.1 Full Ablation Results

Table 6 reports the full ablation matrix, while Table 7 summarizes slot prediction diagnostics.

As shown in Table 7, LaPR achieves strong depth prediction accuracy and substantially higher slot top-5 than top-1 accuracy, suggesting that it can estimate adaptive reasoning depth and recover semantically plausible latent slots. The codebook perplexity further indicates that the learned slots use a broad portion of the codebook instead of collapsing to a few frequent codes.

E.2 Matched-Budget Beam-Search Ablation

We compare joint search over the complete slot–SID sequence with two-stage decoding using the same checkpoint, evaluation instances, hardware, and total beam budget. Joint search places slot trajectories and SID candidates in one beam, whereas two-stage decoding constructs the slot sequence before SID ranking.

Under the matched budget of 20, greedy slot generation followed by 20 SID candidates performs best. Dividing the same budget across four slot trajectories and five SID candidates is slightly worse. The improvement therefore comes from separating slot generation from SID ranking rather than from

Table 8: Beam-search ablation under matched total budgets. Qilin and Amazon results are HR@10; latency is measured per query.

Strategy	B_z	B_s	Total	Qilin	Amazon	Latency
Joint search	–	–	10	0.1517	0.7086	1.087 s
Joint search	–	–	20	0.1624	0.7213	1.184 s
Two-stage	1	10	10	0.3887	0.8686	0.542 s
Two-stage	1	20	20	**0.3951**	**0.8789**	**0.630 s**
Two-stage	4	5	20	0.3913	0.8706	0.656 s

Table 9: Full-corpus retrieval performance. All values are hit rates.

Dataset	Method	HR@1	HR@5	HR@10
Qilin	GenSAR	0.0642	0.1583	0.1933
	OneSearch	0.0792	0.2083	0.2217
	OneRec-Think	0.0881	0.2225	0.2369
	STREAM-REC	0.0775	0.2017	0.2200
	CoConut	0.0817	0.2092	0.2250
	Token Assorted	0.0842	0.2134	0.2341
	LaPR	**0.0908**	**0.2308**	**0.2442**
Amazon	GenSAR	0.3475	0.5442	0.5808
	OneSearch	0.6400	0.6942	0.7175
	OneRec-Think	0.7043	0.7425	0.7567
	STREAM-REC	0.6192	0.6753	0.7000
	CoConut	0.6546	0.7033	0.7242
	Token Assorted	0.6552	0.7057	0.7256
	LaPR	**0.7142**	**0.7550**	**0.7708**

using a larger search budget.

E.3 Full-Corpus Retrieval

We additionally evaluate representative generative methods over the complete Qilin and Amazon item collections. All methods use the same RQ-VAE Semantic IDs, valid-SID prefix constraints, and a total beam budget of 20.

LaPR obtains the best HR@1, HR@5, and HR@10 among the compared generative methods on both datasets.

E.4 Target-Blind Trace Generation

We compare the standard target-aware trace generator with a target-blind variant that observes only the query and user history. Both variants use the same training instances, backbone, codebook, optimization settings, and inference procedure. The verifier remains target-aware in both settings.

Removing target access from the trace generator reduces HR@10 by 1.07 points on Qilin and 2.39 points on Amazon. Because the verifier remains target-aware, this experiment does not remove all target-compatible supervision introduced during verification.

Table 10: Effect of target access during offline trace generation.

Trace generation	Qilin HR@10	Amazon HR@10
Target-aware	0.3951	0.8789
Target-blind	0.3844	0.8550
Difference	−0.0107	−0.0239

Table 11: Sensitivity to maximum reasoning depth $M_{\max}$. Latency is measured per query.

	HR@10		
$M_{\max}$	Qilin	Amazon	Latency (s)
6	0.3927	0.8762	0.595
8	0.3946	0.8781	0.613
12	**0.3951**	**0.8789**	0.630
16	0.3918	0.8726	0.656

E.5 Sensitivity to Reasoning Depth and Codebook Size

We vary the maximum reasoning depth $M_{\max}$ and the codebook size $K = |\mathcal{C}|$ while keeping the remaining settings fixed.

Table 12: Sensitivity to codebook size K. Latency is measured per query.

	HR@10		
K	Qilin	Amazon	Latency (s)
32	0.3842	0.8617	0.627
64	0.3948	0.8776	0.630
128	**0.3951**	**0.8789**	0.630
256	0.3891	0.8704	0.628

Performance is stable around $M_{\max} \in \{8, 12\}$ and $K \in \{64, 128\}$. Increasing $M_{\max}$ moderately increases latency, whereas changing K has little effect on latency.

F Artifact Licenses, Intended Use, and Data Protection

We use two public benchmarks in this work: Qilin (Chen et al., 2025b) and Amazon Electronics (Ai et al., 2017). Qilin is released under the MIT License. The Amazon Electronics personalized product search benchmark is built from the original Amazon 5-core review data (McAuley et al., 2015). We follow the access conditions and citation requirements of the original Amazon Product Data, and use it only for academic research and evaluation. We do not redistribute raw reviews, metadata, user records, or item records.

Our use of these artifacts is consistent with their

intended research purposes. For datasets involving user histories, interactions, or reviews, we only use the anonymized or pseudonymized identifiers provided by the original datasets and do not attempt to identify individual users. All experiments are conducted and reported at the aggregate level. Because user-generated reviews or interaction logs may contain noisy or potentially sensitive content, we avoid releasing raw text or individual-level records and use the data solely for model training and evaluation under the original access conditions.

G Use of AI Assistants

We used AI assistants, including ChatGPT, to support language polishing, wording refinement, and presentation improvement. All scientific ideas, experimental design, analyses, results, and final writing decisions were made and verified by the authors.