LASAR: Latent Adaptive Semantic Aligned Reasoning for Generative Recommendation

Yiwen Chen[1,2] **Fuwei Zhang**[1] **Zehao Chen**[1] **Deqing Wang**[1] **Hehan Li**[2] **Peizhi Xu**[2]
Hanmeng Liu[2] Shuanglong Li[2] Xin Pei[2] Fuzhen Zhuang[1,†] Zhao Zhang[1]

[1]School of Artificial Intelligence, Beihang University, Beijing, China
[2]Baidu, Beijing, China
zhuangfuzhen@buaa.edu.cn

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. Adapting it reveals three unique challenges: (1) the gap between prior-less Semantic ID (SID) symbols and continuous latent reasoning—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 simultaneously improving recommendation quality. Experiments on three real-world datasets demonstrate that LASAR outperforms all baselines. It adds marginal inference latency and is roughly 20× faster than generating explicit CoT text.

1 Introduction

Large Language Models (LLMs) in recommender systems are advancing rapidly along two paths. One is **generative recommendation**: P5 [13] and M6-Rec [5] pioneered the unified recommendation pretraining paradigm, TIGER [32] introduced Semantic ID-based generative retrieval, LC-Rec [57] integrated collaborative semantics into LLMs for direct item ID generation, while other studies [1, 19, 24, 44] further advanced generative recommendation from alignment, indexing, and collaboration perspectives. MiniOneRec [23] built the first fully open-source generative recommendation framework. The other path is **LLM reasoning**: CoT [45, 22] improved task performance by explicitly generating intermediate reasoning steps, while subsequent studies [43, 51, 2, 10, 28] extended the boundaries of explicit reasoning. However, even as DeepSeek-R1 [6] and O1 [30] have pushed explicit reasoning to its limits, the fundamental bottleneck of reasoning latency remains

Preprint.

unresolved. To address this, Coconut [16] 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 [7, 8, 33, 12, 35, 48, 3, 11, 39] advanced this paradigm from various angles.

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 [41], LARES [27], and other studies [55, 50, 37] introduced latent reasoning, but they target the ID-embedding discriminative ranking paradigm, not the mainstream decoder-only generative paradigm. In generative recommendation, GREAM [18] and other studies [42, 9, 25, 52] 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 [56, 15] 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 realize complete latent reasoning with recurrent hidden-state feedback and adaptive step control in mainstream generative recommendation.**

After integrating Coconut-style latent reasoning into a generative recommendation framework, we find it is **not a "free lunch"**. Directly transplanting Coconut's training recipe actually degrades recommendation performance. We identify three challenges unique to 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 3). **(2) Representation drift.** Recommendation lacks ground-truth reasoning chain supervision: there is no "standard reasoning process" to reference. Directly introducing latent reasoning without semantic constraints causes hidden states to drift in continuous space toward meaningless representations, and naively adding latent reasoning without alignment brings negligible gains (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 work to realize complete latent reasoning with recurrent hidden-state feedback and adaptive step control in mainstream generative recommendation. LASAR follows an SFT-then-RL training pipeline with systematic solutions for the above challenges:

(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 3). **(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 anchor the latent states on the correct semantic 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, latent reasoning adds only marginal latency overhead (approximately $20\times$ faster than generating explicit CoT) (Table 4).

Experiments on three real-world datasets show LASAR achieves the best performance on nearly all metric-dataset combinations. Ablation reveals the failure modes of latent reasoning and confirms that semantic alignment, Terminal KL, and REINFORCE are all indispensable.

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 $\mathbf{e}_i$:

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

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 [18, 20]. 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 $\mathcal{S}$: $X = \big[\text{text}_{\text{nl}},\ \mathrm{SID}(i_1),\ \mathrm{SID}(i_2),\ \ldots,\ \mathrm{SID}(i_t)\big]$, where each $\mathrm{SID}(i_j) = (s_1, \ldots, s_M)$ contributes M special tokens. The model generates the target sequence $\mathbf{Y} = \mathrm{SID}(i_{t+1})$ 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 1). 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 proceeds as:

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

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 [16]: 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: $\pi_\phi(\cdot \mid h_0) = \mathrm{Softmax}\big(W_2 \cdot \tanh(W_1 \cdot h_0 + b_1) + b_2\big)$

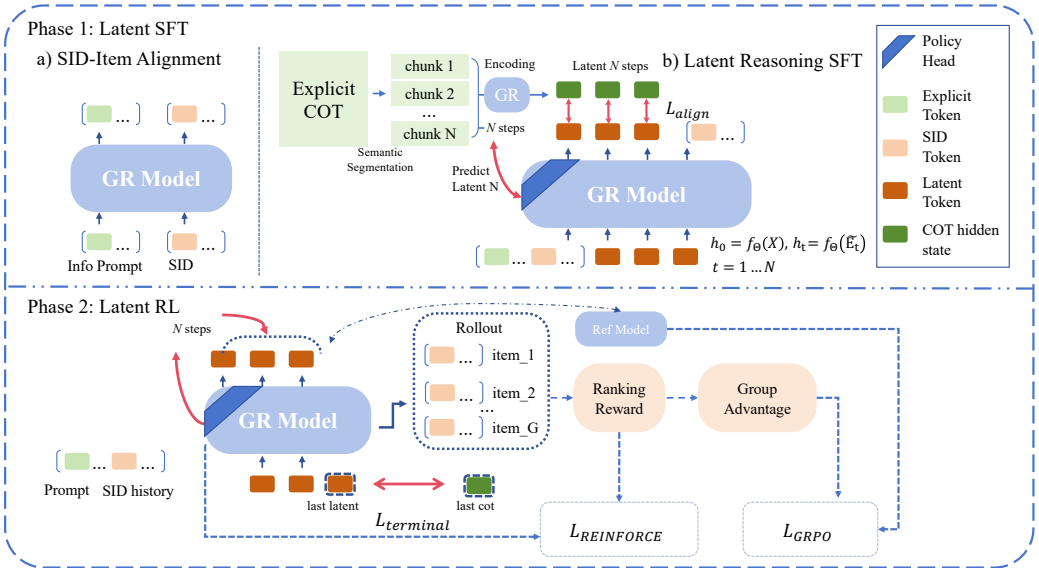


Figure 1: LASAR framework overview. A hidden-state feedback loop iteratively refines latent tokens in continuous space, while a Policy Head predicts per-sample reasoning depth N for adaptive reasoning. SFT: two-stage decoupling + step-wise bidirectional KL alignment with CoT anchors. RL: GRPO (generation quality) + REINFORCE (adaptive reasoning efficiency) + Terminal KL (semantic consistency).

with output dimension $N_{\max}$ (maximum reasoning steps, default 8). During SFT, the head uses $N = \text{argmax}(\pi_\phi)$ and is trained with cross-entropy loss, where the number of CoT semantic segments per sample (Section 2.3) serves as the supervision label. During RL (Section 2.4), the head switches to sampling $N \sim \pi_\phi$ and is optimized via REINFORCE (Section 2.4). 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-N 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 2; Appendix E.1).

Sample	Latent region							Answer
	<s>	<t>	<t>	<t>	<t>	<t>	<e>	
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 2: 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—without semantic anchors, latent reasoning has no stable trajectory to follow. Figure 3 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.

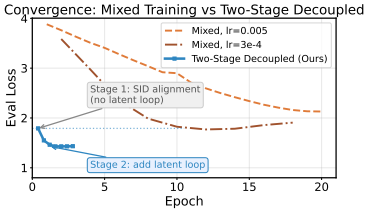


Figure 3: Two-stage decoupling vs. mixed training convergence.

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 $\approx$1.44 within 4 epochs, reducing total training time from 20+ hours to approximately 6 hours.

While decoupling resolves the semantic grounding gap (Challenge 1), representation drift (Challenge 2) remains: latent reasoning in continuous space lacks the natural constraints of discrete tokens, and hidden states can drift toward meaningless representations without explicit semantic guidance. We address this through CoT semantic alignment, anchoring the latent reasoning trajectory to explicit CoT reasoning segments.

CoT Semantic Alignment. Ablation (Table 2) confirms representation drift: naively adding latent reasoning without alignment degrades performance. To prevent this, 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) [46]. The same backbone model then encodes each segment (offline, before training) and extracts the last-token hidden state from the final Transformer layer as pre-computed alignment anchors. This design shares similarities with CODI [36]'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 to preserve probability distribution shape. The number of resulting segments also serves as the supervision label for the Policy Head's step 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 hidden state obtained by encoding the t-th explicit CoT segment through the same backbone.

We adopt bidirectional KL over cosine distance or MSE because it preserves distributional shape information. Ablation (Section 3.3) confirms it is the only alignment method among the compared alternatives yielding positive improvement.

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 — Generation Quality Optimization. For each prompt, G candidates are generated, and the reward combines exact match and ranking quality following prior work [23]: $r = r_{\text{rule}} + r_{\text{NDCG}}$, where r_{rule} is binary (1 if hit, 0 otherwise) and r_{NDCG} penalizes non-target candidates by ranking position (formal definitions in Appendix E.5). The clipped GRPO objective with KL penalty is:

$$L_{\text{GRPO}} = -\mathbb{E}\Big[\min\big(\rho_i(\Theta)\hat{A}_i,\ \text{clip}(\rho_i(\Theta), 1{-}\varepsilon, 1{+}\varepsilon)\hat{A}_i\big)\Big] + \beta\, D_{\text{KL}}\big(\pi_\Theta \| \pi_{\text{ref}}\big) \tag{5}$$

where π_Θ is the backbone policy, π_{ref} a frozen reference, $\rho_i(\Theta) = \frac{\pi_\Theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}$, ε is the clipping ratio, and $\hat{A}_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^G)}{\text{std}(\{r_j\}_{j=1}^G)}$ is the group-normalized advantage [34].

REINFORCE — Adaptive Step Optimization. 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 $N \sim \pi_\phi(\cdot \mid h_0)$ (differing from the SFT phase's argmax), and the sampling strategy is optimized via the REINFORCE algorithm:

$$L_{\text{REINFORCE}} = -\mathbb{E}_{N\sim\pi_\phi}\left[(R_{\text{group}} - b_{\text{EMA}} - \lambda N) \cdot \log \pi_\phi(N \mid h_0)\right] - \eta \cdot H(\pi_\phi) \tag{6}$$

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 KL — Semantic 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. This ensures the reasoning endpoint stays on the correct semantic trajectory regardless of N. The terminal KL loss is added directly to the total loss.

RL Total Loss. The three components are combined as $L_{\text{total}} = L_{\text{GRPO}} + \gamma_{\text{KL}} \cdot L_{\text{Terminal KL}} + \gamma_{\text{RF}} \cdot L_{\text{REINFORCE}}$. The three components are coordinated: GRPO improves generation quality, Terminal KL maintains semantic consistency, and REINFORCE optimizes reasoning efficiency. Ablation (Section 3.3) confirms that all three components are indispensable.

3 Experiments

We evaluate LASAR on three Amazon product review 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)** Does adaptive step allocation outperform fixed configurations (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 [29], Beauty, Instruments, and Sports, which are widely recognized benchmarks in sequential recommendation research. We apply the 5-core filtering protocol and adopt the leave-one-out evaluation, following prior works [21, 40, 17, 32, 57, 23]. Beauty has 22K users, 12K items, and 176K interactions; Instruments has 25K users, 10K items, and 74K interactions; Sports has 36K users, 18K items, and 107K interactions. Sparsity ranges from 99.935% to 99.984%. See Table 6 in Appendix B for detailed dataset statistics.

Baselines. We compare against six baselines spanning four categories: traditional sequential models (SASRec [21], GRU4Rec [17]), LLM-based generative methods (LC-Rec [57], MiniOneRec [23]), latent reasoning (ReaRec [41]), and explicit CoT reasoning (GREAM [18]). All generative baselines share the same base model, prompt, and training data; performance differences thus reflect the reasoning mechanism alone. 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 $\text{CoT}_{\text{GREAM}}$. 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

Table 1: Main results across three Amazon 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

generates M SID tokens per candidate, and beam search ensures that the top-K items are ranked by joint token probability.

Implementation Details. All generative methods share the same SID encoding, training data, and prompt format. CoT reasoning text follows GREAM's structured format, while the input prompt is simplified to MiniOneRec's template for fair comparison. We use Qwen3-0.6B [49] as the base model and further scale to 1.7B (Section 3.5), optimized with AdamW (cosine LR schedule with 0.08 warmup). Each item is represented as $M{=}4$ SID tokens (i.e., 256^4 unique codes). Teacher CoT reasoning text is generated by GPT-5 (see Appendix E.2 for the prompt template and CoT examples). Stage 1 (SID alignment) runs with lr $5{\times}10^{-4}$; Stage 2 (latent loop) with lr $5{\times}10^{-5}$, both with early stopping. RL uses GRPO with lr 10^{-5}, $G{=}8$ generations, KL penalty $\beta{=}10^{-3}$, and REINFORCE step penalty $\lambda{=}5{\times}10^{-4}$. All experiments run on $8{\times}$L40 (48 GB). See Table 7 in Appendix E for the full hyperparameter table and candidate ranges.

3.2 Main Results (RQ1)

Table 1 presents the results across all three 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, with the sole exception of Beauty HR@20 where Explicit CoT yields marginal gains at high recall cutoffs. A clear trend emerges: gains are largest on the sparsest dataset and consistent across others. On Sports (most sparse), LASAR outperforms MiniOneRec by a large margin, while on Instruments and Beauty, the advantage is more modest but consistent, suggesting latent reasoning is particularly beneficial under high sparsity, where the model's semantic understanding compensates for limited collaborative signals. Notably, LASAR consistently outperforms both direct generation and explicit CoT reasoning. The latter's limited gains may stem from representation interference: teaching the model to decode discrete reasoning text creates tension between language modeling and collaborative filtering objectives. LASAR avoids this by reasoning entirely in continuous latent space, injecting multi-step reasoning without disrupting the SID generation objective. Bootstrap tests show significance on Sports and Instruments ($p < 0.05$), and marginal significance on Beauty ($p < 0.1$, except $K{=}20$).

3.3 Ablation Studies (RQ2)

We next decompose LASAR to understand which components drive its gains. Table 2 and Table 3 present ablations across the SFT and RL phases, examining the latent reasoning mechanism, alignment method, two-stage decoupling, and RL objectives individually.

SFT Phase Ablation. Table 2 compares different alignment strategies during the SFT phase on Beauty (see Table 9 for Sports and Instruments). Latent reasoning without alignment yields almost no improvement over the Pure SFT baseline (+0.4% NDCG@10), confirming that **latent reasoning**

requires alignment to be effective. Among the three alignment variants, only KL divergence yields positive improvement, whereas Cosine and MSE both underperform even the *no-alignment* baseline, indicating that alignment choice is critical: KL divergence preserves distributional shape, whereas simpler metrics collapse the representation structure.

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
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%
+ KL Alignment	✓	✓	**KL**	**0.0217**	**0.0285**	**0.0340**	**0.0552**	**+2.9%**
+ Cosine Alignment	✓	✓	Cosine	0.0211	0.0277	0.0341	0.0543	0.0%
+ MSE Alignment	✓	✓	MSE	0.0187	0.0245	0.0295	0.0477	−11.6%

RL-Phase Ablation. The SFT ablation confirms that KL alignment enables latent reasoning to work, and we now ask whether the three RL components (GRPO, Terminal KL, REINFORCE) are equally indispensable. Table 3 presents the RL-phase ablation on Beauty, chosen for its large interaction count (176K) that provides stable gradient signals. Key findings are further validated on Sports in Section 3.4.

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

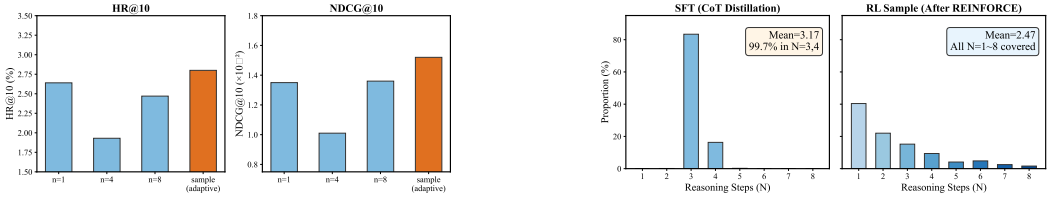
Model	Latent	Terminal KL	REINFORCE	N@5	N@10	HR@5	HR@10	Mean N	Δ N@10
MiniOneRec	N/A	N/A	N/A	0.0232	0.0295	0.0352	0.0542	N/A	–
RL w/ latent reasoning	✓			0.0227	0.0287	0.0346	0.0533	3.59	−2.7%
+ Terminal KL Alignment	✓	✓		0.0233	0.0294	0.0353	0.0543	4.20	+2.4%
+ REINFORCE (LASAR)	✓	✓	✓	**0.0239**	**0.0303**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

Naively adding latent reasoning without proper RL training degrades performance, confirming **latent reasoning is not a free improvement**. Terminal KL alignment recovers this drift and improves NDCG@10, but also increases Mean N: alignment makes each step productive, and without REINFORCE's step penalty there is no incentive to compress depth. 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 each provide independent and complementary gains.

3.4 Step Optimization Analysis (RQ3)

The ablation in the previous section showed that REINFORCE effectively compresses reasoning steps while improving quality. We now investigate this mechanism in detail: whether adaptive step allocation outperforms fixed configurations, and how REINFORCE reshapes the step distribution during training. We note that the SFT supervision labels (Teacher CoT segment counts) correlate positively with sample complexity: on Sports, 83.4% of samples have 3 segments (avg. history length 6.9, category diversity 6.2), 16.3% have 4 segments (hist. 8.5, cat. div. 7.5), and 0.2% have 5 segments (hist. 9.8, cat. div. 8.8) (Table 8 in Appendix F), confirming the Policy Head learns meaningful difficulty-aware depth allocation during SFT.

Force N Experiment: Adaptive Outperforms All Fixed Configurations. Figure 4a compares adaptive sampling against three fixed-N configurations on Sports. Adaptive (HR@10 $= 2.80\%$) surpasses all fixed configurations. Notably, fixed N=4 performs worst (1.93%): forcing all samples through 4 latent iterations introduces unnecessary interference, whereas N=1 preserves representations without disruption.



(a) Force N comparison.

(b) Step distribution: SFT vs. RL.

Figure 4: Adaptive step allocation on Sports: adaptive N outperforms all fixed configurations, and RL redistributes steps toward fewer but better-allocated depths.

RL Dynamics and Per-N Analysis. Figure 5a tracks the RL training dynamics on Sports: Mean N drops sharply from ≈3.4 to ≈1.9 in early training and stabilizes, while Reward rises consistently. REINFORCE thus compresses the average step budget without sacrificing—and indeed improving—quality (Table 3). Figure 4b shows the distribution shifts from SFT's concentrated N=3, 4 (99.7%) to a broader N=1–8 coverage (Mean $= 2.47$).

Crucially, REINFORCE does not simply minimize depth. Per-N analysis (metrics per step count; Figure 5b) reveals that the Policy Head learns a selective allocation strategy: most samples receive shallow reasoning ($N \leq 4$) for efficiency, while the hardest cases are assigned deep reasoning ($N \geq 7$). The intermediate depths ($N=5, 6$) see few samples and lower HR@10, as they are neither efficient nor thorough enough. The key evidence is the gap between Force N and per-N results: forcing all samples to $N=4$ yields the worst result (1.93%), yet the Policy Head achieves 3.38% at $N=4$ for selected samples—confirming it identifies which samples genuinely benefit from deeper reasoning.

(a) RL training dynamics.

(b) Per-N HR@10.

Figure 5: (a) RL training dynamics on Sports: Mean N drops early then stabilizes while Reward rises. (b) Per-N HR@10: non-monotonic relationship with a peak at $N=4$ among mid-range depths, as the Policy Head selectively assigns deeper reasoning to samples that benefit.

3.5 Inference Efficiency and Model Scaling (RQ4)

Beyond recommendation quality, latent reasoning avoids the latency overhead of explicit text generation. We now evaluate LASAR's inference efficiency and its scalability to larger models.

Table 4: Inference efficiency.

Dataset	Method	Time/Sample (s)	Total Time
Beauty	MiniOneRec	0.27	12min
	LASAR	0.29	13min
	Explicit $\text{CoT}_{\text{GREAM}}$ (CoT Gen.)	7.0	5.5h
Instruments	MiniOneRec	0.25	13min
	LASAR	0.29	15min
	Explicit $\text{CoT}_{\text{GREAM}}$ (CoT Gen.)	6.5	5h
Sports	MiniOneRec	0.30	22min
	LASAR	0.32	24min
	Explicit $\text{CoT}_{\text{GREAM}}$ (CoT Gen.)	7.0	8.5h

Inference Efficiency. Table 4 summarizes inference latency across all three datasets under beam width 50 on 8×L40. LASAR adds only negligible overhead over MiniOneRec (≈7–16% depending on dataset), with latent reasoning contributing tens of milliseconds per sample. In contrast, generating explicit CoT text (GREAM's reasoning mode) is over 20× slower, since all of its overhead comes from autoregressive decoding of long reasoning chains before producing the final recommendation. This trend is consistent, placing LASAR at the Pareto-optimal position on the efficiency–effectiveness frontier.

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 $\text{CoT}_{\text{GREAM}}$	0.0293	0.0553	0.0295	0.0561

Model Scaling. To assess scalability, we scale from the default Qwen3-0.6B to Qwen3-1.7B (LoRA) on Beauty, both trained with SFT+RL. Table 5 shows LASAR achieves the best performance at both scales. Both LASAR and MiniOneRec improve comparably in N@10, confirming that latent reasoning does not limit capacity gains. However, LASAR's HR@10 improvement notably exceeds MiniOneRec's, suggesting latent reasoning benefits more from additional capacity at top-ranked positions. Explicit $\text{CoT}_{\text{GREAM}}$ shows the smallest gains, indicating its discrete token decoding bottleneck limits scaling benefits.

4 Conclusion

This paper proposes LASAR, the first framework to realize complete latent reasoning with recurrent hidden-state feedback and adaptive step control in mainstream decoder-only 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 three datasets show that LASAR surpasses all baselines, with the largest gains on the sparsest dataset. Ablation confirms that semantic alignment, Terminal KL, and REINFORCE are all indispensable.

Limitations and future work. A shared challenge across Coconut-style latent reasoning methods is that the hidden-state feedback loop precludes teacher forcing on latent tokens, requiring sequential forward passes that are difficult to parallelize. This is a systemic bottleneck for the paradigm rather than specific to LASAR. Future work may explore efficient execution strategies for the latent loop, and extend latent reasoning to broader recommendation scenarios such as conversational and cross-domain settings.

Acknowledgments and Disclosure of Funding

References

[1] 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 Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song, editors, *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18-22, 2023*, pages 1007–1014. ACM, 2023. doi: 10.1145/3604915.3608857. URL https://doi.org/10.1145/3604915.3608857.

[2] 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 Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *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 2014, February 20-27, 2024, Vancouver, Canada*, pages 17682–17690. AAAI Press, 2024. doi: 10.1609/AAAI.V38I16.29720. URL https://doi.org/10.1609/aaai.v38i16.29720.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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 Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/dfc310e81992d2e4cedc09ac47eff13e-Abstract-Conference.html.

[11] 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.

[12] 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.

[13] 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 Jennifer Golbeck, F. Maxwell Harper, Vanessa Murdock, Michael D. Ekstrand, Bracha Shapira, Justin Basilico, Keld T. Lundgaard, and Even Oldridge, editors, *RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, September 18 - 23, 2022*, pages 299–315. ACM, 2022. doi: 10.1145/3523227.3546767. URL https://doi.org/10.1145/3523227.3546767.

[14] 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, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ph04CRkPdC.

[15] 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.

[16] 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.

[17] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.06939.

[18] 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.

[19] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. How to index item ids for recommendation foundation models. In Qingyao Ai, Yiqin Liu, Alistair Moffat, Xuanjing Huang, Tetsuya Sakai, and Justin Zobel, editors, *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region, SIGIR-AP 2023, Beijing, China, November 26-28, 2023*, pages 195–204. ACM, 2023. doi: 10.1145/3624918.3625339. URL https://doi.org/10.1145/3624918.3625339.

[20] 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 Meeyoung Cha, Chanyoung Park, Noseong Park, Carl Yang, Senjuti Basu Roy, Jessie Li, Jaap Kamps, Kijung Shin, Bryan Hooi, and Lifang He, editors, *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025, Seoul, Republic of Korea, November 10-14, 2025*, pages 6420–6425. ACM, 2025. doi: 10.1145/3746252.3761612. URL https://doi.org/10.1145/3746252.3761612.

[21] Wang-Cheng Kang and Julian J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 197–206. IEEE Computer Society, 2018. doi: 10.1109/ICDM.2018.00035. URL https://doi.org/10.1109/ICDM.2018.00035.

[22] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

[23] 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.

[24] 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.

[25] 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, 2025. URL https://openreview.net/forum?id=YBRU9MV2vE.

[26] 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, 2025. doi: 10.1145/3678004. URL https://doi.org/10.1145/3678004.

[27] 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.

[28] William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=NjNGlPh8Wh.

[29] Jianmo Ni, Jiacheng Li, and Julian J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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, Hong Kong, China, November 3-7, 2019*, pages 188–197. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1018. URL https://doi.org/10.18653/v1/D19-1018.

[30] 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.

[31] 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.

[32] 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 Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/20dcab0f14046a5c6b02b61da9f13229-Abstract-Conference.html.

[33] 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, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=din0lGfZFd.

[34] 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.

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

[36] 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 Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 677–693. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.36. URL https://doi.org/10.18653/v1/2025.emnlp-main.36.

[37] 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.

[38] 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, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

[39] 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 Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/su25g.html.

[40] 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 Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu, editors, *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 1441–1450. ACM, 2019. doi: 10.1145/3357384.3357895. URL https://doi.org/10.1145/3357384.3357895.

[41] 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.

[42] 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 Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Findings of ACL, pages 13176–13188. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.780. URL https://doi.org/10.18653/v1/2024.findings-acl.780.

[43] 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, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

[44] 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 Ricardo Baeza-Yates and Francesco Bonchi, editors, *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pages 3245–3254. ACM, 2024. doi: 10.1145/3637528.3671775. URL https://doi.org/10.1145/3637528.3671775.

[45] 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 Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

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

[47] 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, 2025. doi: 10.48550/ARXIV.2501.09686. URL https://doi.org/10.48550/arXiv.2501.09686.

[48] Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 23336–23351. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.1137/.

[49] 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.

[50] Kun Yang, Yuxuan Zhu, Yazhe Chen, Siyao Zheng, Bangyang Hong, Kangle Wu, Yabo Ni, Anxiang Zeng, Cong Fu, and Hui Li. Mancar: Manifold-constrained latent reasoning with adaptive test-time computation for sequential recommendation. *CoRR*, abs/2602.20093, 2026. doi: 10.48550/ARXIV.2602.20093. URL https://doi.org/10.48550/arXiv.2602.20093.

[51] 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 Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

[52] 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.

[53] 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.

[54] 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), 2025. doi: 10.36227/techrxiv.176287939.92578520/v2. URL https://www.techrxiv.org/doi/abs/10.36227/techrxiv.176287939.92578520/v2.

[55] 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, 2025. doi: 10.48550/ARXIV.2504.09627. URL https://doi.org/10.48550/arXiv.2504.09627.

[56] 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, 2025. doi: 10.48550/ARXIV.2505.19092. URL https://doi.org/10.48550/arXiv.2505.19092.

[57] 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, Utrecht, The Netherlands, May 13-16, 2024*, pages 1435–1448. IEEE, 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. [45] first proposed Chain-of-Thought (CoT) prompting, significantly improving complex task performance by explicitly generating intermediate reasoning steps. Subsequently, Kojima et al. [22] discovered that simply prompting "Let's think step by step" elicits zero-shot reasoning, while subsequent studies [43, 51, 2, 10, 28] 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 [6] and OpenAI O1 [30] have pushed explicit reasoning to its limits through RL, and Snell et al. [38] 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. [16]'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. [7, 8], which gradually internalize explicit reasoning into latent reasoning via knowledge distillation. Contemporaneously, Goyal et al. [14] and Pfau et al. [31] 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 [16]. Cheng and Van Durme [4] and Yu et al. [53] also explored different paths of compressing explicit reasoning into latent space.

Along the Coconut line, follow-up work includes: Looped Transformers [33] proving that the same set of parameters recurrently executed multiple times is equivalent to increasing model depth without adding parameters; Huginn [12] adopting a prelude-core-coda architecture to realize deep recurrent latent reasoning, achieving performance comparable to larger models through test-time compute scaling; CODI [36] compressing explicit CoT into continuous space via self-distillation, matching CoT-SFT performance in a single training step. In diverse explorations, Heima [35] compresses each CoT step into a single thinking token; SoftCoT [48] generates soft thinking tokens through a frozen LLM plus trainable projection module to avoid catastrophic forgetting; LaTRO [3] optimizes latent reasoning through variational inference and self-rewarding mechanisms. In adaptive reasoning, Think-at-Hard [11] selectively triggers latent iterations based on token difficulty, and Token-Assorted [39] 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 [21], BERT4Rec [40], and GRU4Rec [17] use self-attention or bidirectional encoders to extract preference representations from user interaction sequences, computing item scores via dot-product for ranking, a typical discriminative paradigm.

With the rise of LLMs, a series of works have reformulated recommendation as a sequence generation problem. Geng et al.'s P5 [13] and M6-Rec [5] pioneered a unified recommendation pretraining paradigm, unifying diverse recommendation signals into a text generation format. TALLRec [1] further explored parameter-efficient alignment of LLMs with recommendation. In the generative retrieval direction, Rajput et al.'s TIGER [32] groundbreakingly modeled recommendation as token-

by-token generation of Semantic IDs, and subsequent work [19] systematically studied semantic indexing methods for item IDs. Building on these foundations, LC-Rec [57] integrated collaborative semantic information into LLMs, training them via SFT to directly generate recommended item IDs. Other studies [24, 44] 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 [23] 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 [18] proposed applying CoT explicit reasoning to generative recommendation, but the high generation latency of explicit CoT constrains its practical deployment efficiency. Other studies [42, 9, 25, 52] 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 [41] realizes latent reasoning within a SASRec-style architecture, trained with temperature annealing and reasoning-aware contrastive learning, but uses globally fixed step counts. LARES [27] 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 [55] introduces stepwise reasoning in sequential recommendation from the "slow thinking" perspective. ManCAR [50] explicitly identifies the representation drift problem in latent reasoning, proposing a collaborative manifold built from interaction graphs to constrain reasoning trajectories. LCR-SER [37] 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 [56] 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 [15] 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 work to realize complete latent reasoning with recurrent hidden-state feedback and adaptive step control 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 [26, 47, 54] indicate that latent reasoning research in recommender systems is still in its early stages. LASAR pioneers the organic integration of latent recurrent reasoning, LLM generative recommendation, and adaptive computation.

B Dataset Statistics

Table 6 summarizes the statistics of the three Amazon review datasets used in our experiments.

Table 6: Statistics of the three Amazon review datasets used in experiments.

Dataset	#Users	#Items	#Interactions	Sparsity	#Test Samples
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

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:

$$\text{HR@}K = \begin{cases} 1 & \text{if ground-truth item} \in \text{top-}K \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

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):

$$\text{NDCG@}K = \begin{cases} \frac{1}{\log_2(r+1)} & \text{if } r \leq K \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where the ideal DCG (IDCG) is 1 since there is exactly one relevant item, making the normalization trivial.

We report both metrics at $K \in \{5, 10, 20\}$. Results are averaged over all test users.

D Baseline Details

Traditional sequential models. SASRec [21] applies the self-attention mechanism to sequential recommendation, adaptively weighting historical items to capture long-term user preferences. GRU4Rec [17] 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 [57] 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 [23] is a compact generative recommender that directly generates item IDs autoregressively without additional reasoning. It shares the same base model and prompt format as LASAR, ensuring that any performance difference reflects the reasoning mechanism rather than model capacity or prompt design.

Latent reasoning. ReaRec [41] 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 [18] 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.

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 num_generations (G)	8
RL β (KL penalty)	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_kl_weight)	1×10^{-5}
Teacher CoT model	GPT-5
Semantic segmentation	BAAI/bge-small-en-v1.5
Beam search width	50
GPU	8× L40 (48GB)

E.1 Variable-Length N Batch Processing

The Policy Head predicts different N for each sample, resulting in varying numbers of latent tokens per sample within a batch. LASAR handles this through a unified padding-and-masking strategy (Figure 6) 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 6: 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 $\text{CoT}_{\text{GREAM}}$) share an identical prompt template following MiniOneRec [23], ensuring that 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 `<SID_*>` are $M{=}4$-token semantic IDs. Items within the history are represented by their SID tokens (e.g., `<|a_125|><|b_109|><|c_135|><|d_125|>`), not by natural language titles.

Teacher CoT Structure. The teacher CoT reasoning text is generated by GPT-5 following the structured reasoning pipeline of GREAM [18], which defines a 5-stage logical chain: *behavioral evidence extraction* $\rightarrow$ *latent preference modeling* $\rightarrow$ *intent inference* $\rightarrow$ *recommendation formulation* $\rightarrow$ *denoised sequence rewriting*. The CoT text is segmented via BAAI/bge-small-en-v1.5 [46] to determine the reasoning step count N per sample, which serves as the Policy Head's supervision label during SFT.

CoT Case Study. We show a representative teacher CoT example from the Beauty dataset. Given a user's purchase history of skincare products (hydrating serum, facial oils, eye gel, anti-aging moisturizer), GPT-5 produces the following structured reasoning:

Step 1: Extraction of Behavioral Evidence
Product types: Hydrating serum (hyaluronic acid), facial oils (rosehip, argan, emu), eye gel, anti-aging moisturizer. Key concerns: Dryness, fine lines, wrinkles, dark circles, puffiness, uneven tone; emphasis on anti-aging and deep hydration. Ingredient preferences: Natural/organic, cold-pressed, unrefined oils; hyaluronic acid; vitamin E; gentle, non-greasy formulas.

Step 2: Modeling of Latent Preferences
Natural, clean, and gentle formulations for sensitive/dry skin. Multi-benefit, anti-aging products that hydrate, brighten, and improve texture. Lightweight, fast-absorbing, non-greasy textures. Value in larger sizes or long-lasting products.

Step 3: Inference of User Intent and Goal Formulation
The user is building a comprehensive anti-aging, hydrating routine. With serum, moisturizer, eye gel, and oils in place, the next logical addition is a gentle, natural cleanser to prep skin without stripping moisture, ideally with soothing botanicals and hyaluronic acid.

Recommendation: `<|a_125|><|b_109|><|c_135|><|d_125|>`

In LASAR, this explicit text is *never decoded* at inference. Instead, the model aligns its latent hidden states with the semantic segments of the teacher CoT via bidirectional KL divergence during SFT, and the Policy Head learns to allocate reasoning depth proportional to the number of CoT segments.

E.3 SFT Phase Alignment Implementation

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**: Filter out samples where N exceeds the number of CoT segments to prevent out-of-bounds access.
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

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:

$$r_{\text{rule}}^{(i)} = \begin{cases} 1 & \text{if } \hat{y}^{(i)} = y^* \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where i indexes the i-th candidate in the beam-searched group (i.e., the ranking position), $\hat{y}^{(i)}$ is its generated 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 $w_i = -1/\log_2(i+2)$ be the position weight. If the ground-truth item appears among the G candidates, non-target candidates receive:

$$r_{\text{NDCG}}^{(i)} = \begin{cases} 0 & \text{if } \hat{y}^{(i)} = y^* \\ \frac{w_i}{\sum_{j=1}^{G} w_j} & \text{if } \hat{y}^{(i)} \neq y^* \end{cases} \tag{10}$$

If the ground-truth item is absent from all G candidates, every candidate receives $r_{\text{NDCG}}^{(i)} = 0$.

E.6 Inference Algorithm

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 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 ▷ Segment 3
return Top-K recommended items

E.6.1 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 invalid SID sequences that have no item in the catalog.

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 7 illustrates this process with a simplified example where M=3 and each codebook has 4 tokens.

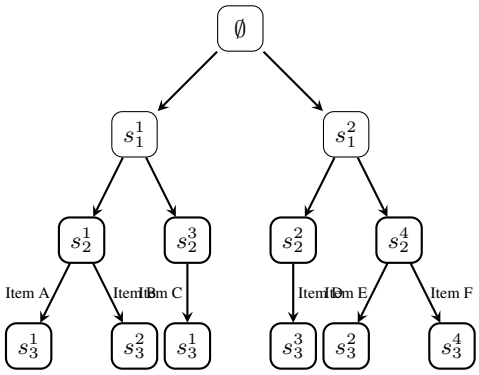


Figure 7: Trie over item SIDs (M=3, $|\mathcal{C}^{(j)}|$=4). Each root-to-leaf path is a valid item. At step j, only children of the current prefix are allowed, preventing invalid sequences.

F 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, indicating more complex user patterns. The Policy Head thus learns to associate input features with appropriate reasoning depth during the initial encoding stage of SFT.

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		↑	↑

G Alignment Ablation on Sports and Instruments

Table 9 extends the alignment ablation (Table 2) to Sports and Instruments. KL alignment consistently yields the best results across both datasets.

KL alignment consistently yields the best results across both datasets, improving NDCG@10 over Pure SFT. The unaligned variant shows inconsistent behavior—slight gains on some Sports metrics but notable degradation on Instruments (−9.1% NDCG@10). This confirms that KL-based alignment is not dataset-specific: it provides stable improvement, while unaligned latent reasoning does not reliably benefit recommendation.

Table 9: Alignment ablation on Sports and Instruments. 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**