LASAR: LATENT ADAPTIVE SEMANTIC ALIGNED REASONING FOR GENERATIVE RECOMMENDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) have demonstrated powerful reasoning capabilities through Chain-of-Thought (CoT) in various tasks, yet the inefficiency of token-by-token generation hinders real-world deployment in latency-sensitive recommender systems. Latent reasoning has emerged as an effective paradigm in LLMs, performing multi-step inference in a continuous hidden-state space to achieve stronger reasoning at lower cost. However, this paradigm remains underexplored in mainstream generative recommendation. Achieving this reveals three key challenges: (1) the gap between prior-less Semantic ID (SID) symbols and continuous latent reasoning, as SIDs lack pre-trained semantics, hindering joint optimization; (2) representation drift due to a lack of reasoning chain supervision; and (3) the suboptimality of applying a globally fixed reasoning depth. To address these, we propose **LASAR** (Latent Adaptive Semantic Aligned Reasoning), an SFT-then-RL framework. **First**, we bridge this gap via two-stage training: Stage 1 grounds SID semantics before Stage 2 introduces latent reasoning, ensuring efficient convergence. **Second**, we mitigate representation drift through explicit CoT semantic alignment. Step-wise bidirectional KL divergence constrains the latent reasoning trajectory using hidden-state anchors extracted from CoT text, while a Policy Head predicts per-sample reasoning depth. **Third**, during the GRPO-based RL phase, terminal-only KL alignment accommodates variable-length reasoning, and REINFORCE optimizes the Policy Head to dynamically allocate steps. This nearly halves the average latent step count while improving recommendation quality. Experiments on four real-world datasets show the best overall performance across the evaluated settings. It adds limited inference latency and is roughly 20× faster than generating explicit CoT text. Code: `https://anonymous.4open.science/r/LASAR-E292`

1 INTRODUCTION

Large Language Models (LLMs) in recommender systems are advancing rapidly along two paths. One is **generative recommendation**: P5 Geng et al. (2022) and M6-Rec Cui et al. (2022) pioneered the unified recommendation pretraining paradigm, TIGER Rajput et al. (2023) introduced Semantic ID-based generative retrieval, LC-Rec Zheng et al. (2024) integrated collaborative semantics into LLMs for direct item ID generation, while other studies Bao et al. (2023); Hua et al. (2023); Li et al. (2023); Wang et al. (2024) further advanced generative recommendation from alignment, indexing, and collaboration perspectives. MiniOneRec Kong et al. (2025) built the first fully open-source generative recommendation framework. The other path is **LLM reasoning**: CoT Wei et al. (2022); Kojima et al. (2022) improved task performance by explicitly generating intermediate reasoning steps, while subsequent studies Wang et al. (2023); Yao et al. (2023); Besta et al. (2024); Feng et al. (2023); Merrill & Sabharwal (2024) extended the boundaries of explicit reasoning. However, even as DeepSeek-R1 DeepSeek-AI (2025) and O1 OpenAI (2024) have pushed explicit reasoning to its limits, the fundamental bottleneck of reasoning latency remains unresolved. To address this, Coconut Hao et al. (2024) proposed moving reasoning from the token space to the continuous latent space, realizing multi-step latent reasoning through a hidden state feedback loop, achieving stronger reasoning at lower cost. Subsequent studies Deng et al. (2023; 2024); Saunshi et al. (2025); Geiping et al. (2025); Shen et al. (2025a); Xu et al. (2025b); Chen et al. (2024); Fu et al. (2025); Su et al.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(2025) advanced this paradigm from various angles. Figure 1 contrasts the resulting recommendation paradigms by their reasoning cost.

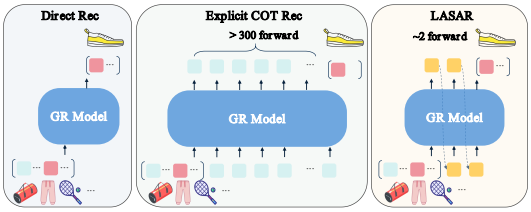


Figure 1: Motivation: long CoT chain to short latent loops.

A natural question arises: **can such latent reasoning bring similar benefits to generative recommendation?** We survey existing work and find that the intersection remains surprisingly limited. In traditional discriminative recommendation, ReaRec Tang et al. (2025), LARES Liu et al. (2025), and other studies Zhang et al. (2025b); Shi et al. (2025) introduced latent reasoning, but they target the ID-embedding discriminative ranking paradigm, not the mainstream decoder-only generative paradigm. In generative recommendation, GREAM Hong et al. (2025) and other studies Tsai et al. (2024); Fang et al. (2025); Lin et al. (2025a); You et al. (2025) leverage explicit CoT reasoning, but text generation incurs high latency and introduces a fundamental tradeoff between reasoning-mode and direct-mode performance, and GREAM's own ablation reveals that its RL post-training (SRPO) degrades Direct recommendation metrics (-5.3% on Instruments), suggesting mode competition. Few preliminary studies Zhang et al. (2025c); Guo et al. (2026) attempt to introduce latent reasoning into LLM-based recommendation, but either perform only shallow single-step attention over history sequences or essentially insert tokens rather than performing recurrent iteration, so neither realizes the full Coconut-style hidden state feedback loop. To the best of our knowledge, **we are the first to perform complete latent reasoning through recurrent hidden-state feedback and adapt reasoning depth to each sample in mainstream generative recommendation.**

When introducing recurrent latent reasoning into a generative recommendation framework, we find that it is **not a "free lunch"**: directly coupling SID grounding with latent reasoning degrades recommendation performance. We identify three key challenges in the generative recommendation setting: **(1) Semantic grounding gap between prior-less SIDs and latent reasoning.** Coconut works well in NLP because its tokens already carry rich pre-trained semantics. In generative recommendation, however, Semantic IDs (SIDs) are an entirely new symbolic system constructed from scratch with zero priors. Jointly training SID learning and latent reasoning forces the model to simultaneously ground a new symbol system *and* perform continuous-space reasoning, causing optimization collapse: without semantic anchors, latent reasoning has no stable foundation to evolve on (Figure 4). **(2) Representation drift.** Recommendation lacks ground-truth reasoning chain supervision: there is no "standard reasoning process" to reference. Without explicit intermediate supervision, latent states can deviate from CoT-derived reference profiles; adding latent reasoning without alignment leaves ranking quality essentially unchanged (Table 2). **(3) Inflexible and inefficient fixed-step reasoning.** Coconut and ReaRec both adopt a globally fixed number of reasoning steps K, applying the same reasoning depth to all samples. A fixed budget is fundamentally suboptimal: many samples can be correctly answered with minimal reasoning, while others benefit from deeper inference. To address these challenges, we propose **LASAR** (Latent Adaptive Semantic Aligned Reasoning), the first framework for adaptive recurrent latent reasoning in mainstream generative recommendation. We further engineer the complete training-and-inference pipeline, with batched variable-depth computation, latent KV-cache reuse, and trie-constrained beam decoding. LASAR adopts an SFT-then-RL paradigm with three designs:

(1) Two-stage decoupling bridges the semantic grounding gap. Within the SFT phase, latent reasoning is deferred to Stage 2, after the model has grounded SID semantics and established basic recommendation capability, resolving the convergence bottleneck and achieving $\sim 3\times$ faster convergence (Figure 4). **(2) Explicit CoT semantic alignment addresses representation drift.** We semantically segment explicit CoT reasoning text, extract hidden states as semantic anchors, and align each latent step to the corresponding CoT segment anchor via bidirectional KL divergence (Section 2.3). The goal is not to replicate explicit reasoning, but to provide semantic guidance for the latent reasoning trajectory. **(3) Policy Head + REINFORCE enables flexible step optimization.** A Policy Head, warm-started during SFT, predicts reasoning depth per sample. Then during the GRPO-based RL phase, REINFORCE optimizes the Policy Head's step allocation, nearly halving the average step count while improving recommendation quality (Section 3.4). At inference, LASAR

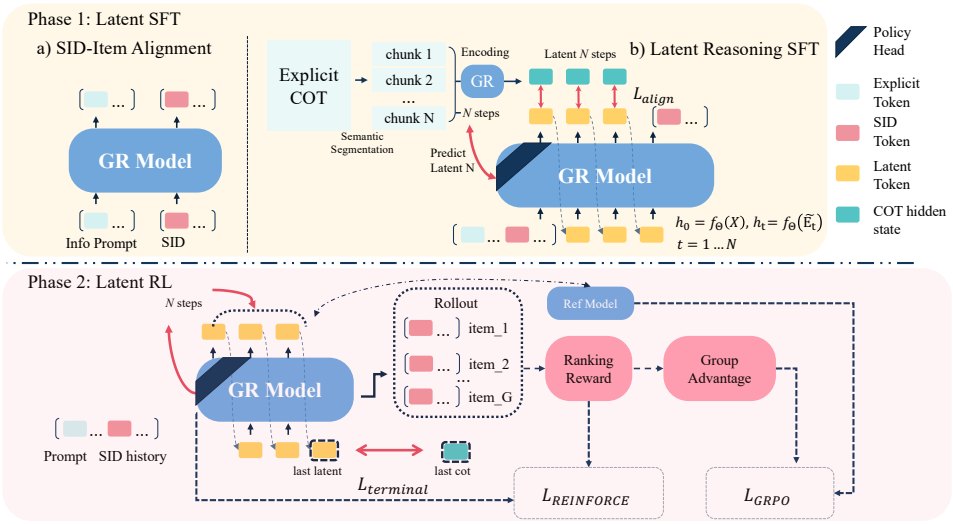


Figure 2: LASAR framework overview. A hidden-state feedback loop iteratively refines latent tokens in continuous space, while a Policy Head predicts per-sample reasoning depth N. SFT: two-stage decoupling + step-wise bidirectional KL alignment with CoT anchors. RL: GRPO (generation quality) + REINFORCE (adaptive reasoning efficiency) + terminal KL (semantic consistency).

incurs limited additional latency over MiniOneRec while remaining over 20× faster than explicit CoT (Table 4).

Experiments on four real-world datasets show LASAR achieves best performance on nearly all metric-dataset combinations, and ablations reveal the failure modes of latent reasoning and isolate each component's contribution.

2 Methodology

The Introduction identified three challenges in adapting latent reasoning to generative recommendation. LASAR addresses them through three corresponding designs: a latent reasoning mechanism with sample-level adaptive step prediction (Section 2.2), an SFT phase that bridges the semantic grounding gap and representation drift via two-stage decoupling and CoT semantic alignment (Section 2.3), and an RL phase that jointly optimizes generation quality and reasoning efficiency (Section 2.4). We first define the generative recommendation task, then describe each component in turn.

2.1 Problem Definition

Let $\mathcal{I}$ be the set of items, where each item $i \in \mathcal{I}$ is associated with text features (e.g., title, description). Given a user's chronological interaction history $\mathcal{S} = \{i_1, i_2, \ldots, i_t\}$, the objective of sequential recommendation is to predict the next item i_{t+1} the user will interact with.

Item Tokenization. To leverage the generative power of LLMs, each item i is represented as a unique sequence of M hierarchical discrete tokens, termed Semantic ID (SID). An item i is mapped to its SID via a quantization function $Q(\cdot)$ applied to its text embedding $\mathbf{e}_i$:

$$\text{SID}(i) = Q(\mathbf{e}_i) = (s_1, s_2, \ldots, s_M), \quad s_j \in \mathcal{C}^{(j)}, \tag{1}$$

where $\mathcal{C}^{(j)}$ denotes the j-th codebook. We follow the Residual Quantization K-Means pipeline, which has been shown to be a strong choice in generative recommendation Hong et al. (2025); Ju et al. (2025). The resulting tokens are integrated into the LLM's vocabulary as special identifiers to capture collaborative signals.

Generative Recommendation. In this framework, the recommendation task is reformulated as conditional sequence generation. We construct the input token sequence X by combining a natural-language prompt (text_{nl}, e.g., task instructions, user or item textual descriptions) with the SID token sequences of the user's history $\mathcal{S}$: $X = \big[\text{text}_{\text{nl}},\ \text{SID}(i_1),\ \text{SID}(i_2),\ \ldots,\ \text{SID}(i_t)\big]$, where

each $\mathrm{SID}(i_j) = (s_1, \ldots, s_M)$ contributes M special tokens. The model generates the target $\mathbf{Y} = \mathrm{SID}(i_{t+1})$ autoregressively:

$$p(\mathbf{Y} \mid X; \Theta) = \prod_{k=1}^{M} p(y_k \mid X, y_1, \ldots, y_{k-1}; \Theta), \tag{2}$$

where y_k is the k-th token of the target SID and Θ denotes the backbone parameters. Our work primarily focuses on the architectural design of the backbone Θ to better capture the complex dependencies within $p(\mathbf{Y} \mid X)$.

2.2 Latent Reasoning Mechanism

LASAR is built on a backbone LLM and performs multi-step reasoning in continuous latent space through a hidden-state feedback loop (Figure 2). The training follows an SFT-then-RL paradigm: the SFT phase (Section 2.3) establishes latent reasoning capability with CoT semantic alignment, and the RL phase (Section 2.4) jointly optimizes generation quality and reasoning efficiency.

Latent Token Design. Three special tokens, `<s>` (start), `<t>` (thought, repeated N times), `<e>` (end), are inserted between the prompt and answer, forming `[Prompt] <s> <t>×N <e> [Answer]`. Unlike Coconut and ReaRec's globally fixed step count K, LASAR predicts a sample-specific N via a Policy Head (Section 2.2).

Recurrent Latent Loop. The core mechanism is a hidden-state feedback loop. Let $h_0 \in \mathbb{R}^D$ denote the last-layer hidden state at the final prompt token. The latent reasoning process,

$$h_0 = f_\Theta(X), \qquad h_t = f_\Theta\big(\tilde{E}_t\big), \quad t = 1, \ldots, N, \tag{3}$$

where $\tilde{E}_t = [E_X,\ h_0,\ h_1,\ \ldots,\ h_{t-1}]$ is the augmented input embedding sequence, where E_X denotes the token embeddings of X. Each subsequent position replaces the standard token embedding with the previous step's hidden state h_{t-1}. After N iterations, the answer segment is generated autoregressively starting from h_N, reusing the accumulated KV cache to avoid recomputing the prompt and latent steps. This design realizes the Coconut-style continuous-space reasoning loop Hao et al. (2024): intermediate states are unobservable dense vectors, and the model iteratively refines its reasoning without generating any discrete tokens.

Adaptive Step Allocation via Policy Head. The Policy Head is a two-layer MLP that predicts the step count N from the prompt-final hidden state h_0: $\pi_\phi(\cdot \mid h_0) = \mathrm{Softmax}\big(W_2 \cdot \tanh(W_1 \cdot h_0 + b_1) + b_2\big)$ with output dimension $N_{\max}$ (maximum reasoning steps, default 8). During SFT, each sample's latent depth N_i equals its number of CoT semantic segments (Section 2.3), and the batch loop runs to $\max_i N_i$; the Policy Head is concurrently CE-trained to predict N_i as a warm start. During RL (Section 2.4), the Policy Head switches to sampling $N \sim \pi_\phi$ and is optimized via REINFORCE (Section 2.4); at inference, $N = \mathrm{argmax}(\pi_\phi)$. A key advantage of pre-predicting N before the latent loop is that all beams of the same prompt share the same N, making the computation graph fully determined during rollout and simplifying batch beam search.

Batch-Efficient Variable Depth Processing. Adaptive N naturally produces per-sample reasoning depths within a batch. We design a padding-and-masking scheme that unifies all samples into a single $\max(N)$-iteration latent loop: short-N samples receive masked pad tokens whose attention weights are zeroed, so the loop runs identically for all samples without branching, retaining full GPU parallelism. Figure 3 illustrates this layout, with further details in Appendix E.1.

	Latent region							
Sample	`<s>`	`<t>`	`<t>`	`<t>`	`<t>`	`<t>`	`<e>`	Answer
A (N=2)	`<s>`	`<t>`	`<t>`	PAD	PAD	PAD	`<e>`	[ans]
B (N=3)	`<s>`	`<t>`	`<t>`	`<t>`	PAD	PAD	`<e>`	[ans]
C (N=5)	`<s>`	`<t>`	`<t>`	`<t>`	`<t>`	`<t>`	`<e>`	[ans]
Attn (A)	1	1	1	0	0	0	1	1
Attn (B)	1	1	1	1	0	0	1	1
Attn (C)	1	1	1	1	1	1	1	1

Figure 3: Batch layout for variable N.

2.3 SFT Phase: Building Semantically Anchored Latent Reasoning (Challenge 1 & 2)

The SFT phase addresses the first two challenges identified in the Introduction: (1) the semantic grounding gap between prior-less SIDs and latent reasoning, and (2) representation drift. With the latent reasoning mechanism defined above, we now describe how it is trained.

Why decoupling is necessary. Unlike NLP tokens that carry pre-trained semantic priors, SID tokens are constructed from scratch with zero prior semantics. Joint training forces the model

to simultaneously ground a new symbolic system *and* perform continuous-space reasoning, and without semantic anchors latent reasoning has no stable trajectory to follow. Figure 4 confirms this: mixed training starts with evaluation loss as high as 3.5–3.9 and converges extremely slowly: after 10 epochs, loss remains above 1.8 (lr=3×10^{-4}) or 2.9 (lr=5×10^{-3}). Counterintuitively, increasing the learning rate from 3×10^{-4} to 5×10^{-3} slows convergence rather than accelerating it, suggesting that the two objectives actively interfere rather than merely lacking optimization capacity. We therefore adopt a **two-stage decoupling** strategy. **Stage 1** serves as a semantic grounding phase: the model learns to generate the recommended item's SID via cross-entropy loss, establishing the SID-to-item semantic mapping so that each symbol acquires stable meaning. After Stage 1 converges, **Stage 2** introduces the latent reasoning mechanism (Section 2.2) with CoT semantic alignment, now able to reason on a stable semantic foundation. In contrast to mixed training, two-stage decoupling starts from 1.79 (Stage 1 has already grounded SID semantics), and Stage 2 converges to ≈1.44 within 4 epochs, reducing total training time from 20+ hours to approximately 6 hours.

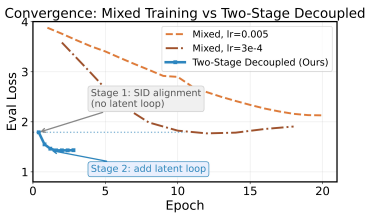


Figure 4: Two-stage decoupling vs. mixed training.

While decoupling resolves the semantic grounding gap (Challenge 1), representation drift (Challenge 2) remains: latent reasoning in continuous space lacks explicit intermediate supervision, allowing hidden states to deviate from CoT-derived reference profiles. We address this through CoT semantic alignment, anchoring the latent reasoning trajectory to explicit CoT segments.

CoT Semantic Alignment. Ablations (Tables 2 and 9) show inconsistent gains from latent reasoning without alignment. To guide the latent trajectory, we anchor each latent reasoning step to explicit CoT semantic segments during SFT, and switch to terminal-only alignment during RL (Section 2.4) to support variable-length reasoning.

Explicit CoT Anchor Construction. The key insight is that explicit CoT reasoning provides a natural trajectory for latent reasoning to follow. Concretely, a large model (e.g., GPT) generates CoT reasoning text per training sample for alignment supervision only. At inference, LASAR requires no CoT text. The CoT text is semantically segmented using an embedding model (e.g., bge-small-en-v1.5) Xiao et al. (2023). The Stage 1 checkpoint then encodes the full trace in a single offline pass, and the final-layer last-token state at each segment boundary forms a pre-computed anchor; anchors are thus context-dependent and held fixed during training. This design shares similarities with CODI Shen et al. (2025b)'s self-distillation, as both guide latent reasoning through hidden states from explicit reasoning, but with a key difference: CODI uses L1 loss for single-token alignment only at the answer generation position, whereas we perform **multi-step alignment** between each latent step and the corresponding explicit CoT segment during SFT, using bidirectional KL divergence between their softmax-normalized activation profiles. The resulting segment count also supervises the Policy Head's depth prediction.

Step-wise Bidirectional KL Alignment. The SFT phase adopts a step-wise alignment strategy: aligning each latent step's hidden state with the corresponding explicit CoT segment's hidden state via bidirectional KL divergence:

$$L_{\text{align}} = \frac{1}{N}\sum_{t=1}^{N} D_{\text{KL}}^{\text{bidir}}(h_t, h_t^{\text{cot}}) \tag{4}$$

where $D_{\text{KL}}^{\text{bidir}}(a,b) = \frac{1}{2}\big(D_{\text{KL}}(\text{Softmax}(a)\|\text{Softmax}(b)) + D_{\text{KL}}(\text{Softmax}(b)\|\text{Softmax}(a))\big)$, h_t is the hidden state from latent step t (Eq. 3), and h_t^{cot} is the fixed Stage 1 anchor at the t-th CoT segment boundary.

This loss matches relative hidden-coordinate activations and supplies direct supervision to each latent step (Appendix F). We select bidirectional KL empirically from the tested alignment losses (Table 2).

SFT Total Loss. The Policy Head is trained with cross-entropy loss using the number of CoT semantic segments as labels for warm start. The total SFT loss combines all objectives: $L_{\text{SFT}} = L_{\text{CE}} + \alpha_{\text{align}} \cdot L_{\text{align}} + \beta_{\text{policy}} \cdot L_{\text{policy}}$, where L_{CE} is the SID generation loss (answer tokens only), L_{policy} is the Policy Head CE loss, and L_{align} is the step-wise bidirectional KL loss.

2.4 RL PHASE: JOINT QUALITY AND EFFICIENCY OPTIMIZATION (CHALLENGE 3)

The SFT phase provides the Policy Head with initial step prediction via CE loss, but fixed CoT segment labels do not directly optimize recommendation quality or reasoning efficiency. The RL phase addresses this through three coordinated objectives: GRPO (generation quality), REINFORCE (step optimization), and Terminal KL (semantic alignment).

GRPO for Recommendation Quality. For each prompt, G candidates are generated, and the reward combines exact match and ranking quality following prior work Kong et al. (2025). We optimize the standard clipped GRPO objective with a KL penalty; Appendix E.5 provides the complete objective, reward, and advantage definitions.

REINFORCE for Adaptive Step Allocation. The CE-trained Policy Head provides a warm-start step distribution, but the label distribution is determined by the segmenter's granularity rather than by what benefits recommendation quality. REINFORCE directly optimizes step allocation through reward-guided exploration, enabling the Policy Head to learn more efficient distributions that improve recommendation quality while reducing average step count. During the RL phase, the Policy Head samples $N \sim \pi_\phi(\cdot \mid h_0)$, and the sampling strategy is optimized via the REINFORCE algorithm:

$$\begin{aligned} L_{\text{REINFORCE}} = & -\mathbb{E}_{N\sim\pi_\phi}[(R_{\text{group}} - b_{\text{EMA}} - \lambda N)\ \log \pi_\phi(N \mid h_0)] \\ & - \eta \cdot H(\pi_\phi) \end{aligned} \tag{5}$$

where R_{group} is the group-level reward for the current prompt (averaged over G candidates), b_{EMA} is the exponential moving average baseline of reward (reducing variance), λN is the step count penalty (encouraging efficiency), and $H(\pi_\phi)$ is an entropy regularization term with coefficient η (preventing degeneration to a less diverse step count). Switching from argmax to sampling enables REINFORCE to learn better step allocation through exploration. The warm-start distribution established by SFT ensures meaningful exploration starting points rather than starting from a random policy.

Terminal Alignment for Variable-Length Reasoning. The SFT phase's step-wise alignment requires each step to correspond to a fixed CoT segment, which is no longer applicable during RL: N is dynamically sampled, producing variable-length reasoning chains that cannot be aligned step-by-step to the fixed-length CoT anchors.

Therefore, the RL phase switches to a **Terminal-only strategy**: aligning only at the **last** latent step of the reasoning chain, $L_{\text{Terminal KL}} = D_{\text{KL}}^{\text{bidir}}(h_N, h_{\text{final}}^{\text{cot}})$, where h_N is the final latent hidden state from Eq. 3 and $h_{\text{final}}^{\text{cot}}$ is the hidden state of the last explicit CoT segment, encouraging the reasoning endpoint to remain semantically aligned with the final CoT anchor regardless of N.

RL Total Loss. We use $L_{\text{total}} = L_{\text{GRPO}} + \gamma_{\text{KL}} L_{\text{Terminal KL}} + \gamma_{\text{RF}} L_{\text{REINFORCE}}$.

3 EXPERIMENTS

We evaluate LASAR on four datasets to answer four questions: **(RQ1)** Does LASAR outperform traditional, generative, latent-reasoning, and explicit-CoT methods (Section 3.2)? **(RQ2)** What are the individual contributions of latent reasoning, alignment, and REINFORCE (Section 3.3)? **(RQ3)** How does adaptive step allocation compare with fixed reasoning depths (Section 3.4)? **(RQ4)** What are LASAR's efficiency and scaling properties (Section 3.5)?

3.1 EXPERIMENTAL SETUP

Datasets. We evaluate on three Amazon product review datasets Ni et al. (2019) (Beauty, Instruments, Sports) and the Steam games dataset Kang & McAuley (2018), spanning two distinct domains. We apply core filtering and leave-one-out evaluation following prior works Kang & McAuley (2018); Sun et al. (2019); Hidasi et al. (2016); Rajput et al. (2023); Zheng et al. (2024); Kong et al. (2025), using 5-core on the Amazon domains and 10-core on the denser Steam data. Sparsity ranges from 99.468% to 99.984%; detailed statistics are in Table 6 (Appendix B).

Baselines. We compare against six baselines spanning four categories: traditional sequential models (SASRec Kang & McAuley (2018), GRU4Rec Hidasi et al. (2016)), LLM-based generative methods (LC-Rec Zheng et al. (2024), MiniOneRec Kong et al. (2025)), latent reasoning (ReaRec Tang et al. (2025)), and explicit CoT reasoning (GREAM Hong et al. (2025)). All generative baselines share the

Table 1: Main results across four datasets. Best in **bold**, second-best underlined.

	Model	N@5	N@10	N@20	HR@5	HR@10	HR@20
Sports	**LASAR**	**0.0121**	**0.0152**	**0.0188**	**0.0185**	**0.0280**	**0.0425**
	Explicit CoT$_{\text{GREAM}}$	0.0089	0.0118	0.0153	0.0138	0.0228	0.0370
	MiniOneRec	0.0099	0.0126	0.0152	0.0155	0.0237	0.0339
	ReaRec	0.0086	0.0112	0.0143	0.0151	0.0233	0.0355
	LC-Rec	0.0081	0.0100	0.0118	0.0123	0.0184	0.0254
	GRU4Rec	0.0062	0.0080	0.0101	0.0090	0.0147	0.0232
	SASRec	0.0060	0.0074	0.0094	0.0089	0.0132	0.0212
Instruments	**LASAR**	**0.0612**	**0.0667**	**0.0730**	**0.0763**	**0.0937**	**0.1184**
	Explicit CoT$_{\text{GREAM}}$	0.0574	0.0621	0.0674	0.0703	0.0850	0.1060
	MiniOneRec	0.0604	0.0640	0.0677	0.0715	0.0826	0.0974
	ReaRec	0.0494	0.0548	0.0604	0.0705	0.0873	0.1095
	LC-Rec	0.0533	0.0561	0.0587	0.0616	0.0701	0.0803
	SASRec	0.0449	0.0475	0.0502	0.0536	0.0617	0.0725
	GRU4Rec	0.0422	0.0454	0.0489	0.0527	0.0629	0.0769
Beauty	**LASAR**	**0.0239**	**0.0303**	**0.0366**	**0.0365**	**0.0563**	0.0813
	Explicit CoT$_{\text{GREAM}}$	0.0228	0.0293	0.0365	0.0351	0.0553	**0.0837**
	MiniOneRec	0.0232	0.0295	0.0358	0.0352	0.0542	0.0795
	ReaRec	0.0201	0.0255	0.0307	0.0296	0.0464	0.0673
	LC-Rec	0.0178	0.0222	0.0261	0.0260	0.0407	0.0592
	SASRec	0.0159	0.0195	0.0229	0.0232	0.0343	0.0480
	GRU4Rec	0.0144	0.0190	0.0242	0.0226	0.0370	0.0573
Steam	**LASAR**	**0.0326**	**0.0429**	**0.0548**	**0.0505**	**0.0862**	**0.1389**
	Explicit CoT$_{\text{GREAM}}$	0.0273	0.0371	0.0464	0.0417	0.0722	0.1090
	MiniOneRec	0.0258	0.0357	0.0481	0.0427	0.0732	0.1224
	ReaRec	0.0277	0.0393	0.0510	0.0485	0.0852	0.1317
	LC-Rec	0.0253	0.0384	0.0509	0.0432	0.0727	0.1126
	SASRec	0.0237	0.0338	0.0460	0.0384	0.0698	0.1185
	GRU4Rec	0.0276	0.0356	0.0494	0.0476	0.0722	0.1266

same base model, prompt, and training data; Explicit CoT$_{\text{GREAM}}$ uses the same CoT data as LASAR, so differences isolate how reasoning is represented rather than what supervision is available. For GREAM, we retain its core CoT chain but strip the 19 augmentation prompts to MiniOneRec's prompt and adopt its own ablation's strongest configuration denoted Explicit CoT$_{\text{GREAM}}$. See Appendix D for details.

Evaluation Metrics. We report NDCG@K and Hit Rate@K for $K \in \{5, 10, 20\}$ (Appendix C). Beam search width is set to 50 for all generative methods; at inference, the model autoregressively generates M SID tokens per candidate, and beam search ensures that the top-K items are ranked by joint token probability.

Implementation Details. CoT reasoning text follows GREAM's structured format, while the input prompt is simplified to MiniOneRec's template. We use Qwen3-0.6B Yang et al. (2025) as the base model, the scale of recent industrial SID-based generative recommenders Collins et al. (2026); Tuo et al. (2026), and further scale to 1.7B (Section 3.5); models are optimized with AdamW (cosine LR schedule with 0.08 warmup). Each item is represented as $M{=}4$ SID tokens from 256-entry codebooks (up to 256^4 sequences). CoT reasoning text is generated following GREAM (see Appendix E.2). All experiments run on $8{\times}$L40 (48 GB); full hyperparameters are in Table 7 (Appendix E).

3.2 Main Results (RQ1)

Table 1 presents the results across all four datasets under NDCG and Hit Rate at $K{=}5, 10, 20$. Among all methods, LASAR achieves the best performance on nearly all metric–dataset combinations. On Sports (most sparse), LASAR achieves particularly strong gains; on Instruments, it shows consistent improvements that widen at larger K ($+7.8\%$ NDCG@20 over MiniOneRec), and on Beauty gains reach $+3.9\%$ HR@10 over MiniOneRec, suggesting latent reasoning is especially beneficial under high sparsity, where the model's semantic understanding compensates for limited collaborative signals. On Steam, a considerably denser non-Amazon domain, LASAR is best on all six metrics ($+9.2\%$ NDCG@10 over ReaRec, the strongest baseline there), so the gains are not confined to sparse review data. Explicit CoT's limited gains may stem from representation interference between language modeling and collaborative filtering objectives, while LASAR reduces such competition by keeping intermediate reasoning in continuous latent space. Improvements are significant on Sports,

Instruments, and Steam ($p < 0.05$) and marginally significant on Beauty ($p < 0.1$, except K=20); see Appendix C.

3.3 Ablation Studies (RQ2)

We next decompose LASAR to understand which components drive its gains (Tables 2, 3).

SFT Phase Ablation. Table 2 compares different alignment strategies during the SFT phase on Beauty (see Table 9 for the other three datasets). Without alignment, latent reasoning improves HR@10 by 3.6% but leaves NDCG@10 essentially flat (+0.4%). These latent steps receive no explicit intermediate supervision from CoT anchors. **CoT alignment improves both retrieval and ranking**, consistent with Coconut Hao et al. (2024), which likewise trains latent reasoning under a staged text-CoT curriculum. Among the three variants, only bidirectional KL improves both NDCG@10 and HR@10 over the no-alignment baseline; MSE degrades sharply. This comparison motivates our empirical choice of bidirectional KL.

Table 2: SFT-phase ablation on Beauty: alignment methods.

Model	Two-Stage	Latent	Alignment	N@5	N@10	HR@5	HR@10	Δ N@10	Δ H@10
Pure SFT (MiniOneRec)			–	0.0212	0.0277	0.0329	0.0531	–	–
+ Latent (w/o align.)	✓	✓	None	0.0207	0.0278	0.0327	0.0550	+0.4%	+3.6%
+ KL Alignment	✓	✓	**KL**	**0.0217**	**0.0285**	**0.0340**	**0.0552**	**+2.9%**	**+4.0%**
+ Cosine Alignment	✓	✓	Cosine	0.0211	0.0277	0.0341	0.0543	0.0%	+2.3%
+ MSE Alignment	✓	✓	MSE	0.0187	0.0245	0.0295	0.0477	−11.6%	−10.2%

RL-Phase Ablation. The SFT ablation confirms that KL alignment enables latent reasoning to work, and we now assess the distinct contribution of each of the three RL components (GRPO, Terminal KL, REINFORCE). Table 3 reports the RL-phase ablation on Beauty (176K interactions for stable gradients), with validation on Sports in Section 3.4.

Naively adding latent reasoning without proper RL training degrades performance, confirming **latent reasoning is not a free improvement**. Terminal KL alignment improves NDCG@10 by 2.4% over unaligned latent RL, recovering most of the performance loss, while Mean N rises from 3.59 to 4.20. Terminal alignment guides the reasoning endpoint; REINFORCE supplies the explicit depth penalty. Adding REINFORCE further improves NDCG@10 while compressing Mean N from ≈4.2 to ≈2.5, demonstrating **step compression and quality improvement are concurrent**. Terminal KL and REINFORCE provide complementary gains.

3.4 Step Optimization Analysis (RQ3)

The SFT step labels track sample complexity on Sports: 83.4% of samples have 3 teacher CoT segments (mean history length 6.9; category diversity 6.2), 16.3% have 4 (8.5; 7.5), and 0.2% have 5 (9.8; 8.8). This provides difficulty-aware supervision for the Policy Head before RL (Table 8 in Appendix G). We next examine how REINFORCE allocates those steps.

Force N Experiment: Adaptive Allocation Outperforms Fixed Depths. On Sports, adaptive sampling achieves HR@10 of 2.80%, exceeding all fixed-N settings (Figure 5a). Fixed N=4 performs worst (1.93%), suggesting that uniform extra iterations can disrupt predictions.



(a) Force N comparison.

(b) Step distribution: SFT vs. RL.

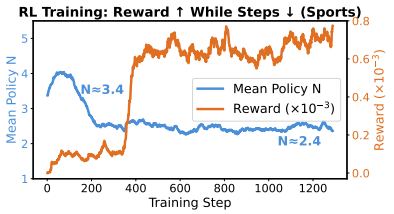
Figure 5: Adaptive versus fixed step allocation and the SFT-to-RL step distribution on Sports.

RL Dynamics and Per-N Analysis. During RL on Sports, Mean N falls from ≈3.4 and plateaus near ≈2.4 as Reward rises (Figure 6a), so RL compresses the average step budget while improving recommendation quality (Table 3). The step distribution broadens from SFT's N=3, 4 concentration (99.7%) to N=1–8 after RL (Mean = 2.47; Figure 5b).

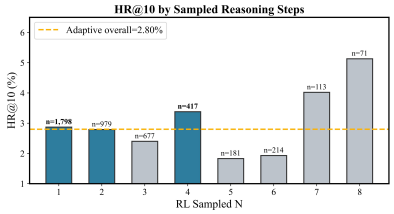
Table 3: RL-phase ablation on Beauty. Δ: relative to previous row.

Model	Latent	Terminal KL	REINFORCE	N@5	N@10	HR@5	HR@10	Mean N	Δ N@10
MiniOneRec	N/A	N/A	N/A	0.0232	0.0295	0.0352	0.0542	N/A	–
RL w/ latent reasoning	✓			0.0227	0.0287	0.0346	0.0533	3.59	−2.7%
+ Terminal KL Alignment	✓	✓		0.0233	0.0294	0.0353	0.0543	4.20	+2.4%
+ REINFORCE (LASAR)	✓	✓	✓	**0.0239**	**0.0303**	**0.0365**	**0.0563**	**2.47**	**+3.1%**

REINFORCE does not simply minimize depth. Per-N analysis (Figure 6b) shows that the Policy Head assigns shallow reasoning ($N \leq 4$) to most samples but retains deeper reasoning ($N \geq 7$) for a smaller group: the samples routed to $N=4$ reach 3.38% HR@10, whereas forcing $N=4$ on all samples yields 1.93%. A within-group analysis confirms the pattern on matched samples: restricting each group to each depth level in turn, every group attains its best HR@10 under its own assigned level on Sports, Instruments, and Beauty (Figure 9, Appendix G).



(a) RL training dynamics.



(b) Per-N HR@10.

Figure 6: RL dynamics and per-N HR@10 on Sports. Mean N falls as Reward rises; per-N HR@10 is non-monotonic, peaking at $N=4$ among mid-range depths.

3.5 Inference Efficiency and Model Scaling (RQ4)

We now evaluate inference efficiency and model scaling.

Table 4: Inference efficiency.

Dataset	Method	GPU-sec/Sample	Wall-clock (8×L40)
Beauty	MiniOneRec	0.27	12min
	LASAR	0.29	13min
	Explicit CoT$_{\text{GREAM}}$ (CoT Gen.)	7.0	5.5h
Instruments	MiniOneRec	0.25	13min
	LASAR	0.29	15min
	Explicit CoT$_{\text{GREAM}}$ (CoT Gen.)	6.5	5h
Sports	MiniOneRec	0.30	22min
	LASAR	0.32	24min
	Explicit CoT$_{\text{GREAM}}$ (CoT Gen.)	7.0	8.5h

Inference Efficiency. Table 4 reports inference latency across the three Amazon datasets under beam width 50 on 8×L40. LASAR adds only ≈7–16% inference overhead over MiniOneRec. Explicit CoT$_{\text{GREAM}}$ has two inference modes: the main results (Table 1) use its *direct-answer* mode (no CoT text; the strongest configuration in GREAM's own ablation, Appendix D), while Table 4 reports its *CoT-generation* mode, in which explicit reasoning actually occurs at inference. LASAR is more accurate overall than the direct-answer mode (which does not reason at inference) and over 20× faster than the CoT-generation mode.

Table 5: Model scaling on Beauty.

	0.6B Full FT		1.7B LoRA	
Method	N@10	HR@10	N@10	HR@10
LASAR	**0.0303**	**0.0563**	**0.0307**	**0.0592**
MiniOneRec	0.0295	0.0542	0.0299	0.0556
Explicit CoT$_{\text{GREAM}}$	0.0293	0.0553	0.0295	0.0561

Model Scaling. On Beauty, LASAR leads at both Qwen3-0.6B (full fine-tuning) and Qwen3-1.7B (LoRA), each trained with SFT+RL (Table 5). Its N@10 scales comparably to MiniOneRec, so latent reasoning does not cap capacity gains, while its HR@10 gain from 0.6B to 1.7B exceeds the gains of both baselines.

4 Conclusion

This paper proposes LASAR, the first framework to perform complete latent reasoning with recurrent hidden-state feedback and per-sample adaptive depth in mainstream generative recommendation. Through two-stage SFT decoupling, explicit CoT semantic alignment with bidirectional KL divergence, and REINFORCE-based adaptive step optimization, LASAR simultaneously improves recommendation quality and reduces inference cost. Experiments on four datasets show the best overall performance across the evaluated settings, with ablations attributing the gains to semantic alignment, Terminal KL, and REINFORCE.

Limitations and future work. The hidden-state feedback loop precludes teacher forcing, requiring sequential forward passes, and LASAR depends on offline teacher CoT synthesis, which never enters inference. Future work may target efficient latent-loop execution and conversational recommendation.

AI USE STATEMENT

Generative AI tools assisted with language editing and drafting of this manuscript; the authors reviewed all outputs and retain full responsibility.

REPRODUCIBILITY STATEMENT

Dataset statistics (Appendix B), evaluation metrics (Appendix C), baseline configurations (Appendix D), and engineering implementation details (Appendix E)—including the LoRA configuration, SID item embedding construction, RL hyperparameters, prompt and CoT formats, reward formulation, and the training and inference algorithms—are provided in the appendix. Code is available at `https://anonymous.4open.science/r/LASAR-E292`

REFERENCES

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, pp. 1007–1014, 2023. doi: 10.1145/3604915.3608857. URL https://doi.org/10.1145/3604915.3608857.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024*, pp. 17682–17690, 2024. doi: 10.1609/AAAI.V38I16.29720. URL https://doi.org/10.1609/aaai.v38i16.29720.

Haolin Chen, Yihao Feng, Zuxin Liu, Weiran Yao, Akshara Prabhakar, Shelby Heinecke, Ricky Ho, Phil Mui, Silvio Savarese, Caiming Xiong, and Huan Wang. Language models are hidden reasoners: Unlocking latent reasoning capabilities via self-rewarding. *CoRR*, abs/2411.04282, 2024. doi: 10.48550/ARXIV.2411.04282. URL https://doi.org/10.48550/arXiv.2411.04282.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *CoRR*, abs/2412.13171, 2024. doi: 10.48550/ARXIV.2412.13171. URL https://doi.org/10.48550/arXiv.2412.13171.

Liam Collins, Jiwen Ren, Donald Loveland, Bhuvesh Kumar, Clark Mingxuan Ju, Xuan Guo, Mo Li, Alvin Hou, Yi Cui, Peng Yang, Jian Wang, Saud Afzal Shafi, Nga Than, Ruiming Lu, Wenfeng Zhuo, Dongheng Li, Lili Zhang, Mingtao Zhang, Jinchao Ye, Vincent Xue, Chunhui Zhu, and Neil Shah. Llm-based generative retrieval for snapchat content recommendation. *CoRR*, abs/2607.28895, 2026. doi: 10.48550/ARXIV.2607.28895. URL https://doi.org/10.48550/arXiv.2607.28895.

Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. M6-rec: Generative pretrained language models are open-ended recommender systems. *CoRR*, abs/2205.08084, 2022. doi: 10.48550/ARXIV.2205.08084. URL https://doi.org/10.48550/arXiv.2205.08084.

Sunhao Dai, Jiakai Tang, Jiahua Wu, Kun Wang, Yuxuan Zhu, Bingjun Chen, Bangyang Hong, Yu Zhao, Cong Fu, Kangle Wu, Yabo Ni, Anxiang Zeng, Wenjie Wang, Xu Chen, Jun Xu, and See-Kiong Ng. Onepiece: Bringing context engineering and reasoning to industrial cascade ranking system, 2025. URL https://arxiv.org/abs/2509.18091.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart M. Shieber. Implicit chain of thought reasoning via knowledge distillation. *CoRR*, abs/2311.01460,

2023. doi: 10.48550/ARXIV.2311.01460. URL https://doi.org/10.48550/arXiv.2311.01460.

Yuntian Deng, Yejin Choi, and Stuart M. Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *CoRR*, abs/2405.14838, 2024. doi: 10.48550/ARXIV.2405.14838. URL https://doi.org/10.48550/arXiv.2405.14838.

Yi Fang, Wenjie Wang, Yang Zhang, Fengbin Zhu, Qifan Wang, Fuli Feng, and Xiangnan He. Reason4rec: Large language models for recommendation with deliberative user preference alignment. *CoRR*, abs/2502.02061, 2025. doi: 10.48550/ARXIV.2502.02061. URL https://doi.org/10.48550/arXiv.2502.02061.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/dfc310e81992d2e4cedc09ac47eff13e-Abstract-Conference.html.

Tianyu Fu, Yichen You, Zekai Chen, Guohao Dai, Huazhong Yang, and Yu Wang. Think-at-hard: Selective latent iterations to improve reasoning language models. *CoRR*, abs/2511.08577, 2025. doi: 10.48550/ARXIV.2511.08577. URL https://doi.org/10.48550/arXiv.2511.08577.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *CoRR*, abs/2502.05171, 2025. doi: 10.48550/ARXIV.2502.05171. URL https://doi.org/10.48550/arXiv.2502.05171.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (RLP): A unified pretrain, personalized prompt & predict paradigm (P5). In *RecSys '22: Sixteenth ACM Conference on Recommender Systems*, pp. 299–315, 2022. doi: 10.1145/3523227.3546767. URL https://doi.org/10.1145/3523227.3546767.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations, ICLR 2024*, 2024. URL https://openreview.net/forum?id=ph04CRkPdC.

Zihao Guo, Jian Wang, Ruxin Zhou, Youhua Liu, Jiawei Guo, Jun Zhao, Xiaoxiao Xu, Yongqi Liu, and Kaiqiao Zhan. S^2gr: Stepwise semantic-guided reasoning in latent space for generative recommendation. *CoRR*, abs/2601.18664, 2026. doi: 10.48550/ARXIV.2601.18664. URL https://doi.org/10.48550/arXiv.2601.18664.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *CoRR*, abs/2412.06769, 2024. doi: 10.48550/ARXIV.2412.06769. URL https://doi.org/10.48550/arXiv.2412.06769.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.06939.

Minjie Hong, Zetong Zhou, Zirun Guo, Ziang Zhang, Ruofan Hu, Weinan Gan, Jieming Zhu, and Zhou Zhao. Generative reasoning recommendation via llms. *CoRR*, abs/2510.20815, 2025. doi: 10.48550/ARXIV.2510.20815. URL https://doi.org/10.48550/arXiv.2510.20815.

Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. How to index item ids for recommendation foundation models. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region, SIGIR-AP 2023*, pp. 195–204, 2023. doi: 10.1145/3624918.3625339. URL https://doi.org/10.1145/3624918.3625339.

Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvesh Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. Generative recommendation with semantic ids: A practitioner's handbook. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025*, pp. 6420–6425, 2025. doi: 10.1145/3746252.3761612. URL https://doi.org/10.1145/3746252.3761612.

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

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

Xiaoyu Kong, Leheng Sheng, Junfei Tan, Yuxin Chen, Jiancan Wu, An Zhang, Xiang Wang, and Xiangnan He. Minionerec: An open-source framework for scaling generative recommendation. *CoRR*, abs/2510.24431, 2025. doi: 10.48550/ARXIV.2510.24431. URL https://doi.org/10.48550/arXiv.2510.24431.

Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. E4srec: An elegant effective efficient extensible solution of large language models for sequential recommendation. *CoRR*, abs/2312.02443, 2023. doi: 10.48550/ARXIV.2312.02443. URL https://doi.org/10.48550/arXiv.2312.02443.

Jiacheng Lin, Tian Wang, and Kun Qian. Rec-r1: Bridging generative large language models and user-centric recommendation systems via reinforcement learning. *Trans. Mach. Learn. Res.*, 2025, 2025a. URL https://openreview.net/forum?id=YBRU9MV2vE.

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. How can recommender systems benefit from large language models: A survey. *ACM Trans. Inf. Syst.*, 43(2): 28:1–28:47, 2025b. doi: 10.1145/3678004. URL https://doi.org/10.1145/3678004.

Enze Liu, Bowen Zheng, Xiaolei Wang, Wayne Xin Zhao, Jinpeng Wang, Sheng Chen, and Ji-Rong Wen. LARES: latent reasoning for sequential recommendation. *CoRR*, abs/2505.16865, 2025. doi: 10.48550/ARXIV.2505.16865. URL https://doi.org/10.48550/arXiv.2505.16865.

William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. In *The Twelfth International Conference on Learning Representations, ICLR 2024*, 2024. URL https://openreview.net/forum?id=NjNGlPh8Wh.

Jianmo Ni, Jiacheng Li, and Julian J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pp. 188–197, 2019. doi: 10.18653/V1/D19-1018. URL https://doi.org/10.18653/v1/D19-1018.

OpenAI. Openai o1 system card. *CoRR*, abs/2412.16720, 2024. doi: 10.48550/ARXIV.2412.16720. URL https://doi.org/10.48550/arXiv.2412.16720.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models. *CoRR*, abs/2404.15758, 2024. doi: 10.48550/ARXIV.2404.15758. URL https://doi.org/10.48550/arXiv.2404.15758.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Mahesh Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/20dcab0f14046a5c6b02b61da9f13229-Abstract-Conference.html.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*, 2025. URL https://openreview.net/forum?id=din0lGfZFd.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Xuan Shen, Yizhou Wang, Xiangxi Shi, Yanzhi Wang, Pu Zhao, and Jiuxiang Gu. Efficient reasoning with hidden thinking. *CoRR*, abs/2501.19201, 2025a. doi: 10.48550/ARXIV.2501.19201. URL https://doi.org/10.48550/arXiv.2501.19201.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025*, pp. 677–693, 2025b. doi: 10.18653/V1/2025.EMNLP-MAIN.36. URL https://doi.org/10.18653/v1/2025.emnlp-main.36.

Teng Shi, Weicong Qin, Weijie Yu, Xiao Zhang, Ming He, Jianping Fan, and Jun Xu. Bridging search and recommendation through latent cross reasoning. *CoRR*, abs/2508.04152, 2025. doi: 10.48550/ARXIV.2508.04152. URL https://doi.org/10.48550/arXiv.2508.04152.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. In *Forty-second International Conference on Machine Learning, ICML 2025*, 2025. URL https://proceedings.mlr.press/v267/su25g.html.

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

Jiakai Tang, Sunhao Dai, Teng Shi, Jun Xu, Xu Chen, Wen Chen, Wu Jian, and Yuning Jiang. Think before recommend: Unleashing the latent reasoning power for sequential recommendation. *CoRR*, abs/2503.22675, 2025. doi: 10.48550/ARXIV.2503.22675. URL https://doi.org/10.48550/arXiv.2503.22675.

Alicia Tsai, Adam Kraft, Long Jin, Chenwei Cai, Anahita Hosseini, Taibai Xu, Zemin Zhang, Lichan Hong, Ed Huai-hsin Chi, and Xinyang Yi. Leveraging LLM reasoning enhances personalized recommender systems. In *Findings of the Association for Computational Linguistics, ACL 2024*, pp. 13176–13188, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.780. URL https://doi.org/10.18653/v1/2024.findings-acl.780.

Jiayi Tuo, Hehan Li, Dongjun Fu, Xin Lu, Ling Zhuang, Fuwei Zhang, Meifang Li, Peizhi Xu, Hanmeng Liu, Shuanglong Li, Liwei Qian, Yanbiao Ma, and Fuzhen Zhuang. Icegr: An intent-coherent end-to-end generative retrieval framework for e-commerce search. *CoRR*, abs/2608.29652, 2026. doi: 10.48550/ARXIV.2608.29652. URL https://doi.org/10.48550/arXiv.2608.29652.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, and Zhenhua Dong. EAGER: two-stream generative recommender with behavior-semantic collaboration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024*, pp. 3245–3254, 2024. doi: 10.1145/3637528.3671775. URL https://doi.org/10.1145/3637528.3671775.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. C-pack: Packaged resources to advance general chinese embedding, 2023.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Chenyang Shao, Yuwei Yan, Qinglong Yang, Yiwen Song, Sijian Ren, Xinyuan Hu, Yu Li, Jie Feng, Chen Gao, and Yong Li. Towards large reasoning models: A survey of reinforced reasoning with large language models. *CoRR*, abs/2501.09686, 2025a. doi: 10.48550/ARXIV.2501.09686. URL https://doi.org/10.48550/arXiv.2501.09686.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025*, pp. 23336–23351, 2025b. URL https://aclanthology.org/2025.acl-long.1137/.

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, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

Runyang You, Yongqi Li, Xinyu Lin, Xin Zhang, Wenjie Wang, Wenjie Li, and Liqiang Nie. R^2ec: Towards large recommender models with reasoning. *CoRR*, abs/2505.16994, 2025. doi: 10.48550/ARXIV.2505.16994. URL https://doi.org/10.48550/arXiv.2505.16994.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *CoRR*, abs/2407.06023, 2024. doi: 10.48550/ARXIV.2407.06023. URL https://doi.org/10.48550/arXiv.2407.06023.

Jiaqi Zhang, Junliang Yu, Zongwei Wang, Wei Yuan, Tong Chen, Quoc Viet Hung Nguyen, Bin Cui, and Hongzhi Yin. Towards reasoning-aware recommender systems: A survey in the llm era. *TechRxiv*, 2025(1117), 2025a. doi: 10.36227/techrxiv.176287939.92578520/v2. URL https://www.techrxiv.org/doi/abs/10.36227/techrxiv.176287939.92578520/v2.

Junjie Zhang, Beichen Zhang, Wenqi Sun, Hongyu Lu, Wayne Xin Zhao, Yu Chen, and Ji-Rong Wen. Slow thinking for sequential recommendation. *CoRR*, abs/2504.09627, 2025b. doi: 10.48550/ARXIV.2504.09627. URL https://doi.org/10.48550/arXiv.2504.09627.

Yang Zhang, Wenxin Xu, Xiaoyan Zhao, Wenjie Wang, Fuli Feng, Xiangnan He, and Tat-Seng Chua. Reinforced latent reasoning for llm-based recommendation. *CoRR*, abs/2505.19092, 2025c. doi: 10.48550/ARXIV.2505.19092. URL https://doi.org/10.48550/arXiv.2505.19092.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *40th IEEE International Conference on Data Engineering, ICDE 2024*, pp. 1435–1448, 2024. doi: 10.1109/ICDE60146.2024.00118. URL https://doi.org/10.1109/ICDE60146.2024.00118.

A Related Work

A.1 From Explicit to Latent LLM Reasoning

LLM reasoning capability has become a central research direction in recent years. Wei et al. Wei et al. (2022) first proposed Chain-of-Thought (CoT) prompting, significantly improving complex task performance by explicitly generating intermediate reasoning steps. Subsequently, Kojima et al. Kojima et al. (2022) discovered that simply prompting "Let's think step by step" elicits zero-shot reasoning, while subsequent studies Wang et al. (2023); Yao et al. (2023); Besta et al. (2024); Feng et al. (2023); Merrill & Sabharwal (2024) extended the boundaries of explicit CoT from angles including multi-sample majority voting, tree/graph-structured search, and theoretical expressiveness. However, the fundamental cost of explicit CoT lies in token-by-token reasoning text generation, with latency scaling linearly with chain length. Even as DeepSeek-R1 DeepSeek-AI (2025) and OpenAI O1 OpenAI (2024) have pushed explicit reasoning to its limits through RL, and Snell et al. Snell et al. (2025) provided optimal test-time compute allocation strategies, the generation cost of explicit reasoning remains a fundamental bottleneck.

To break this bottleneck, researchers have begun exploring the transfer of reasoning from the discrete token space to the continuous latent space. The core paradigm is: feed the hidden state from one LLM layer directly as the input embedding for the next step, iteratively refining reasoning through multi-step recurrent loops in continuous space, with intermediate states being unobservable dense vectors rather than readable discrete tokens. Under this definition, simply inserting additional learnable tokens or adding single-shot cross-attention does not constitute this form of latent reasoning.

Hao et al. Hao et al. (2024)'s Coconut is the pioneering work of this paradigm: feeding the last-layer hidden state as the next-step input embedding, realizing multi-step latent recurrent reasoning without generating any explicit tokens. Coconut adopts progressive curriculum training and discovers that latent space naturally supports BFS-like parallel path exploration. Its precursor works include Deng et al. Deng et al. (2023; 2024), which gradually internalize explicit reasoning into latent reasoning via knowledge distillation. Contemporaneously, Goyal et al. Goyal et al. (2024) and Pfau et al. Pfau et al. (2024) showed that inserting learnable virtual tokens can allocate additional computation to the model, but such discrete-token reasoning is significantly weaker than Coconut's continuous recurrent representations Hao et al. (2024). Cheng and Van Durme Cheng & Durme (2024) and Yu et al. Yu et al. (2024) also explored different paths of compressing explicit reasoning into latent space.

Along the Coconut line, follow-up work includes: Looped Transformers Saunshi et al. (2025) proving that the same set of parameters recurrently executed multiple times is equivalent to increasing model depth without adding parameters; Huginn Geiping et al. (2025) adopting a prelude-core-coda architecture to realize deep recurrent latent reasoning, achieving performance comparable to larger models through test-time compute scaling; CODI Shen et al. (2025b) compressing explicit CoT into continuous space via self-distillation, matching CoT-SFT performance in a single training step. In diverse explorations, Heima Shen et al. (2025a) compresses each CoT step into a single thinking token; SoftCoT Xu et al. (2025b) generates soft thinking tokens through a frozen LLM plus trainable projection module to avoid catastrophic forgetting; LaTRO Chen et al. (2024) optimizes latent reasoning through variational inference and self-rewarding mechanisms. In adaptive reasoning, Think-at-Hard Fu et al. (2025) selectively triggers latent iterations based on token difficulty, and Token-Assorted Su et al. (2025) explores mixed reasoning with latent and text tokens.

In summary, latent reasoning has achieved significant progress in math and logic reasoning domains, but none of these methods have entered the mainstream decoder-only generative recommendation domain. This paper aims to fill exactly this gap.

A.2 LLM-based Generative Recommendation

Recommender systems are undergoing a shift from the traditional ID-embedding discriminative paradigm toward an LLM-driven generative paradigm. Traditional sequential recommendation methods such as SASRec Kang & McAuley (2018), BERT4Rec Sun et al. (2019), and GRU4Rec Hidasi et al. (2016) use self-attention or bidirectional encoders to extract user preferences and rank items via dot products.

With the rise of LLMs, a series of works have reformulated recommendation as a sequence generation problem. Geng et al.'s P5 Geng et al. (2022) and M6-Rec Cui et al. (2022) pioneered a unified recommendation pretraining paradigm, unifying diverse recommendation signals into a text generation format. TALLRec Bao et al. (2023) further explored parameter-efficient alignment of LLMs with recommendation. In the generative retrieval direction, Rajput et al.'s TIGER Rajput et al. (2023) modeled recommendation as token-by-token generation of Semantic IDs, and subsequent work Hua et al. (2023) systematically studied semantic indexing methods for item IDs. Building on these foundations, LC-Rec Zheng et al. (2024) integrated collaborative semantic information into LLMs, training them via SFT to directly generate recommended item IDs. Other studies Li et al. (2023); Wang et al. (2024) advanced generative recommendation efficiency and effectiveness from perspectives of LLM-based sequential recommendation, single-LLM semantic tokenization, and behavior-semantic dual-stream collaboration, respectively. Recently, MiniOneRec Kong et al. (2025) built the first fully open-source generative recommendation framework upon a streamlined OneRec, providing a complete LLM generative recommendation SFT+RL post-training pipeline and a solid foundation for our work.

In the reasoning-enhanced direction, GREAM Hong et al. (2025) proposed applying CoT explicit reasoning to generative recommendation, but the high generation latency of explicit CoT constrains its practical deployment efficiency. Other studies Tsai et al. (2024); Fang et al. (2025); Lin et al. (2025a); You et al. (2025); Dai et al. (2025) also explored combining LLM reasoning with recommender systems from different angles, but all rely on explicit textual reasoning chains. Overall, existing generative recommendation methods lack exploration of multi-step recurrent reasoning mechanisms in continuous latent space.

A.3 Latent Reasoning in Recommendation

Bringing the above latent reasoning paradigm into recommender systems is a frontier direction.

Latent recurrent reasoning for traditional sequential recommendation. ReaRec Tang et al. (2025) realizes latent reasoning within a SASRec-style architecture, trained with temperature annealing and reasoning-aware contrastive learning, but uses globally fixed step counts. LARES Liu et al. (2025) proposes a depth-recurrent latent reasoning framework, repeatedly refreshing all input token hidden states through the same set of Transformer layers at each step to increase computation density, with a two-stage strategy of self-supervised pretraining plus RL post-training. STREAMRec Zhang et al. (2025b) introduces stepwise reasoning in sequential recommendation from the "slow thinking" perspective. LCR-SER Shi et al. (2025) performs latent cross-modal reasoning through dual-tower cross-attention iteration in a joint search-and-recommendation scenario, but this is essentially an information fusion mechanism rather than a hidden state feedback loop. The common limitation of the above methods is that they are based on the ID-embedding + dot-product discriminative ranking paradigm, a different technical route from LASAR's LLM decoder-only generative recommendation.

Latent reasoning explorations in generative recommendation. LatentR^3 Zhang et al. (2025c) is among the few works introducing latent reasoning into LLM-based recommendation; its LatentRATT module uses modified GRPO for two-stage training, but reasoning only extracts information from the LLM's final hidden state through single-layer attention. S^{2}GR Guo et al. (2026), a concurrent work, inserts thinking tokens in generative retrieval and aligns them with SIDs, but this is essentially sequence insertion rather than recurrent iteration. Neither realizes the Coconut-style multi-step hidden state feedback loop in generative recommendation.

Key distinctions from LASAR. To our knowledge, LASAR is the first framework to perform complete latent reasoning through recurrent hidden-state feedback and adapt its reasoning depth to each sample in mainstream generative recommendation, with the following core features: (1) multi-step hidden state feedback loop with beam search; (2) explicit CoT semantic alignment (bidirectional KL) to prevent representation drift; (3) Policy Head + REINFORCE for sample-level adaptive step optimization. Systematic literature surveys Lin et al. (2025b); Xu et al. (2025a); Zhang et al. (2025a) indicate that latent reasoning research in recommender systems is still in its early stages. LASAR combines recurrent latent reasoning, generative recommendation, and adaptive depth.

B Dataset Statistics

Table 6 summarizes the statistics of the four datasets used in our experiments: three Amazon review domains and the Steam games dataset. We use Steam interactions from 2016-10-01 to 2018-01 and apply 10-core filtering on both users and items.

Table 6: Statistics of the four datasets used in experiments.

Dataset	#Users	#Items	#Inter.	Sparsity	#Test
Beauty	22,363	12,101	176,139	99.935%	22,363
Instruments	24,772	9,922	74,316	99.970%	24,772
Sports	35,598	18,357	106,794	99.984%	35,598
Steam	11,472	3,478	212,303	99.468%	11,472

C Evaluation Metrics

We adopt leave-one-out evaluation: the last interaction of each user is held out as the test item. For each user, the model generates a ranked list of candidates via beam search, and the following metrics are computed against the ground-truth next item.

Hit Rate@K (HR@K). Measures whether the ground-truth item appears in the top-K ranked list:

$$\text{HR@}K = \begin{cases} 1 & \text{if ground-truth item} \in \text{top-}K \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Normalized Discounted Cumulative Gain@K (NDCG@K). Assigns higher scores when the ground-truth item is ranked closer to the top. With a single ground-truth item at rank r (where $r{=}1$ is the highest):

$$\text{NDCG@}K = \begin{cases} \frac{1}{\log_2(r+1)} & \text{if } r \le K \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Since there is exactly one relevant item, the ideal DCG (IDCG) is 1.

We report both metrics at $K \in \{5, 10, 20\}$. Results are averaged over all test users. Significance is assessed with a one-sided paired bootstrap: for each dataset–metric, we resample test users with replacement (2000 replicates) and recompute the per-user metric for LASAR versus the strongest non-LASAR baseline on that dataset–metric, reporting the one-sided p-value of LASAR exceeding the baseline.

D Baseline Details

Traditional sequential models. SASRec Kang & McAuley (2018) applies the self-attention mechanism to sequential recommendation, adaptively weighting historical items to capture long-term user preferences. GRU4Rec Hidasi et al. (2016) is a pioneering session-based recommendation model that uses GRU to model user click sequences, representing the classical RNN-based approach.

LLM-based generative methods. LC-Rec Zheng et al. (2024) bridges collaborative filtering signals with LLM semantics through vector-quantized discrete item indices and multi-task alignment, enabling LLMs to perform end-to-end generative recommendation. MiniOneRec Kong et al. (2025) is a compact generative recommender that directly generates item IDs autoregressively without additional reasoning. It shares LASAR's base model and prompt format, so performance differences reflect the reasoning mechanism rather than model capacity or prompt design.

Latent reasoning. ReaRec Tang et al. (2025) performs multiple forward passes in continuous hidden space with a globally fixed step count, trained with progressive reasoning curriculum and contrastive alignment. We adopt its published implementation as a representative latent reasoning baseline.

Explicit CoT reasoning. GREAM Hong et al. (2025) is a multi-component framework integrating (1) 19 collaborative-semantic alignment prompts for data augmentation, (2) structured multi-step

CoT reasoning supervision, and (3) SRPO reinforcement learning post-training. We retain GREAM's core CoT reasoning chain while simplifying the 19 alignment prompts to MiniOneRec's prompt format, ensuring that all generative baselines share identical input–output templates and training data. GREAM's own ablation explores all combinations of inference mode (generating CoT text vs. answering directly) and training strategy (SFT vs. SFT+RL); we adopt its strongest configuration (CoT SFT + direct answer inference). We denote this variant as Explicit $\text{CoT}_{\text{GREAM}}$ in result tables.

E Engineering Implementation Details

Table 7 lists the key hyperparameters used across all experiments. All generative methods share the same base model (Qwen3-0.6B, scaled to 1.7B in the main text), data split, SID encoding ($M{=}4$ tokens per item, 256 entries per codebook, up to 256^4 possible SID sequences), and prompt format, so that performance differences reflect the reasoning mechanism rather than model capacity or data. Training uses AdamW with a cosine learning-rate schedule and a 0.08 warmup ratio; the two-stage SFT schedule (Stage 1 plain SID grounding, Stage 2 latent reasoning with KL alignment) and the RL configuration (GRPO with terminal KL and REINFORCE for step allocation) follow the main text. All experiments run on $8{\times}$L40 (48 GB). Inference uses beam search with width 50, ranking candidates by joint token probability, and the bootstrap resampling procedure used for significance testing follows the setup in Appendix C.

Table 7: Implementation details and hyperparameters.

Parameter	Value
Optimizer	AdamW
Batch size	512
LR scheduler	Cosine, warmup ratio 0.08
Max sequence length	512
Stage 1 epochs	10
Stage 1 learning rate	5×10^{-4}
Stage 2 epochs	20
Stage 2 learning rate	5×10^{-5}
Max latent steps (N_{max})	8
α_{align} (align_weight)	0.1
β_{policy} (policy_weight)	0.1
RL learning rate	1×10^{-5}
RL epochs	1
RL num_generations (G)	8
RL β (GRPO ref-KL coef.)	1×10^{-3}
RL ε (clip ratio)	0.2
RL γ_{RF} (reinforce_coef)	0.01
RL reinforce_n_penalty (λ)	$\{0.0001, 0.0005, 0.001\}$
RL γ_{KL} (terminal align. coef.)	1×10^{-5}
Teacher CoT model	GPT-5
Semantic segmentation	BAAI/bge-small-en-v1.5
Beam search width	50
GPU	$8\times$ L40 (48GB)

LoRA configuration (1.7B). The 1.7B scaling experiments use LoRA with rank $r{=}16$, scaling factor $\alpha_{\text{LoRA}}{=}32$, and dropout 0.05, applied to all attention and MLP projection modules (`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`).

SID item embeddings. Following the Residual Quantization K-Means pipeline of prior generative recommenders Hong et al. (2025); Ju et al. (2025), each item's title and description are concatenated and encoded by Qwen3-4B-Instruct; the item embedding is the mean-pooled last-layer hidden state

over non-padding token positions, computed offline. The `bge-small-en-v1.5` encoder is used only for semantic segmentation of teacher CoT traces.

REINFORCE hyperparameters. The entropy coefficient is η=0.01, and the EMA baseline uses a decay factor of 0.99, i.e., $b_{\text{EMA}} \leftarrow 0.99\, b_{\text{EMA}} + 0.01\, \bar{r}$. The full RL phase runs for one epoch over the training set; the backbone is optimized by GRPO and terminal KL, while the Policy Head is optimized by REINFORCE.

E.1 Variable-Length N Batch Processing

LASAR processes a variable per-sample N, resulting in varying numbers of latent tokens within a batch. LASAR handles this through a unified padding-and-masking strategy (Figure 7) that requires no per-sample branching during the latent loop:

Sample	Prompt (left-padded)			Latent region					Answer
	p_1	p_2	p_3	<s>	<t>	<t>	<t>	<e>	
A (N=2, short)	PAD	x_1	x_2	<s>	<t>	<t>	PAD	<e>	Y_A
B (N=3, long)	x_1	x_2	x_3	<s>	<t>	<t>	<t>	<e>	Y_B
Attn (A)	0	1	1	1	1	1	0	1	1
Attn (B)	1	1	1	1	1	1	1	1	1
Loss mask	0	0	0	0	0	0	0	0	1

Figure 7: Detailed batch layout for variable N: prompt left-padding aligns <s> positions, latent region is right-padded with masked attention, and loss is computed only on answer tokens.

1. **Prompt left-padding**: Left-pad prompts with `pad_token_id` to the batch's maximum prompt length, aligning the first ⟨thought⟩ position across all samples.
2. **Per-sample latent insertion**: Insert N_i copies of ⟨thought⟩ per sample after the prompt, producing naturally different latent region lengths.
3. **Latent region right-padding**: For samples with $N_i < \max(N)$, pad the remaining latent slots with `pad_token_id` (*not* ⟨thought⟩) and set attention mask to 0. The latent loop iterates $\max(N)$ times uniformly; extra steps for short-N samples operate on masked positions and produce hidden states that are ignored by subsequent attention.
4. **Sequence right-padding**: Use `pad_sequence` to align total sequence lengths (prompt + latent region + answer) to the batch maximum.
5. **Loss masking**: The LM loss is computed only on answer tokens, and the alignment loss filters positions exceeding each sample's actual N_i via a `valid_mask`, so padded latent positions contribute zero gradient.

E.2 Prompt Format and CoT Reasoning Details

All generative methods (LASAR, MiniOneRec, LC-Rec, and Explicit CoT$_{\text{GREAM}}$) share an identical prompt template following MiniOneRec Kong et al. (2025), so performance differences reflect the reasoning mechanism rather than prompt design. The template takes the form:

```
Below is an instruction that describes a task, paired with an input
that provides further context.  Write a response that appropriately
completes the request.
### Instruction:
Can you predict the next possible item that the user may expect?
### User Input:
The user has interacted with items <SID_1>, <SID_2>, ..., <SID_L> in
chronological order.  Can you predict the next possible item that the
user may expect?
### Response:
<SID_target>
```

where each <SID_i> is a fixed-length four-token semantic ID (e.g., <|a_125|><|b_109|><|c_135|><|d_125|>), not a natural-language title. Items are listed in chronological order and separated by commas, following the MiniOneRec natural-language template; there is no dedicated reserved boundary token beyond this comma delimiter.

CoT Generation. Following GREAM Hong et al. (2025), we generate the CoT reasoning text offline with GPT-5, producing one trace per training sample in a single pass; this cost is incurred only once, before training, and never at inference. Traces are generated exclusively from training instances and their training-set targets; no validation or test targets are used for trace construction. Generated traces were spot-checked on a sampled subset to confirm overall quality; no per-trace manual filtering was applied. The prompt requests a four-step structure: *(1) analyze the purchase history*, *(2) build user preferences*, *(3) uncover purchase intent*, and *(4) recommend the item*. Steps 1–3 constitute the forward reasoning chain, whereas Step 4 states the recommended item itself.

Step Selection for Latent Alignment. LASAR aligns its latent reasoning steps only to Steps 1–3 of the teacher trace (analyze history, build preferences, uncover intent) and excludes Step 4 before segmentation. Latent alignment targets a plausible *forward inferential trajectory* supported by the interaction history, rather than the answer representation itself. Steps 1–3 form the *evidence* $\rightarrow$ *preference* $\rightarrow$ *intent* forward chain.

CoT Segmentation. The retained CoT text is segmented by a semantic splitter built on sentence embeddings from BAAI/bge-small-en-v1.5 Xiao et al. (2023). The text is first split into sentences; a sliding window of `buffer_size`=2 adjacent sentences forms candidate chunks; each chunk is embedded with bge, and the cosine distance between consecutive chunk embeddings is computed. A breakpoint is inserted wherever this distance exceeds the 80th percentile of all consecutive distances in the trace (`breakpoint_percentile_threshold`=80), so boundaries fall at the points of largest semantic shift. The number of segments N is thus data-dependent, and directly serves as the Policy Head's per-sample supervision label during SFT.

CoT Example. Below is a teacher trace from the Sports dataset (user 6729, history length 5); Steps 1–3 are used for alignment supervision and Step 4 is excluded.

Teacher input — user purchase history (titles shown for brevity):

Item 1: 15 Pack of 16 Gram Threaded CO2 Cartridges.
Item 2: Cateye HL-EL135N Bicycle Head Light.
Item 3: Concept2 Model D Indoor Rowing Machine.
Item 4: Petzl Tikka XP 2 Headlamp.
Item 5: Delta Leonardo Single-Bike Rack.

Next Item: Planet Bike Blinky Super Flash Tail Light.

Teacher output — full CoT (Steps 1–4 as originally generated):

Step 1: Analyze the purchase history:
- Strong cycling focus: CO2 cartridges (tire inflation), a bicycle headlight for visibility, and a wall-mounted bike rack for storage.
- Emphasis on safety/visibility and lighting: bicycle headlight and a high-performance headlamp (Petzl) for hands-free illumination.
- Preference for durable, high-quality, and practical gear with long runtimes, easy installation, and versatility (Cateye light, Petzl headlamp, Concept2 rower).
- Signs of active riding/commuting and home organization for the bike.
- Potential gaps: no rear tail light listed, no mention of reflective gear, no CO2 inflator head (though they may already own one), and no lock or saddle bag.

Step 2: Build user preferences:
- Prioritizes safety and visibility while cycling (front light, lighting redundancy).
- Values long battery life, reliability, and compact, easy-mount designs.
- Favors reputable, performance-oriented brands.
- Seeks practical cycling accessories that complement daily/commuter use and home storage.

Step 3: Uncover purchase intent:
- After adding a front headlight, a highly likely next step is completing the safety setup with a rear tail light

for visibility to traffic.
- A rear blinker with wide side visibility, multiple flash modes, long runtime, and easy mounting aligns with their safety focus and gear choices.
- Brand consistency may be appealing given the existing CatEye headlight.

Step 4: Recommend the item: *(removed before alignment)*
- Title: CatEye Omni 5 Rear Bicycle Tail Light
- Description: The CatEye Omni 5 Rear Tail Light enhances on-road safety with five bright LEDs, 360-degree Omni-directional optics, and multiple modes (constant and flashing) for day and night visibility. Running up to 120 hours on two AAA batteries, it offers long-lasting performance, is weather-resistant, and installs tool-free with a secure FlexTight bracket on seatposts, seatstays, or bags. Compact and lightweight with a quick-release mount, it is an easy, reliable upgrade to complete a commuter-ready lighting setup.

Steps 1–3 are semantically segmented and encoded into alignment anchors; Step 4 is excluded before segmentation. In this trace, Step 1 identifies the absence of a rear tail light despite the presence of a front bicycle light, Step 2 abstracts the user's safety and visibility preference, and Step 3 infers the intent to complete the lighting setup with a rear blinker offering flash modes and side visibility, the same category and attributes as the observed next item. Aligning latent steps to this forward chain (evidence $\rightarrow$ preference $\rightarrow$ intent) provides a semantically relevant trajectory for next-item prediction, even though Step 4's specific pick (a CatEye tail light) is a plausible prediction rather than a replication of the observed item (a Planet Bike Blinky) and is discarded. The model is trained to generate the next item's SID via cross-entropy during the SFT phase. None of this CoT text is decoded at inference: the model aligns its latent hidden states to the Step 1–3 segments via bidirectional KL divergence during SFT, and the Policy Head learns to allocate reasoning depth in proportion to the number of segments.

E.3 SFT Phase Alignment Implementation

Anchor construction. Each CoT anchor is obtained in a *single* forward pass over the full reconstructed CoT text (segments rejoined in order): we take the last-token hidden state at each segment boundary from the final Transformer layer. Each anchor is therefore contextualized by the preceding CoT segments and serves as a reference for the latent trajectory. The CoT anchors are pre-computed offline and kept fixed; gradients are propagated only through the latent states.

During SFT, each latent step's hidden state must be aligned with the corresponding explicit CoT segment's hidden state:

1. **Semantic segment matching**: Use `cumsum` to compute each latent token's ordinal within the batch, vectorized-matching to the corresponding CoT segment.
2. **Boundary safety**: Mask padded latent positions (batch padding to $\max_i N_i$); each non-padding latent position is matched in order to its corresponding CoT segment, since N_i equals the segment count by construction.
3. **Device consistency**: CoT embeddings are preloaded on CPU and dynamically converted to GPU dtype and device during training.

E.4 RL Phase Terminal Alignment

What constrains the latent states during RL. During the RL phase the latent states receive gradients from two sources: (i) the GRPO objective, which scores answer-token log-probabilities and back-propagates through the latent forward pass into the base LM; and (ii) the terminal KL loss on the final latent step. The step-wise alignment used in SFT is not applied here because N is sampled per instance, so variable-length reasoning chains no longer correspond one-to-one to the fixed set of CoT segments. The trajectory is anchored at both ends: it starts from the SFT-aligned policy and its final step is aligned by terminal KL, while GRPO shapes the reasoning in between through the recommendation reward.

RL-phase alignment targets only the **last** latent step of the reasoning chain:

1. **Natural latent embedding collection**: During Phase 2 training, hidden states from each step of the latent loop are automatically collected as a `latent_embs` list (cached from the forward pass, with no extra forward passes needed).
2. **Dynamic step indexing**: `latent_embs[`$N-1$`]` directly retrieves the last step's hidden state for bidirectional KL computation with the CoT final state.
3. **Alignment as direct loss, not reward**: If alignment were a reward component, GRPO's within-group advantage zero-mean property would cancel it out. Adding it as a direct loss to the total objective ensures stable gradient signals.

E.5 Reward Formulation

For each prompt, G candidates are generated via beam search. The exact match reward is:

$$r_{\text{rule}}^{(i)} = \begin{cases} 1 & \text{if } \hat{y}^{(i)} = y^* \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $i \in \{0, \ldots, G-1\}$ indexes the candidate in the beam-searched group (0-indexed ranking position, $i = 0$ for the top-ranked candidate), $\hat{y}^{(i)}$ is its SID, and y^* is the ground-truth SID.

The within-group NDCG reward penalizes non-target candidates by their ranking position within the group. Let $w_i = -1/\log_2(i+2)$ (for $i = 0, \ldots, G-1$) be the position weight, which is negative with larger magnitude for higher-ranked candidates. If the ground-truth item appears among the G candidates, non-target candidates receive:

$$r_{\text{NDCG}}^{(i)} = \begin{cases} 0 & \text{if } \hat{y}^{(i)} = y^* \\ \frac{w_i}{\sum_{j=0}^{G-1} |w_j|} & \text{if } \hat{y}^{(i)} \neq y^* \end{cases} \tag{9}$$

If the ground-truth item is absent from all G candidates, every candidate receives $r_{\text{NDCG}}^{(i)} = 0$. Since r_{rule} is also 0 for all candidates in this case, the group's rewards are all zero, so the normalized advantage $\hat{A}_i$ is zero and the group contributes no gradient.

The final reward is $r_i = r_{\text{rule}}^{(i)} + r_{\text{NDCG}}^{(i)}$, and the group-normalized advantage is $\hat{A}_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G}) + \epsilon}$, where $\epsilon = 10^{-4}$ for numerical stability Shao et al. (2024). The clipped GRPO objective with KL penalty is

$$\begin{aligned} L_{\text{GRPO}} = & -\mathbb{E}\Big[\min\big(\rho_i(\Theta)\,\hat{A}_i,\ \text{clip}(\rho_i(\Theta), 1-\varepsilon, 1+\varepsilon)\,\hat{A}_i\big)\Big] \\ & + \beta\, D_{\text{KL}}\big(\pi_\Theta \,\|\, \pi_{\text{ref}}\big), \end{aligned} \tag{10}$$

where π_Θ is the backbone policy, π_{ref} is a frozen reference policy used only for the KL term, $\pi_{\Theta_{\text{old}}}$ is the rollout policy, $\rho_i(\Theta) = \frac{\pi_\Theta(y_i|x)}{\pi_{\Theta_{\text{old}}}(y_i|x)}$, and ε is the clipping ratio.

E.6 Inference Algorithm

Algorithm 1 outlines LASAR inference: the Policy Head fixes a per-sample step count N_i; prompts are batch-aligned so the recurrent latent loop runs uniformly to $\max(N_i)$ with masked padding for shorter samples; and a trie-constrained beam search decodes the final item SID from the KV cache.

E.7 Training Algorithm

Algorithm 2 summarizes the full LASAR training pipeline: SID construction, offline anchor preparation, two-stage SFT, and RL post-training.

Trie-Constrained Decoding

Since SID tokens are hierarchical codes $(s_1, s_2, \ldots, s_M)$ with each s_j drawn from codebook $\mathcal{C}^{(j)}$, not every token combination corresponds to a valid item. Unconstrained beam search may produce SID sequences absent from the catalog.

Algorithm 1 LASAR Inference

$N_i \leftarrow \text{PolicyHead}(\text{prompt}_i)$ for each sample i
Insert <s> <t> $\times N_i$ <e> after each prompt
Batch-align: left-pad prompts, right-pad latent region with PAD (mask=0)

$h_0, \text{KV} \leftarrow \text{LLM}(\text{prompt tokens})$ ▷ Encode prompt
for $t = 1$ to $\max(N_i)$ **do** ▷ Latent loop (equiv. to Eq. 3 of the main paper via KV cache)
    $h_t \leftarrow \text{LLM}(h_{t-1},\ \text{KV})$ ▷ PAD steps ignored by mask
end for
Update KV cache with latent hidden states $\{h_t\}$ ▷ Prepare for answer generation

Beam search from KV, constrained by item prefix tree ▷ Decode the M-token SID
return Top-K recommended items

Algorithm 2 LASAR Training

Construct item SIDs via Residual Quantization K-Means
Generate and segment teacher CoT traces for training samples
Stage 1: SID grounding
for each Stage-1 minibatch **do**
    Optimize the answer-token cross-entropy loss
end for
Encode full CoT traces with the Stage 1 checkpoint and cache segment-boundary anchors
Stage 2: Latent reasoning SFT
for each Stage-2 minibatch **do**
    Set N_i to the CoT segment count for each sample i
    Execute the masked variable-N batch loop to $\max_i(N_i)$
    Compute SID generation, step-wise alignment, and Policy Head CE (warm-start) losses
    Update the backbone and Policy Head
end for
RL post-training
for each prompt group **do**
    Sample $N \sim \pi_\phi(\cdot \mid h_0)$
    Generate G candidate SIDs and compute group rewards
    Compute GRPO, terminal KL, and REINFORCE objectives
    Update the backbone and Policy Head using their respective objectives
end for

To ensure generation validity, we build a **prefix tree (trie)** over all item SIDs in the catalog. Each root-to-leaf path encodes exactly one item's SID sequence. During beam search, at each decoding step j, the trie is queried with the current prefix $(s_1, \ldots, s_{j-1})$ to retrieve the set of valid next tokens $\mathcal{V}_j \subseteq \mathcal{C}^{(j)}$. A custom `ConstrainedLogitsProcessor` masks out all tokens outside $\mathcal{V}_j$ before the softmax, so every beam is guaranteed to trace a valid root-to-leaf path. This adds negligible overhead since trie lookups are $O(M)$ per step and the masking is a single tensor operation.

Figure 8 shows a simplified trie with M=3 and four tokens per codebook.

F Properties and Rationale of CoT-Guided KL Alignment

This section characterizes the alignment target, the discrepancy controlled by the loss, and the resulting supervision signal. We first establish the relationship between normalized activation profiles and hidden-coordinate differences, then derive an exact identity, a profile error bound, a local quadratic approximation, and fixed-anchor gradients. These properties explain the regularizer's role in LASAR; its recommendation benefit is evaluated in Tables 2 and 9.

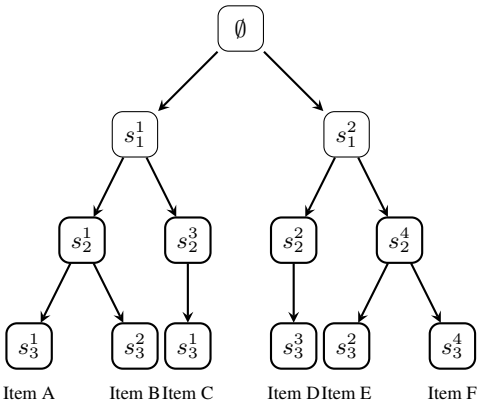


Figure 8: Trie over item SIDs (M=3, $|\mathcal{C}^{(j)}|$=4). Each root-to-leaf path is a valid item. At step j, only children of the current prefix are allowed, preventing invalid sequences.

F.1 Setup and Relative Activation Structure

For one matched latent step, let $h \in \mathbb{R}^d$ denote the trainable latent state and $a \in \mathbb{R}^d$ the fixed CoT anchor. With temperature one and natural logarithms, define

$$p_j = \frac{e^{h_j}}{\sum_k e^{h_k}}, \qquad q_j = \frac{e^{a_j}}{\sum_k e^{a_k}}, \qquad J(h,a) = \frac{1}{2}\left[D_{\mathrm{KL}}(p\|q) + D_{\mathrm{KL}}(q\|p)\right]. \tag{11}$$

The index j denotes a hidden coordinate, and p, q are relative activation profiles over hidden channels. The semantic reference comes from the contextual CoT encoding: the Stage 1 checkpoint encodes the full trace, and its final-layer state at each segment boundary forms an anchor. The latent backbone starts from the same checkpoint, providing a common initial coordinate system. The anchors remain fixed during training.

Proposition F.1 (Relative activation matching). For any pair of coordinates j, k,

$$\log \frac{p_j}{p_k} = h_j - h_k, \qquad \log \frac{q_j}{q_k} = a_j - a_k. \tag{12}$$

Furthermore,

$$J(h,a) = 0 \quad \Longleftrightarrow \quad p = q \quad \Longleftrightarrow \quad h - a = c\mathbf{1} \text{ for some } c \in \mathbb{R}. \tag{13}$$

Proof. Taking the ratio of two softmax coordinates cancels the normalization: $p_j/p_k = e^{h_j - h_k}$. Taking logarithms gives the first identity, and the second follows identically. Both directional KL terms are nonnegative and vanish exactly when $p = q$. Equality of the profiles implies equality of all log ratios, hence $h_j - a_j = h_k - a_k$ for every pair j, k. Conversely, adding the same constant to every coordinate leaves softmax unchanged. □

Thus zero alignment loss matches all pairwise activation differences, with freedom to shift all coordinates by a common constant. Scaling the states generally changes their profiles. This characterizes the objective's target relative to raw-vector MSE, which penalizes a common shift, and cosine alignment, which is invariant to positive rescaling.

F.2 Exact Identity and Profile Error Bound

Proposition F.2 (Hidden-coordinate identity). The symmetric objective satisfies

$$J(h,a) = \frac{1}{2}\sum_j (p_j - q_j)\log\frac{p_j}{q_j} = \frac{1}{2}(h-a)^\top (p-q). \tag{14}$$

Proof. Expanding the two directional KL terms and combining them gives the first equality. Let $Z_h = \sum_k e^{h_k}$ and $Z_a = \sum_k e^{a_k}$. Since

$$\log\frac{p_j}{q_j} = h_j - a_j - \log Z_h + \log Z_a, \tag{15}$$

the normalization terms are constant across coordinates and vanish when summed against $p - q$, because $\sum_j (p_j - q_j) = 0$. The remaining terms yield the second equality. □

This is an exact relationship between the original activation discrepancy $h - a$ and the normalized profile discrepancy $p - q$.

Proposition F.3 (Profile error control). For any finite h, a,

$$\|p - q\|_1 \leq \sqrt{2J(h, a)}. \tag{16}$$

For N matched steps with $L_{\text{align}} = N^{-1} \sum_{t=1}^{N} J(h_t, a_t)$,

$$\frac{1}{N} \sum_{t=1}^{N} \|p_t - q_t\|_1 \leq \sqrt{2L_{\text{align}}}. \tag{17}$$

Proof. Pinsker's inequality with natural logarithms gives $D_{\text{KL}}(p\|q) \geq \frac{1}{2}\|p - q\|_1^2$. Applying the same inequality to the reverse direction shows that their average has the same lower bound, proving Eq. 16. Applying this result at each step and Jensen's inequality for the concave square-root function yields

$$\frac{1}{N} \sum_t \|p_t - q_t\|_1 \leq \frac{1}{N} \sum_t \sqrt{2J(h_t, a_t)} \leq \sqrt{\frac{2}{N} \sum_t J(h_t, a_t)}, \tag{18}$$

which is Eq. 17. □

Reducing the SFT alignment objective therefore controls the average discrepancy between the normalized profiles along the supervised trajectory. This bound concerns profile error; the common additive shift identified in Proposition F.1 remains unconstrained by alignment.

F.3 Local Geometry of the Objective

Proposition F.4 (Local quadratic form). For fixed finite a, write $h = a + \delta$ and let $F(q) = \text{diag}(q) - qq^\top$. As $\delta \to 0$,

$$\begin{aligned} J(a + \delta, a) &= \frac{1}{2} \delta^\top F(q) \delta + O(\|\delta\|_2^3) \\ &= \frac{1}{2} \left[\sum_j q_j \delta_j^2 - \left(\sum_j q_j \delta_j \right)^2 \right] + O(\|\delta\|_2^3). \end{aligned} \tag{19}$$

Proof. The softmax Jacobian at a is $F(q)$, so Taylor expansion gives $p - q = F(q)\delta + O(\|\delta\|_2^2)$. Substitution into Eq. 14 yields the first equality; expanding the quadratic form gives the second. □

The leading term is one half of the anchor-weighted variance of coordinate discrepancies: $\text{Var}_{j \sim q}(\delta_j)$. A uniform shift has zero variance, whereas unequal coordinate deviations receive a penalty weighted by the anchor profile. This local approximation explains how the loss penalizes changes in relative activation structure near the anchor. Both directional KL terms have the same second-order expansion at the match point. We adopt symmetric KL as our alignment objective; on Beauty, it achieves the best results among the tested alignment objectives (Table 2).

F.4 Gradients with Fixed CoT Anchors

Proposition F.5 (Latent-state gradients). Holding a and q fixed,

$$\nabla_h D_{\text{KL}}(q\|p) = p - q, \qquad \nabla_h D_{\text{KL}}(p\|q) = F(p)(h - a), \tag{20}$$

$$\nabla_h J(h, a) = \frac{1}{2} \left[(p - q) + F(p)(h - a) \right], \tag{21}$$

where $F(p) = \text{diag}(p) - pp^\top$.

Proof. For the first direction, $D_{\mathrm{KL}}(q\|p) = \sum_j q_j \log q_j - \sum_j q_j \log p_j$. The first term is fixed, and the second is cross-entropy with target q, whose logit gradient is $p - q$. For the other direction, the derivative with respect to p_j is $\log(p_j/q_j) + 1$. Differentiating through softmax therefore gives

$$\nabla_h D_{\mathrm{KL}}(p\|q) = F(p)\big[\log(p/q) + \mathbf{1}\big]. \tag{22}$$

Using $\log(p/q) = h - a - (\log Z_h - \log Z_a)\mathbf{1}$ and $F(p)\mathbf{1} = 0$ reduces this expression to $F(p)(h-a)$. Averaging the two directional gradients proves Eq. 21. □

Both directions thus update the latent branch even when the CoT anchors are detached. The combined gradient contains a direct profile-matching signal and an activation-discrepancy signal weighted by the current latent profile.

F.5 Role in LASAR Training

Intermediate supervision during SFT. The answer-token loss supervises the final recommendation, while step-wise alignment supplies a direct reference at every matched latent state. Successive anchors come from successive boundaries of a contextualized CoT trace, so the references vary with reasoning progress. Propositions F.1–F.3 identify the relative activation structure being matched and bound its average profile discrepancy; Proposition F.5 specifies the gradients supplied to the latent states. Together, these properties explain how the regularizer guides the intermediate trajectory toward CoT-derived references while SID cross-entropy trains recommendation behavior.

Endpoint supervision during RL. When the Policy Head explores variable depth N, the terminal loss applies the same operator to h_N and the final CoT anchor. It guides the endpoint profile without assigning a separate anchor to every intermediate step. Gradients still propagate through the latent loop, and the intermediate path can adapt to the sampled depth under the RL objectives.

Empirical choice and interpretation. We select symmetric KL from the tested KL, cosine, and MSE losses using the Beauty ablation (Table 2). Table 9 evaluates the adopted regularizer against unaligned latent reasoning on the other datasets. The propositions characterize its alignment target, profile error control, local geometry, and supervision signal. Recommendation improvements are supported by the experiments. Semantic guidance is inherited from the CoT anchors in the shared backbone coordinate system, and task performance is learned jointly with the recommendation objective.

G SFT Labels and Sample Complexity

In the SFT phase, the Policy Head's supervision labels come from the Teacher CoT semantic segment count. Table 8 shows that samples with more segments consistently have longer histories and higher category diversity: interaction histories that span more diverse categories and longer sequences yield more CoT segments and thus a larger predicted depth. The Policy Head therefore learns a difficulty-aware allocation, assigning more latent steps to more complex user patterns.

Fixed-N comparison protocol. The fixed-$N \in \{1, 4, 8\}$ curves in the Force-N analysis (main text, RQ3) are produced from the trained LASAR checkpoint: at inference we disable the Policy Head and set every sample to a constant number of latent steps. This analysis asks whether the depth the Policy Head assigns to each sample actually helps, so we vary only the allocation while leaving the model itself unchanged. Because every learned parameter is held fixed, the performance gap reflects how reasoning depth is distributed across samples.

Within-group depth analysis. The adaptive depth assignment serves only to define groups: test users are binned into three depth levels per dataset by the N the Policy Head assigns (ranges in Figure 9). We then disable the Policy Head and force each level in turn; a cell reports the group's average HR@10 under that forced level, averaging over all users of the group and all N within the level. Every group attains its best value under its own assigned level on all three datasets: groups assigned shallow depths degrade when forced deeper (Sports L_1: $2.73\% \to 1.88\%$ under L_2), the deep group degrades when forced shallower (Sports L_3: $2.61\% \to 2.43\%$ under L_1; Instruments L_3: $10.32\% \to 9.94\%$; Beauty L_3: $6.67\% \to 5.56\%$), and the mid group is nearly depth-insensitive.

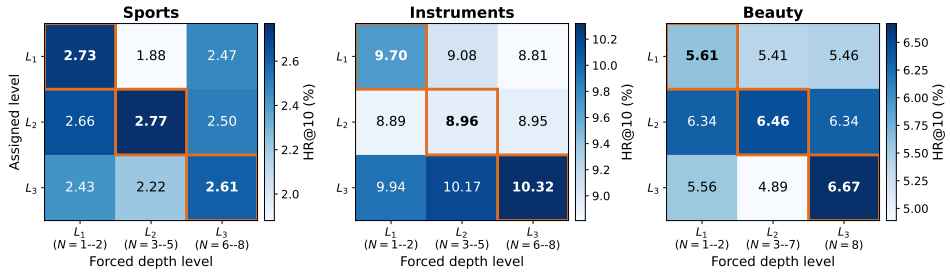


Figure 9: Within-group HR@10 under depth-level restriction. Rows: test users binned by the Policy Head's assigned depth level; columns: forced depth level (ranges on the x-axis); a cell averages the hit indicator over all users of its row group under all depths in its column level. Outlined cells mark each group's own assigned level, where every group attains its best HR@10.

Table 8: Teacher CoT segment count vs. sample complexity on Sports.

Segments	Prop.	Hist. Len.	Cat. Div.
3 segments	83.4%	6.9	6.2
4 segments	16.3%	**8.5**	**7.5**
5 segments	0.2%	**9.8**	**8.8**
Trend		↑	↑

H Alignment Ablation on Sports, Instruments, and Steam

Table 9 extends the alignment ablation (Table 2) beyond Beauty to the remaining three datasets.

Table 9: Alignment ablation on Sports, Instruments, and Steam. Best in **bold**.

	Model	N@5	N@10	HR@5	HR@10
Sports	Pure SFT	0.0105	0.0136	0.0163	0.0260
	+ Latent (w/o align.)	0.0104	0.0139	0.0168	0.0276
	+ KL Alignment	**0.0109**	**0.0143**	**0.0174**	**0.0281**
Instruments	Pure SFT	0.0580	0.0634	0.0720	0.0888
	+ Latent (w/o align.)	0.0517	0.0576	0.0693	0.0878
	+ KL Alignment	**0.0586**	**0.0643**	**0.0731**	**0.0910**
Steam	Pure SFT	0.0272	0.0372	0.0449	0.0762
	+ Latent (w/o align.)	0.0281	0.0384	0.0448	0.0771
	+ KL Alignment	**0.0302**	**0.0409**	**0.0480**	**0.0815**

KL alignment yields the best results on every dataset, improving NDCG@10 over Pure SFT by 5.1% on Sports, 1.4% on Instruments, and 9.9% on Steam. The unaligned variant behaves inconsistently, with slight gains on some Sports and Steam metrics but notable degradation on Instruments (−9.1% NDCG@10), indicating that KL-based alignment is not dataset-specific: it provides stable improvement, whereas unaligned latent reasoning does not reliably benefit recommendation.