# Do Recommenders Really Think? Stepwise Reasoning for Generative Recommendation

**Anonymous ACL submission**


## Abstract

Generative recommendation predicts the next item by autoregressively decoding its hierarchical Semantic ID. Existing reasoning-enhanced approaches prepend natural-language rationales to this process, but their reasoning signals are weakly grounded in the token-level decisions that actually specify the target item. We characterize this limitation as a Reasoning–Action Gap and propose **StepRec**, a stepwise reasoning framework that aligns local reasoning with coarse-to-fine Semantic ID generation. By interleaving reasoning segments and ID-token predictions, StepRec turns hierarchical recommendation into a sequence of explicitly guided semantic decisions. We further develop SID-Constrained Search to ensure valid identifier decoding and Decision-Token Weighted Optimization to strengthen the learning of key decision tokens. Experiments on three benchmarks show that StepRec consistently outperforms representative sequential, generative, and LLM-based baselines. The code is available at `https://anonymous.4open.science/r/StepRec-54DE`.


## 1 Introduction

Sequential recommendation aims to model users' dynamic preferences from their historical interactions and predict items they may interact with in the future. It has been widely applied in real-world scenarios such as e-commerce (Tang et al., 2025b) and streaming platforms (Qu et al., 2025). In recent years, end-to-end generative recommendation has emerged as an important paradigm in recommender systems (Hou et al., 2025; Lin et al., 2025). These methods typically assign each item a Semantic ID (SID), which represents item semantics as a hierarchical token sequence from coarse-grained to fine-grained levels (Zhan et al., 2026). Based on such representations, the model autoregressively generates the SID of the target item and ranks candidate items accordingly.

Inspired by the reasoning ability of large language models, recent studies have introduced natural-language rationales into generative recommendation (GR) (Liu et al., 2025b; Lin et al., 2026). A common strategy is to first generate a global reasoning trajectory that summarizes the user's interests, and then generate the target SID conditioned on this reasoning context. However, this raises a critical question: does such reasoning actually participate in the token-level decisions that determine the final recommendation?

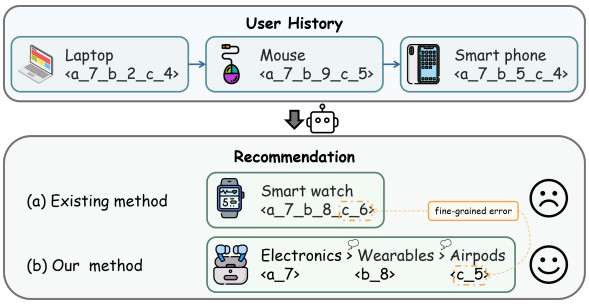


Figure 1: Motivation of our work. Existing methods rely on global reasoning before SID generation, which may lead to fine-grained token bias. Our method aligns reasoning with the hierarchical generation of SID tokens, enabling staged and decision-aware recommendation.

In this paper, we refer to this mismatch as the **Reasoning–Action Gap**. As shown in Fig. 1, existing reasoning-enhanced generative recommendation methods usually follow a "reason-then-recommend" paradigm, where reasoning is generated once before SID prediction. As a result, the reasoning process mainly provides coarse global context and is weakly aligned with the hierarchical token decisions in SID generation. Moreover, the final recommended item is determined by a small number of SID tokens, while natural-language reasoning tokens may dominate the generation process without directly improving these critical decisions. This can lead the model to produce plausible reasoning text but still make inaccurate semantic-token predictions.

To address this issue, we propose Stepwise Reasoning for Generative Recommendation (StepRec), a reasoning framework tailored to the hierarchical generation process of SIDs. StepRec enables targeted recommendation by transforming reasoning into staged decision signals that directly act on token generation. We further introduce two strategies to better align training and inference with SID decisions. First, we design **SID-Constrained Search** (SCS), which constrains the generation paths and guides the model to focus on a more effective candidate decision space. We also introduce **Decision-Token Weighted Optimization** (DWO), which increases the importance of key tokens in the training objective, making the reasoning process more faithful to the actual recommendation decision.

The main contributions of this paper are summarized as follows:

- We identify the phenomenon of **Reasoning-Action Gap** in reasoning-enhanced generative recommendation.
- We propose **StepRec**, a stepwise reasoning framework that interleaves local reasoning with hierarchical SID generation, enabling reasoning to support coarse-to-fine recommendation decisions.
- We introduce SCS and DWO to align inference and training with key SID tokens, and validate their effectiveness on three Amazon recommendation benchmarks.

## 2 Methodology

### 2.1 Overall Framework

Given a user historical interaction sequence $\mathcal{H}_u = \{v_1, \ldots, v_n\}$, generative recommendation aims to predict the semantic identifier of the next target item $v_{n+1}$. Following existing generative recommendation methods, each item is represented as a hierarchical semantic ID:

$$\mathrm{SID}(v_{n+1}) = \langle s_1, s_2, \ldots, s_k \rangle,$$

where $k$ denotes the number of semantic levels. In our setting, we use a four-level SID structure constructed by RQ-KMeans. Its coarse-to-fine structure is semantically related to, but not directly assigned from, the item category hierarchy. More details are provided in Appendix C.

Existing reasoning-enhanced generative recommendation methods usually first generate a global natural-language rationale and then generate the complete semantic ID:

$$\mathcal{H}_u \rightarrow r \rightarrow \langle s_1, s_2, \ldots, s_k \rangle.$$

Although this paradigm introduces explicit reasoning, the reasoning process mainly serves as a global prefix before generation and is weakly coupled with the token-level decision process of semantic ID generation. As a result, the generated rationale may not be effectively transformed into useful decision signals for recommendation.

To address this limitation, we propose **StepRec**, which reformulates semantic ID generation as a sequence of reasoning-decision steps.

$$\mathcal{H}_u \rightarrow r_1 \rightarrow s_1 \rightarrow r_2 \rightarrow s_2 \rightarrow \cdots \rightarrow r_k \rightarrow s_k.$$

Here, $r_k$ denotes the local reasoning segment serving the $k$-th semantic token decision. Our reasoning path is constructed based on an empirical observation that the hierarchical semantic structure in item metadata exhibits a strong alignment with the hierarchical embedding structure of semantic IDs. More detailed analyses are provided in Appendix B.5. In this way, reasoning is no longer merely a global explanation before generation, but progressively participates in the hierarchical decision process of semantic ID generation.

At the $k$-th stage, the model first generates a local reasoning segment conditioned on the user history, previous reasoning segments, and the already generated semantic ID prefix:

$$r_k \sim P(r_k \mid \mathcal{H}_u, r_{<k}, s_{<k}).$$

Then, the model predicts the current semantic ID token based on the local reasoning segment:

$$s_k \sim P(s_k \mid \mathcal{H}_u, r_{\leq k}, s_{<k}).$$

Therefore, the complete output sequence is formulated as:

$$Y = [r_1, s_1, r_2, s_2, r_3, s_3, r_4, s_4].$$

This design aligns the granularity of reasoning with the granularity of semantic ID generation, enabling each reasoning segment to directly support the corresponding semantic decision.

### 2.2 SID-Constrained Search and Decision-Token Weighted Optimization

During inference, directly applying beam search over the entire output sequence may cause the

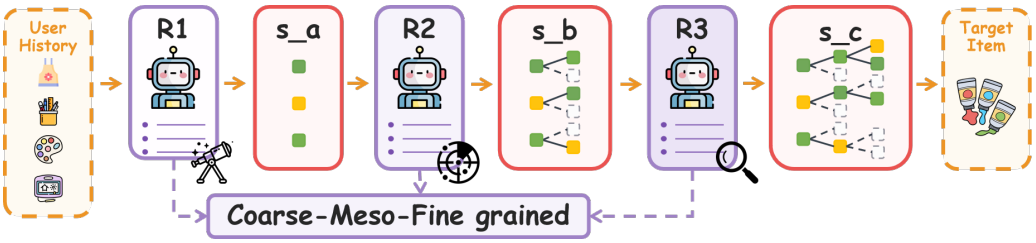


Figure 2: Conceptual overview of StepRec, where reasoning is interleaved with hierarchical SID generation.

search process to be dominated by natural-language reasoning tokens, whose surface variations do not necessarily lead to better recommendation performance. Therefore, we propose **SID-Constrained Search**, which restricts beam search to semantic ID token positions, while generating reasoning segments autoregressively.

Specifically, when generating the $k$-th semantic token, we construct a legal candidate set $\mathcal{V}_k(s_{<k})$ according to the prefix structure of semantic IDs. The next semantic token is selected only from this legal candidate set:

$$s_k^* = \arg \max_{s_k \in \mathcal{V}_k(s_{<k})} P(s_k \mid \mathcal{H}_u, r_{\leq k}, s_{<k}).$$

This constraint ensures that the generated semantic ID corresponds to a valid item, while concentrating the search budget on the key tokens that directly determine the recommendation result. The final ranking score of a candidate item is computed only using the probabilities of semantic ID tokens:

$$\text{Score}(v) = \sum_{k=1}^{4} \log P(s_k \mid \mathcal{H}_u, r_{\leq k}, s_{<k}).$$

We exclude the probabilities of reasoning tokens from the final ranking score, as they are sensitive to length and surface form variations and do not reliably reflect recommendation confidence.

During training, the model learns the interleaved output sequence $Y$ with the standard next-token prediction objective. However, reasoning tokens and semantic ID tokens play different roles in recommendation: reasoning tokens provide intermediate decision contexts, whereas semantic ID tokens directly determine the final recommended item. Treating all tokens equally may underemphasize these key decision tokens.

To address this issue, we propose **Decision-Token Weighted Optimization**, which assigns larger training weights to semantic ID tokens:

$$\mathcal{L} = -\sum_{t=1}^{|Y|} w_t \log P(y_t \mid \mathcal{H}_u, y_{<t}),$$

where

$$w_t = \begin{cases} \lambda_k, & y_t = s_k, \\ 1, & \text{otherwise}. \end{cases}$$

Equivalently, the objective can be written as:

$$\mathcal{L} = \sum_{k=1}^{4} \mathcal{L}_{r_k} + \sum_{k=1}^{4} \lambda_k \mathcal{L}_{s_k}.$$

Here, $\lambda_k > 1$ increases the supervision on the $k$-th SID token, with weights selected on the validation set according to the relative importance of different SID levels. This objective complements SID-Constrained Search by emphasizing key decision tokens during training and focusing decoding on valid hierarchical paths during inference. Further implementation details of the training pipeline, token weighting, and constrained decoding are provided in Appendix C.

## 3 Experiments

### 3.1 Experimental Setup

**Datasets and Preprocessing.** We evaluate on three sequential recommendation benchmarks: Amazon[1] Instruments, Amazon Movie and Amazon Beauty. The statistics of these datasets are reported in Appendix B.1.

**Baselines.** The baseline methods used for comparison can be divided into three categories: (1) traditional sequential recommendation methods, including SASRec (Kang and McAuley, 2018); (2) generative recommendation methods, including TIGER (Rajput et al., 2023); (3) LLM-based recommendation methods, including TallRec (Bao et al., 2023) and OneRec-Think (Liu et al., 2025b). Detailed descriptions of these baselines are provided in Appendix B.3.

**Evaluation Protocol.** For fair comparison, all methods are evaluated under the same held-out next-item ranking protocol with the same semantic-identifier-based item representation. LLM-based

[1] https://amazon-reviews-2023.github.io/

Table 1: Overall performance comparison on Beauty, Instruments, and Movie&TV. The best results are highlighted in bold, while sub-optimal results are underlined. The *Improv* row reports the relative improvement of our method compared to the best baseline.

| Method | Beauty | | | | Instruments | | | | Movie&TV | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HR@5 | HR@10 | N@5 | N@10 | HR@5 | HR@10 | N@5 | N@10 | HR@5 | HR@10 | N@5 | N@10 |
| SASRec | 0.0325 | 0.0512 | 0.0275 | 0.0345 | 0.0596 | 0.0764 | 0.0468 | 0.0586 | 0.0418 | 0.0645 | 0.0332 | 0.0410 |
| TIGER | 0.0378 | 0.0531 | 0.0296 | 0.0346 | 0.0483 | 0.0749 | 0.0333 | 0.0418 | 0.0357 | 0.0556 | 0.0237 | 0.0301 |
| TallRec | 0.0338 | 0.0526 | 0.0235 | 0.0295 | 0.0581 | 0.0602 | 0.0328 | 0.0403 | 0.0432 | 0.0670 | 0.0302 | 0.0381 |
| OneRec-Think | 0.0465 | 0.0691 | 0.0326 | 0.0415 | 0.0716 | 0.0890 | 0.0536 | 0.0620 | 0.0628 | 0.0902 | 0.0471 | 0.0587 |
| **StepRec** | **0.0501** | **0.0750** | **0.0358** | **0.0448** | **0.0743** | **0.0932** | **0.0582** | **0.0665** | **0.0663** | **0.0955** | **0.0504** | **0.0615** |
| *Improv* | 7.74% | 8.54% | 9.82% | 7.95% | 3.71% | 4.72% | 7.25% | 5.57% | 5.57% | 5.88% | 7.01% | 4.77% |

Table 2: Ablation study on the Beauty Dataset with StepRec.

| Method | HR@5 | HR@10 | N@5 | N@10 |
|---|---|---|---|---|
| w/o SCS | 0.0479 | 0.0727 | 0.0334 | 0.0424 |
| w/o DWO | 0.0483 | 0.0735 | 0.0347 | 0.0419 |
| **StepRec** | **0.0501** | **0.0750** | **0.0358** | **0.0448** |

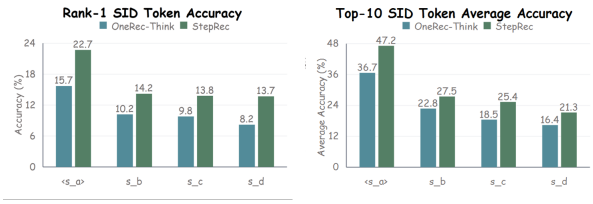


Figure 3: Prediction accuracy of hierarchical SID tokens on the beauty dataset.

baselines are implemented with the same fine-tuned Qwen3-1.7B backbone. We report standard cutoff-based metrics, including HR@5, HR@10, NDCG@5, and NDCG@10 (Sun et al., 2019). More implementation and evaluation details are provided in Appendix B.2.

### 3.2 Overall Performance

As shown in Table 1, StepRec achieves the best performance across all datasets and evaluation metrics, demonstrating its effectiveness in providing useful signals for next-item prediction. Compared with existing LLM-based recommendation methods, StepRec explicitly aligns the reasoning process with the hierarchical generation of semantic identifiers, enabling reasoning signals to be more directly transformed into effective ranking decisions. This design leads to a clear performance advantage. A detailed case study is presented in Appendix B.4.

### 3.3 Ablation Study

Table 2 presents the ablation results of StepRec on the Beauty dataset. The results show that each component is indispensable: SCS improves decoding by restricting the search to valid SID paths, while DWO strengthens the model's ability to predict hierarchical semantic identifiers. The full StepRec model achieves the best performance, indicating that the two components complement each other and jointly improve the effectiveness of sequential recommendation.

### 3.4 Prediction Accuracy Analysis

Figure 3 analyzes the prediction accuracy of StepRec at the SID-token level. The results show that StepRec achieves higher prediction accuracy than OneRec-Think across all SID tokens, indicating that it provides a more effective optimization of the SID generation process as a whole. The accuracy of $s_a$ is noticeably higher than that of subsequent tokens, suggesting that coarse-grained semantic categories are easier to infer from users' historical behaviors. Meanwhile, StepRec still maintains consistent advantages at finer-grained levels, demonstrating that its stepwise reasoning mechanism can effectively support the progressive prediction of hierarchical SID tokens.

## 4 Conclusion

This paper revisits the role of reasoning in generative recommendation and shows that existing methods often leave a gap between generated rationales and final SID decisions. To address this issue, we introduce StepRec, which formulates recommendation as a coarse-to-fine process of reasoning and semantic-token prediction. With SID-Constrained Search and Decision-Token Weighted Optimization, StepRec directs both inference and training toward the key tokens that determine the recommended item. Experiments on three benchmarks demonstrate its effectiveness in hierarchical SID prediction.

## Limitations

Our experiments are mainly conducted on preprocessed medium-scale Amazon recommendation datasets. Although this setting covers multiple product domains and ensures controlled comparison with existing methods, it does not fully reflect large-scale industrial retrieval scenarios or online recommendation environments. Future work may further evaluate the scalability of StepRec under larger candidate spaces, longer user histories, and more complex item semantic structures.

Furthermore, as with other recommender systems, improving recommendation accuracy may amplify biases in historical user–item interactions or reinforce narrow user preferences if deployed without appropriate monitoring. Since our study is based on public offline benchmarks and does not introduce new user data collection or deployment-specific profiling, we do not identify additional risks beyond those generally associated with recommender systems.

## References


Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM conference on recommender systems*, pages 1007–1014.

Zihao Guo, Jian Wang, Ruxin Zhou, Youhua Liu, Jiawei Guo, Jun Zhao, Xiaoxiao Xu, Yongqi Liu, and Kaiqiao Zhan. 2026. $S^2$gr: Stepwise semantic-guided reasoning in latent space for generative recommendation. *arXiv preprint arXiv:2601.18664*.

Yupeng Hou, Jiacheng Li, Ashley Shin, Jinsung Jeon, Abhishek Santhanam, Wei Shao, Kaveh Hassani, Ning Yao, and Julian McAuley. 2025. Generating long semantic ids in parallel for recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 956–966.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.

Xinyu Lin, Pengyuan Liu, Wenjie Wang, Yicheng Hu, Chen Xu, Fuli Feng, Qifan Wang, and Tat-Seng Chua. 2026. Bringing reasoning to generative recommendation through the lens of cascaded ranking. In *Proceedings of the ACM Web Conference 2026*, pages 8939–8949.

Xinyu Lin, Haihan Shi, Wenjie Wang, Fuli Feng, Qifan Wang, See-Kiong Ng, and Tat-Seng Chua. 2025. Order-agnostic identifier for large language model-based generative recommendation. In *Proceedings of the 48th international ACM SIGIR conference on research and development in information retrieval*, pages 1923–1933.

Enze Liu, Bowen Zheng, Xiaolei Wang, Wayne Xin Zhao, Jinpeng Wang, Sheng Chen, and Ji-Rong Wen. 2025a. Lares: Latent reasoning for sequential recommendation. *arXiv preprint arXiv:2505.16865*.

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

Yunke Qu, Liang Qu, Tong Chen, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2025. Efficient multimodal streaming recommendation via expandable side mixture-of-experts. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 2460–2470.

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

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450.

Jiakai Tang, Xu Chen, Wen Chen, Jian Wu, Yuning Jiang, and Bo Zheng. 2026. Parallel latent reasoning for sequential recommendation. *arXiv preprint arXiv:2601.03153*.

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

Jiakai Tang, Chuan Wang, Gaoming Yang, Han Wu, Jiahao Yu, Jian Wu, Jianwu Hu, Junjun Zheng, Longbin Li, Shuwen Xiao, and 1 others. 2025b. Reaseq: Unleashing world knowledge via reasoning for sequential modeling. *arXiv preprint arXiv:2512.21257*.

Haibo Xing, Hao Deng, Yucheng Mao, Jinxin Hu, Yi Xu, Hao Zhang, Jiahao Wang, Shizhun Wang, Yu Zhang, Xiaoyi Zeng, and Jing Zhang. 2025. Reg4rec: Reasoning-enhanced generative model for large-scale recommendation systems. *arXiv preprint arXiv:2508.15308*.

Tianyu Zhan, Kairui Fu, Zheqi Lv, and Shengyu Zhang. 2026. Rastp: Representation-aware semantic token

pruning for generative recommendation with semantic identifiers. In *Proceedings of the ACM Web Conference 2026*, pages 8309–8312.

# A Related Work

## A.1 Sequential and Generative Recommendation

Sequential recommendation focuses on capturing users' evolving preferences from their interaction histories and modeling temporal dependency patterns to predict future item interactions. Representative methods such as SASRec (Kang and McAuley, 2018) and BERT4Rec (Sun et al., 2019) capture sequential dependencies with self-attention, while recent studies further incorporate multimodal signals, streaming adaptation, and world knowledge to improve user representation under sparse and evolving preferences (Qu et al., 2025; Tang et al., 2025b). Despite their effectiveness, these methods usually follow a discriminative scoring paradigm, which is less aligned with language-model-style recommendation.

Generative recommendation formulates next-item prediction as directly generating the target item's identifier. TIGER (Rajput et al., 2023) is a representative semantic-ID-based framework that constructs hierarchical Semantic IDs through residual quantization and autoregressively generates the target SID. Follow-up studies improve Semantic ID generation from different perspectives, such as parallel long-ID generation (Hou et al., 2025), order-agnostic identifiers (Lin et al., 2025), and representation-aware token pruning (Zhan et al., 2026). However, most existing methods still directly generate the complete SID from user history, leaving the intermediate decision process behind hierarchical SID generation insufficiently modeled.

## A.2 Reasoning-Enhanced Generative Recommendation

The reasoning ability of large language models has motivated recent efforts to enhance recommendation with explicit or implicit reasoning. Explicit approaches first verbalize user interests or item-matching rationales and then produce recommendations. TallRec (Bao et al., 2023) aligns LLMs with recommendation tasks through instruction tuning, OneRec-Think (Liu et al., 2025b) activates in-text reasoning for generative recommendation, and REG4Rec (Xing et al., 2025) explores multiple reasoning paths with self-reflection. Although these methods enhance model performance, the generated rationales serve as a global prefix positioned prior to the item or semantic identifier prediction stage. Consequently, this leads to a weak semantic correlation with the crucial semantic identifier tokens that dictate the final recommendation outcomes.

Implicit or latent reasoning methods instead perform additional computation in representation space. ReaRec (Tang et al., 2025a), LARES (Liu et al., 2025a), and PLR (Tang et al., 2026) explore multi-step or parallel latent reasoning for sequential recommendation. For generative recommendation, CARE (Lin et al., 2026) studies SID generation through cascaded ranking, and $S^2$GR (Guo et al., 2026) inserts semantically supervised thinking tokens before SID generation steps. Different from prior reason-then-recommend methods, StepRec reformulates SID prediction as interleaved reasoning and action, where each local reasoning segment directly supports the corresponding coarse-to-fine semantic decision.

Table 3: Statistics of the processed Amazon datasets.

| Dataset | #Users | #Items | #Interactions |
|---|---|---|---|
| Beauty | 22,363 | 12,101 | 198,360 |
| Instruments | 24,733 | 9,923 | 206,153 |
| Movies&TV | 20,000 | 16,203 | 157,346 |

# B Detailed Discussion and Analysis

## B.1 Dataset Statistics

Importantly, the statistics in Table 3 represent the preprocessed datasets used in our experiments rather than the raw Amazon corpora. Specifically, we filter items based on valid metadata titles and minimum interaction frequencies, subsequently reconstructing user sequences and retaining only valid users. Thus, the reported counts reflect the data post-filtering. To ensure fair comparison, all methods are evaluated on identical data splits using a standard next-item re-ranking protocol, where each held-out target item is ranked alongside a fixed candidate set (Kang and McAuley, 2018).

## B.2 Implementation Details

During decoding, the search space is restricted to valid SID paths corresponding to items in the fixed candidate set. Beam search is applied only to SID-level candidates, and each item is ranked

by accumulating the generation log-probabilities of its SID subtokens, while the natural-language reasoning text is not directly included in the ranking score. StepRec is implemented by LoRA fine-tuning Qwen3-1.7B[2], with rank $r = 16$, scaling factor $\alpha = 32$, dropout 0.05, and learning rate $1 \times 10^{-5}$. Here, $r$ controls the capacity of the low-rank adaptation matrices, while $\alpha$ determines the LoRA scaling strength, resulting in an effective ratio of $\alpha/r = 2$. We further apply level-specific loss weights to SID tokens, with $\lambda_a = 16$, $\lambda_b = 8$, $\lambda_c = 4$, and $\lambda_d = 2$, assigning larger weights to earlier semantic levels to strengthen the learning of key hierarchical decisions.

For evaluation, all methods are trained on the full training set and evaluated on the full test set under the same chronological data split. Each test instance consists of a user history and the held-out next item. We adopt a fixed-candidate ranking protocol, where the held-out target item is ranked alongside a fixed candidate set. All methods are evaluated on exactly the same test instances and candidate sets to ensure a fair comparison. For SASRec, each complete SID path is treated as the unique item identity for sequential modeling and item-level scoring.

We report HR@$K$ and NDCG@$K$ with $K = 5, 10$. HR@$K$ measures whether the held-out target item appears in the top-$K$ ranked list, while NDCG@$K$ further accounts for its ranking position. These metrics jointly reflect the model's ability to retrieve the target item from the candidate set and rank it at higher positions within a short recommendation list.

**Artifacts and Responsible Use.** We use publicly available Amazon recommendation datasets, Qwen3-1.7B, and baseline implementations under their released terms of use. These artifacts are used only for academic research, offline evaluation, comparison, and reproducibility.

We do not collect new user data. Experiments are conducted on public offline benchmarks with anonymized user–item interactions. Item titles and categories are used only for prompt construction and semantic description, without using personal identifiers or attempting user re-identification. We do not manually audit all item titles for offensive content, but the data are used solely for offline evaluation.

All results are reported under a fixed data split. Unless otherwise specified, they correspond to a single run rather than averages over multiple random seeds.

StepRec is implemented with PyTorch, Hugging Face Transformers, and PEFT/LoRA, with model settings and training hyperparameters reported in this section.

[2] https://huggingface.co/Qwen/Qwen3-1.7B

### B.3 Baselines

The baseline methods used in our experiments are divided into three categories.

(1) Traditional Sequential Recommendation Method:

- SASRec (Kang and McAuley, 2018) is a sequential recommendation model that employs the self-attention mechanism to predict the next item a user is likely to interact with.

(2) Generative Recommendation Method:

- TIGER (Rajput et al., 2023) represents items as hierarchical Semantic IDs and performs recommendation by autoregressively generating the semantic identifier of the target item.

(3) LLM-based Recommendation Methods:

- TallRec (Bao et al., 2023) adapts large language models to user–item recommendation tasks through recommendation-oriented instruction tuning.
- OneRec-Think (Liu et al., 2025b) introduces explicit textual reasoning into generative recommendation to enhance the interpretability and controllability of the recommendation.

### B.4 Case Study

As shown in Figure 4, StepRec generates the final semantic identifier through hierarchical semantic reasoning. In this case, the user's historical purchases are consistently related to musical instruments, with recent interactions focusing on instrument accessories. StepRec first identifies the coarse preference for "Musical Instruments > Instrument Accessories", then narrows the candidate space to "Guitar & Bass Accessories" by recognizing the user's electric guitar and related accessory purchases, such as a strap button, capo, and tuner. It further refines the prediction to guitar strings, a frequently replaced accessory that is highly consistent with the user's previous behaviors, and finally selects electric guitar strings as the fine-grained

## Input Prompt

**System instruction.** *You are a professional recommendation expert who needs to recommend the next possible purchase for users based on their purchase history. Please predict the most likely next product that the user will purchase based on the user's historical purchase information.*

**Purchase history.** The user has purchased several musical-instrument-related products, including an electric guitar and multiple guitar-related accessories:

<s_a_21><s_b_195><s_c_58><s_d_178> : "Squier by Fender Bullet Strat with Tremolo"
*Musical Instruments* > *Guitars* > *Electric Guitars* > *Solid Body*,
<s_a_101><s_b_6><s_c_235><s_d_49> : "Martin Guitar Leather Strap Button"
*Musical Instruments* > *Instrument Accessories* > *Guitar & Bass Accessories* > *Straps & Strap Locks*,
<s_a_114><s_b_5><s_c_176><s_d_32> : "D'Addario J65 Ukulele Strings"
*Musical Instruments* > *Instrument Accessories* > *Folk & World Instrument Accessories* > *Ukulele Accessories*,
<s_a_89><s_b_1><s_c_163><s_d_183> : "Kyser 6-String Capo"
*Musical Instruments* > *Instrument Accessories* > *Guitar & Bass Accessories* > *Capos*,
<s_a_55><s_b_1><s_c_176><s_d_45> : "D'Addario NS Micro Clip-On Tuner"
*Musical Instruments* > *Instrument Accessories* > *Guitar & Bass Accessories* > *Tuners*.

## Reasoning Process of StepRec

**Step 1: Coarse semantic decision.** The user's purchase history is consistently related to musical instruments, and recent interactions mainly involve instrument accessories. Therefore, the model first identifies the user's current preference as *Musical Instruments* > *Instrument Accessories*, generating:

<s_a_84>.

**Step 2: Subcategory refinement.** Since the user purchased an electric guitar and several guitar-related accessories, such as a strap button, capo, and tuner, the model further narrows the candidate space to *Musical Instruments* > *Instrument Accessories* > *Guitar & Bass Accessories*, generating:

<s_b_71>.

**Step 3: Accessory-type decision.** Within guitar and bass accessories, the user has already purchased several basic supporting accessories. Guitar strings are highly related to electric guitar usage and are frequently replaced consumable accessories. Moreover, the user's history contains a D'Addario string-related item, which further supports the prediction toward *Strings*, generating:

<s_c_1>.

**Step 4: Fine-grained item-level decision.** Considering that the user's main instrument is an electric guitar, the most suitable fine-grained category is *Electric Guitar Strings*. The predicted item matches both the user's instrument type and the observed accessory-purchase pattern, generating:

<s_d_128>.

## Final Prediction

**Prediction Rank 1:** <s_a_84><s_b_71><s_c_1><s_d_128>

**Target:** <s_a_84><s_b_71><s_c_1><s_d_128>

The prediction exactly matches the ground-truth target Semantic ID.

**Target item:** D'Addario EXL110 Nickel Wound Electric Guitar Strings, Regular Light, 10–46.
**Target category:** *Musical Instruments* > *Instrument Accessories* > *Guitar & Bass Accessories* > *Strings* > *Electric Guitar Strings*.

Figure 4: A case study on the Musical Instruments dataset illustrating the stepwise reasoning process of StepRec. The model first identifies the user's coarse preference for instrument accessories, then progressively narrows the candidate space to guitar and bass accessories, guitar strings, and finally electric guitar strings.

target category. This stepwise refinement process reduces semantic drift from less relevant historical items and improves the accuracy of fine-grained next-item recommendation.

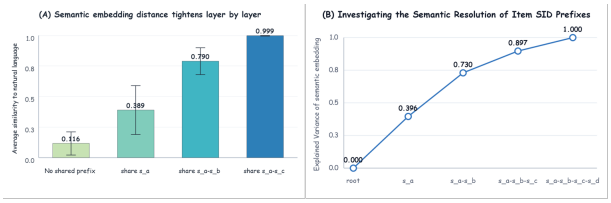


Figure 5: Semantic analysis of hierarchical SIDs with Beauty dataset. (A) Items sharing longer SID prefixes exhibit higher semantic similarity in the embedding space, showing that SID prefixes progressively narrow the semantic region. (B) The cumulative explained variance increases with longer SID prefixes, indicating a coarse-to-fine decomposition of item semantics.

### B.5 Property Analysis of Hierarchical Semantic IDs

To validate the rationality of the proposed generative reasoning path $\mathcal{H}_u \rightarrow r_1 \rightarrow s_1 \rightarrow \cdots \rightarrow r_k \rightarrow s_k$, we examine the intrinsic correspondence between discrete Hierarchical Semantic IDs (SIDs) and the continuous semantic embedding space.

As shown in Fig. 5(A), the average similarity between item representations in the natural-language semantic space increases monotonically as the shared SID prefix becomes longer. Items with no shared prefix exhibit only weak semantic similarity, whereas items sharing progressively longer prefixes become increasingly closer in the embedding space. In particular, the similarity reaches $0.999$ when items share the prefix $s_a$–$s_c$, indicating that the hierarchical SID structure effectively narrows the semantic space from coarse to fine granularity.

Fig. 5(B) further reports the cumulative explained variance of semantic embeddings with respect to different SID prefixes. As more SID tokens are introduced, the explained variance steadily increases from the root level to the full SID, eventually reaching $1.000$. Meanwhile, the marginal gain becomes smaller at deeper levels, suggesting a coarse-to-fine semantic decomposition pattern: earlier tokens mainly determine the coarse semantic region, while later tokens provide finer local distinctions.

Thus, hierarchical SIDs are well aligned with the continuous semantic embedding space. Therefore, generating SID tokens autoregressively can be interpreted as a progressive routing process over the item semantic space, where each token refines the candidate region at a finer granularity. This provides empirical support for our stepwise reasoning path, in which intermediate reasoning segments are coupled with the sequential generation of coarse-to-fine SID tokens.

## C Method Details

### C.1 Hierarchical Semantic ID Construction and Interpretation

We construct hierarchical Semantic IDs for items using RQ-KMeans. Specifically, each item is first encoded into a continuous semantic representation based on its textual metadata. RQ-KMeans then discretizes the item semantic space through layer-wise residual quantization. Each residual quantization layer produces a discrete codeword index, which is mapped to one token in the Semantic ID. As a result, each item is represented as a coarse-to-fine semantic identifier:

$$\mathrm{SID}(v) = \langle s_a, s_b, s_c, s_d \rangle.$$

In this structure, earlier SID tokens mainly capture coarse-grained semantic regions, while later SID tokens further encode residual information for fine-grained item distinction.

It is important to note that Semantic IDs are not directly assigned according to the Amazon category hierarchy. Category information is only used as semantic evidence for interpretation and reasoning, rather than determining the SID tokens themselves. Although the two hierarchies are constructed from different sources, they are naturally correlated in semantic structure: RQ-KMeans partitions the item semantic space into coarse-to-fine hierarchical regions, while product categories also organize items from broad domains to fine-grained subcategories. Therefore, StepRec uses category descriptions to support the semantic expression of local reasoning, while the final recommendation decision is still made by generating valid SID tokens constructed by RQ-KMeans.

### C.2 Three-Stage Training Pipeline

StepRec adopts a three-stage training pipeline adapted from OneRec-Think, including itemic alignment, stepwise SID recommendation training, and reasoning activation. This pipeline progressively aligns the language model with SID tokens, hierarchical recommendation decision paths, and local reasoning.

The first stage is *itemic alignment*. Since SID tokens are newly introduced special tokens for recommendation, they do not exist in the original vocabulary space of the backbone language model.

Therefore, we first expand the tokenizer and fine-tune the model with alignment data, enabling the model to recognize, represent, and generate these SID tokens.

The second stage is *stepwise SID recommendation training*. Unlike directly generating the complete SID, StepRec already adopts a stepwise output structure at this stage, allowing the model to learn a coarse-to-fine decision path from user history to the target item SID:

$$\mathcal{H}_u \rightarrow r_1 \rightarrow s_a \rightarrow r_2 \rightarrow s_b \rightarrow r_3 \rightarrow s_c \rightarrow r_4 \rightarrow s_d.$$

In our current implementation, the local rationale segments $r_i$ in the SIDRec stage are mainly template-based or fixed-style rationales, such as `<think_1>...</think_1>` to `<think_4>...</think_4>`. These scaffolds help the model become familiar with the interleaved "local reasoning–SID decision" generation format. However, the main training focus of this stage remains on the SID tokens, where the model learns to progressively narrow the candidate space along the hierarchy constructed by RQ-KMeans and generate the valid SID path corresponding to the target item.

The third stage is *reasoning activation*. This stage does not introduce a new output format from scratch. Instead, based on the stepwise structure already established in SIDRec, it further strengthens the semantic coupling between local reasoning and each SID-level decision. In other words, the RA stage encourages $r_i$ to be more than a format placeholder, making it a meaningful local reasoning segment that supports the corresponding SID decision.

### C.3 Decision-Token Weighted Optimization

Although StepRec is trained with next-token prediction over the complete output sequence, different tokens have different effects on the final recommendation. Reasoning tokens mainly provide intermediate semantic context, while SID tokens directly determine the generated item. Therefore, we assign larger training weights to SID tokens to strengthen the learning of key decision tokens.

Moreover, SID tokens at different levels have different importance. Earlier tokens determine coarser candidate subspaces; once an earlier token is incorrectly predicted, subsequent tokens will be constrained within an incorrect branch. Later tokens mainly perform fine-grained distinction within an already narrowed candidate set. Based on this observation, we adopt a decreasing weighting scheme for SID tokens:

$$\lambda_a = 16, \quad \lambda_b = 8, \quad \lambda_c = 4, \quad \lambda_d = 2.$$

In implementation, we construct a token-level mask for SID-token positions in the output sequence and multiply the corresponding cross-entropy losses by their level-specific weights. Other reasoning tokens keep the standard weight of 1.0. This design preserves supervision for local reasoning while emphasizing the SID tokens that directly determine the recommendation result.

### C.4 SID-Constrained Search (SCS)

During inference, StepRec does not apply standard beam search over the entire natural-language output sequence. Instead, it performs constrained search over valid Semantic ID paths. Specifically, we organize all item SIDs in the candidate set as a prefix tree, where each root-to-leaf path corresponds to a valid candidate item.

When generating the $k$-th SID token, the current SID prefix $s_{<k}$ determines a legal successor set $\mathcal{V}_k(s_{<k})$. The model is only allowed to select the next SID token from this set:

$$s_k \in \mathcal{V}_k(s_{<k}).$$

For example, after generating the prefix $\langle s_a, s_b \rangle$, the next token $s_c$ can only be selected from the child nodes that actually appear under this prefix in the candidate set. This prevents the model from generating invalid SID combinations that do not correspond to any candidate item.

Different from standard beam search, our beam expansion is applied only at SID decision positions. Local reasoning segments are still generated autoregressively to provide context for the current SID decision, but their token probabilities are not included in the final ranking score. For a candidate item $v$, the final score is computed only by accumulating the log probabilities of its SID tokens:

$$\mathrm{Score}(v) = \sum_{k=1}^{4} \log P(s_k \mid \mathcal{H}_u, r_{\leq k}, s_{<k}).$$

This design concentrates inference on the tokens that directly determine recommendation results and avoids ranking bias caused by the length and surface-form variation of reasoning tokens. Decision-Token Weighted Optimization strengthens SID-token learning during training, while SID-Constrained Search expands and ranks only valid

SID tokens during inference, thereby maintaining consistency between the training objective and the final recommendation decision.

## D LLM Usage

During the preparation of this work, the authors used GPT-5 to improve the readability and language quality of the manuscript and to assist in drafting minor code snippets for experimental implementation. All AI-assisted content was carefully reviewed and edited by the authors, who take full responsibility for the final published work.