LOHRec: Leveraging Order and Hierarchy in Generative Sequential Recommendation

Anonymous ACL submission

Abstract

Recently, sequential recommendation with generative retrieval has garnered significant attention. However, the training of such generative recommenders typically maximizes the prediction probability of the next item. This approach explicitly considers the accuracy of the recommendation results, but in reality, lacks awareness of other feasible items. Although leveraging large language models (LLMs) that incorporate world knowledge and introducing various auxiliary tasks can mitigate this issue, the high inference costs associated with these LLM-based recommenders make them challenging to deploy in practical scenarios. In this paper, we propose a novel learning framework, LOHRec, which exploits the order and hierarchy in generative recommendations using quantized identifiers to further explore the effectiveness ceiling of lightweight generative recommenders. Under fair comparisons with similar parameter sizes, comprehensive experiments demonstrate that generative recommenders employing our framework consistently outperform previous state-of-the-art (SOTA) models across different datasets. Additionally, we also empirically show that LOHRec can effectively align lightweight generative recommenders with LLM recommendation preferences in low-resource scenarios. Our code is available at `https://anonymous.4open.science/r/LOHRec/`.

1 Introduction

Recommender systems are increasingly popular in addressing the information overload problem on web platforms such as shopping sites, video platforms, and social media. These systems can help users discover items of interest and enhance their experience and engagement (Fayyaz et al., 2020; Ko et al., 2022). Among these recommendation paradigms, sequential recommendation (Kang and McAuley, 2018; Li et al., 2020) has recently garnered considerable attention due to its superior performance in modeling the dynamic evolving pattern of the chronological item sequence.

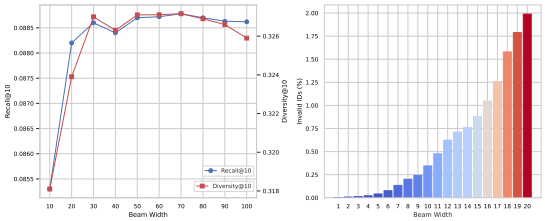

Figure 1: (Left): A demonstrated correlation between the beam width and diverse evaluation metrics on public Amazon-Games data. The Diversity@10 represents the intra-list diversity in the top 10 of the recommendation list. (Right): Percentage of invalid IDs on the test set of Amazon-Games dataset when generating item semantic IDs in a beam search manner with various beam widths.

In recent years, neural recommenders have achieved remarkable progress, such as using CNNs (Tang and Wang, 2018), RNNs (Hidasi and Karatzoglou, 2018), or attention-based models (Kang and McAuley, 2018) to capture users' evolving interests over time. With the Transformer architecture having demonstrated astonishing performance in a spectrum of domains (Vaswani et al., 2017), the Transformer-based sequential recommender is increasingly popular in research and practical application (Kang and McAuley, 2018). However, as previous works commonly use atomic and random item IDs to represent various items (HidasiB et al., 2015), the abundant information from diverse modalities is not integrated adequately to achieve sound performance. In addition, as there are a gigantic number of items in the practical application (billion or even more), traditional sequential recommendation frameworks necessitate the same size of vocabulary for the item set, which makes them hard to apply in industrial scenarios.

To address the challenges in the sequential recommendation, recent research has moved towards a generative retrieval paradigm by adopting neural quantized representations. Specifically, Rajput et al.

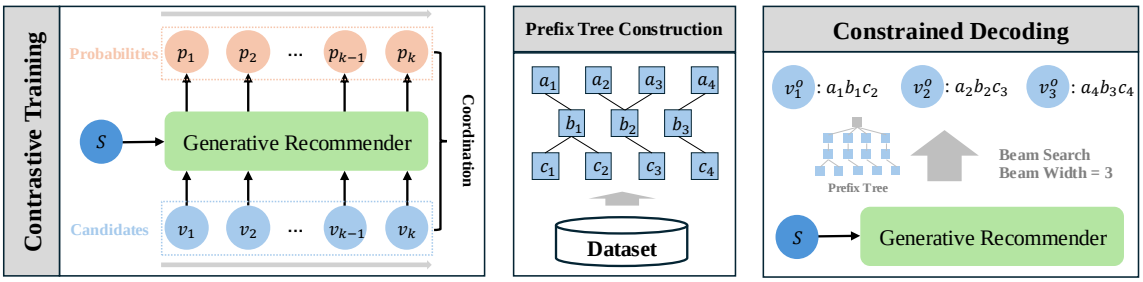

Figure 2: The overall framework of our LOHRec. We enhance generative sequential recommenders by leveraging the order information for contrastive learning and conducting constrained decoding via the hierarchy of quantized semantic item IDs.

(2024) proposes a residual quantization method that converts item embeddings into multiple codewords, rather than using low-information numerical IDs. This approach allows the number of representable items to be the product of the cardinality of each level in semantic IDs, thereby significantly alleviating the scalability issues in large item corpora. Additionally, modeling sequences of semantic item IDs facilitates the discovery of deeper correlations between items, enhancing the learning of sequence transitions and improving the performance of recommenders. Further, rather than optimizing Transformer-based models from scratch, Zheng et al. (2024) proposes leveraging the robust language comprehension capabilities of large language models (LLMs) for the sequential recommendation. They adapt LLMs for sequential recommenders through a series of alignment tasks, further boosting the effectiveness of recommenders.

While the generative recommendation paradigm has shown promising performance across various public real-world datasets, several weaknesses are blocking the availability and reliability of the method. Firstly, since the generative models are commonly trained with maximum likelihood estimation (MLE), the probability score assigned to the next target item is maximized given a user interaction sequence. In this case of Figure 1 (Left), we observe that the diversity of the top 10 recommended items tends to stabilize and even decline as the beam width increases, indicating that the recommendations generated by the generative recommender system are prone to homogenization. However, we contend that in real-world applications, there is typically a collection of acceptable candidate items in recommendation tasks. A sequential recommender that lacks diversity is therefore inadequate for delivering personalized recommendations and meeting user expectations. While LLM-based sequential recommenders may alleviate this issue by incorporating rich language semantics, their excessively high training and inference costs render them impractical for industrial use.

This work proposes LOHRec, **L**everaging the **O**rder and **H**ierarchy in the generative sequential **Rec**ommendation, to address the limitations mentioned above. To enable the generative sequential recommender to consider the ordered item correlation rather than a single target item, we introduce a margin ranking loss. This assists the model in learning the characteristics of the ordered correlation between semantic IDs. On the other hand, considering the characteristics of hierarchical semantics, we propose optimized training paradigms to enable generative recommenders to perceive more feasible items, thereby comprehensively taking into account both the accuracy and diversity of the recommendation results. To sum up, the main contributions of this paper can be presented as follows:

- To the best of our knowledge, We are the first to propose leveraging hierarchical semantics and ordered relationships of quantized item identifiers to explore the upper limits of model performance in the sequential recommendation with the generative retrieval paradigm.
- We explored a range of approaches to enable lightweight generative recommenders to efficiently align with the preference of LLM-based recommenders. Experiments demonstrate that with our method, a model with 13 million parameters can achieve performance close to that of a 7 billion parameter LLM-based recommender.
- We conduct comprehensive experiments illustrating that all variants derived from LOHRec consistently surpasses previous baselines. Furthermore, We provide ablation studies and further analyses to substantiate the superior performance of our method.

2 Preliminaries

2.1 Training Definition for Sequential Recommendation

In general, the sequential recommendation aims to predict the users' next interaction with an item according to their historical interaction sequences that are arranged chronologically. Given the user set $\mathcal{U} = \{u_1, u_2, \cdots, u_{|\mathcal{U}|}\}$ and item set $\mathcal{V} = \{v_1, v_2, \cdots, v_{|\mathcal{V}|}\}$, the interaction history of one user u can be written as $\mathcal{S}^u = [v_1^u, v_2^u, \cdots, v_{T_u}^u]$, where $T_u = |\mathcal{S}^u|$ and $v_t^u \in \mathcal{V}$ denotes the t-th interacted item in the chronological sequence. Formally, the next item prediction is defined as follows:

$$v_{T_u+1}^u = \arg\max_{v^u \in \mathcal{V}} P(v^u | \mathcal{S}^u; \theta). \tag{1}$$

2.2 RQ-VAE for Item Semantic IDs

As for the sequential recommendation, a key operation is how to represent each item in a large collection of items in a distinguishable way. A vanilla method is to associate each item with a random unique ID. However, such a scheme usually introduces too large a vocabulary of item IDs when there is a gigantic item set. Furthermore, this method is difficult to adapt to the actual scenario where a dynamic collection of constantly updated valid items is necessary.

To address these problems, some works introduce the Residual-Quantized Variational AutoEncoder (RQ-VAE), a multi-level vector quantizer that recursively quantizes the residual vectors from coarse to fine, to generate the semantic ID consisting of several tokens (i.e., codewords), where each discrete token can be shared by diverse items. **The basic idea is that similar items tend to be assigned with a portion of common semantic codewords**, such that each unique semantic ID can be aligned to latent semantics.

Following Zheng et al. (2024), to derive these semantic IDs, the first step is to encode the text information of items as text embedding. Further, the vector quantization approach is leveraged to create discrete codewords based on item embeddings. Specifically, we take the item embeddings as input and then train RQ-VAE, which consists of the residual quantizer with D-level codebooks and the DNN encoder-decoder, to generate item IDs. Given an item embedding x, RQ-VAE first encodes it into a latent representation z. During the residual vector quantization, at the d-th level (start from 0), we have a codebook $\mathcal{C}_d = \{e_n^d\}_{n=1}^{N_d}$, where N_d is the size of the d-th level codebook and each codebook vector e_n^d is learnable. Then the residual quantization process can be expressed as:

$$c_d = \arg\min_n ||r_d - e_n^d||, \tag{2}$$

$$r_{d+1} = r_d - e_{c_d}^d, \tag{3}$$

where c_d is the d-th codeword of the semantic item ID and r_d is the residual vector in the d-th level, and we set $r_0 = z$.

During the decoding stage, the quantization representation of z can be obtained according to $\hat{z} = \sum_{d=0}^{D-1} e_{c_d}^d$. Then $\hat{z}$ will be used as decoder input to reconstruct the item embedding $\hat{x}$. The overall loss function is as follows:

$$\mathcal{L}_{\text{recon}} = ||x - \hat{x}||^2, \tag{4}$$

$$\mathcal{L}_{\text{cb}} = \sum_{d=0}^{D-1} ||\text{sg}[r_d] - e_{c_d}^d||^2 + \beta ||r_d - \text{sg}[e_{c_d}^d]||^2, \tag{5}$$

$$\mathcal{L}_{\text{RQ-VAE}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{cb}}, \tag{6}$$

where $\text{sg}[\cdot]$ is the stop-gradient operator, and the straight-through estimator is used for the backpropagation through the RQ module. β is a loss coefficient, usually set to 0.25. Note that $\mathcal{L}_{\text{cb}}$ is the sum of quantization errors from every level of the residual quantizer. $\mathcal{L}_{\text{recon}}$ is the reconstruction loss, and $\mathcal{L}_{\text{cb}}$ is the residual quantization loss used to minimize the distance between codebook vectors and residual vectors.

3 Proposed Method

Specifically, this section first elucidates three training variants of generative recommendation that utilize hierarchical semantics. Subsequently, we introduce two- and one-stage methods that further enhance the recommender's performance by incorporating ordered relationships.

3.1 Leveraging Hierarchical Semantics for Generative Recommendation

Through the residual quantization operation, we can acquire the corresponding codebook-based identifier consisting of multiple tokens (i.e., multiple codewords) for each item. Accordingly, we can conduct the generative recommendation in an autoregressive manner. Mathematically, given a semantic identifier sequence $\mathcal{S} = [v^{(1)}, v^{(2)}, \cdots, v^{(|\mathcal{S}|)}]$, where the i-th semantic item identifier $v^{(i)} = (c_1^{(i)}, c_2^{(i)}, \cdots, c_D^{(i)}) \in \mathcal{V}$, we model the following

conditional probability:

$$P_\theta(v^{(i)}|\mathcal{S}_{<i}) = \prod_{j=1}^{D} p(c_j^{(i)}|v_{<j}^{(i)}, \mathcal{S}_{<i}; \theta), \tag{7}$$

where $\mathcal{S}_{<i}$ represents the sub-sequence prior to the item $v^{(i)}$, $v_{<j}^{(i)}$ contains these codewords before $c_j^{(i)}$. Especially, $v_{<1}^{(i)}$ is a special begin-of-sequence token. The training objective of generative recommendation can be transformed to minimize the following negative log-likelihood loss (NLL):

$$\mathcal{L}_{nll} = -\frac{\sum_{j=1}^{D} \log p(c_j^{(i)}|v_{<j}^{(i)}, \mathcal{S}_{<i}; \theta)}{D}, \tag{8}$$

where the model is required to maximize the conditional log-likelihood of the target item $v^{(i)}$.

Label Smoothing via Hierarchical Semantics. As the NLL loss in Equation 8 tends to assign all probabilities to the reference items, generative recommenders often only focus on items similar to them, while ignoring other items that the user might currently be interested in. To mitigate the phenomenon and regularize the model for generalization, we adjust the standard generative objective to consider the conditional probabilities of all possible codewords in the item semantic space further:

$$\mathcal{L}_{ls} = -\frac{\sum_{j=1}^{D} \sum_{k=1}^{\mathcal{V}} w_{jk}^{(i)} \log p(\hat{c}_{jk}^{(i)}|v_{<j}^{(i)}, S_{<i}; \theta)}{D}, \tag{9}$$

where $\mathcal{V}$ represents the number of all codewords instead of the whole vocabulary size to reduce the memory footprint during training. w_{jk} is the conditional weight:

$$w_{jk}^{(i)} = \begin{cases} 1 - \xi_j, & \hat{c}_{jk}^{(i)} = c_j^{(i)}; \\ \frac{\xi_j}{\mathcal{V}-1}, & \hat{c}_{jk}^{(i)} \neq c_j^{(i)}, \end{cases} \tag{10}$$

where ξ_j adjusts the probabilities assigned to codewords not in target items. While ξ_j at j-th level is independent, to reduce the complexity of our experiments, we consistently set it to 0.1.

Hierarchical Semantics from LLM-based Recommender. In this work, we propose a vocabulary-agnostic method for efficiently leveraging the hierarchical semantics of LLM-based recommenders. Specifically, unlike previous model distillation methods, we collect the probability distributions only of the additional tokens (codewords) of LLM-based recommenders by employing a teacher-forcing strategy (Bengio et al., 2015). By using these additional tokens as intermediaries, we can learn the preferences from the LLM-based recommender even if our model's initial vocabulary is incompatible. **Considering that KL divergence and cross-entropy are equivalent in training in this context**, following Equation 9, we can simplify the training objective to:

$$\mathcal{L}_{dis} = -\frac{\sum_{j=1}^{D} \sum_{k=1}^{\mathcal{V}} q_{jk}^{(i)} \log p(\hat{c}_{jk}^{(i)}|v_{<j}^{(i)}, S_{<i}; \theta)}{D}, \tag{11}$$

$$q_{jk}^{(i)} = \frac{p(\hat{c}_{jk}^{(i)}|v_{<j}^{(i)}, S_{<i}; \theta_{\text{LLM}})}{\sum_{k=1}^{\mathcal{V}} p(\hat{c}_{jk}^{(i)}|v_{<j}^{(i)}, S_{<i}; \theta_{\text{LLM}})}. \tag{12}$$

3.2 Coordinating Generative Recommendation

To improve recommenders beyond the standard generative recommendation paradigm, we adopt a learning-to-rank objective that can optimize the recommendation performance on the acceptable ordered candidates instead of a single target item. Specifically, given an input interaction sequence $\mathcal{S}$ and the corresponding ordered collection of candidate semantic item IDs $\{v_i^c\}_{i=1}^k$, where v_p^c is superior to v_q^c, $\forall p < q$. To perceive the ordered correlation among the item collection, we introduce a modification to Equation 7 in the reference-free scenario. To be specific, given the generative sequential recommender θ, the model-predicted probabilities $\hat{\mathcal{M}}_\theta$ of the target semantic ID v_i^c with respect to the input sequence $\mathcal{S}$ can be defined as follows:

$$\hat{\mathcal{M}}_\theta(v_i^c, \mathcal{S}) = \frac{\log P_\theta(v_i^c|\mathcal{S})}{|v_i^c|^\eta}, \tag{13}$$

where $|v_i^c|$ is the number of codewords for the item v_i^c and the hyper-parameter η controls the degree of length penalty. Especially, as we use the equal-length semantic IDs in our experiments, η is set to 1.0 uniformly. Accordingly, we note that the model-predicted probability ranges from $-\infty$ to 0. Finally, we formulate the ranking objective:

$$\mathcal{L}_{rank} = \sum_{i=1}^{k} \sum_{j=i+1}^{k} \max\{\hat{\mathcal{M}}_\theta(v_j^c, \mathcal{S}) - \hat{\mathcal{M}}_\theta(v_i^c, \mathcal{S}) + (j-i) \times \lambda, 0\} \tag{14}$$

where λ is the threshold judging whether the difference of model-predicted probabilities of diverse semantic IDs engages in backpropagation.

Constructing Order Prior. In generative recommendation, a critical factor in leveraging benefits

from the aforementioned ranking paradigms lies in constructing a well-ordered set of candidate items. In this work, we demonstrate the effectiveness of building ordered prior information based on recommendations from LLM-based recommenders. Additionally, for comparative experiments as baselines, we also explore the effectiveness of two alternative schemes for constructing ordered candidate sets (See Appendix B).

To construct prior information implying LLM preference, we employ a representative LLM-based recommender with the beam search strategy during the inference phase to generate a candidate semantic ID collection. As the generation process uses cumulative probability for sampling, if the beam width is greater than 1, these generated candidates can effectively align with the preference of the LLM-based recommender, providing more information relative to a single target item:

$$\begin{aligned} \mathbf{C}_1 &= \{v_i\}_{i=1}^{b} = \text{Beam\_Search}(b, \mathcal{S}, \theta_{\text{LLM}}), \\ \mathbf{C}_2 &= \{v_i^c\}_{i=1}^{k} = \text{Correct}(\mathbf{C}_1), \end{aligned} \tag{15}$$

where b represents the preset beam width, and $\mathcal{S}$ is the user's historical interaction sequence. Note that the results recommended by the LLM are often not optimal. We have implemented an additional correction process to further improve them. In this "Correct" process, we perform operations such as removing invalid semantic IDs, placing the reference at the forefront, removing duplicates, and selecting the top k items, to design a more reasonable ordered list.

Two-stage Generative Recommendation. Our two-stage recommendation process comprises a retrieval stage and a ranking stage, employing two lightweight generative models as the retriever and the ranker, respectively. Specifically, for the retriever, we investigate the training objectives in Equation 9 and 11 to enable the model to acquire generative retrieval capabilities. For the ranker, we use Equation 14 to train the model to learn from the ordered results generated by the LLM, thereby aligning with the LLM's preferences. Assume that the retriever generates b candidate items using beam search, which are then fed to the ranker, which selects the top k items based on probability scores as the final recommendation ($b \gg k$).

One-stage Recommendation with Ranking. Since the multiple results generated by the autoregressive model using the beam search strategy already exhibit an ordered relationship based on probability scores, the ranking training described in Equation 14 can be naturally integrated into the generative recommendation training for the generative recommendation with ranking. Therefore, we combine the generative loss (Equation 8, 9, or 11) and the ranking loss (Equation 14) into a universal loss function:

$$\mathcal{L}_{all} = \mathcal{L}_{gen} + \gamma \mathcal{L}_{rank}, \tag{16}$$

where γ is the weight of the ranking loss and is set to 1.0 in our experiments.

4 Experiments

4.1 Datasets and Baselines

We evaluate the proposed approach using three subsets of Amazon review data: "Musical Instruments (Instruments)", "Arts, Crafts and Sewing (Arts)", and "Video Games (Games)". These datasets encompass user reviews from May 1996 to October 2018, with each item having a title and description. Consistent with previous research, we exclude users and items with fewer than five interactions. Subsequently, we generate user behavior sequences in chronological order, setting the maximum item sequence length uniformly to 20 to align with all baseline requirements. The statistics of the preprocessed datasets are shown in Table 3.

We employ several representative recommendation models as baselines for comparison, with details provided in the Appendix C.2. Among them, FDSA (Zhang et al., 2019), S^3-Rec (Zhou et al., 2020), P5-CID (Geng et al., 2022; Hua et al., 2023), and TIGER (Rajput et al., 2024) are some of the strong baselines that have demonstrated state-of-the-art results in previous studies.

4.2 Implementation Details

Semantic Quantization. For the generation of semantic item IDs, following the setting of (Zheng et al., 2024), we first use the pre-trained LLaMA model to encode textual title and description of items in the dataset as its embeddings of 4096 dimension and use mean pooling to aggregate multiple token hidden representations. During the quantization process, the level of semantic IDs is set to 4, with each level consisting of 256 codebook vectors, and each vector has a dimension of 32. The RQ-VAE contains three components: a DNN encoder that encodes the input semantic embedding into a latent representation, a residual quantizer that outputs a quantized representation, and a DNN

decoder that decodes the quantized latent representation back to the semantic input embedding space. Both the encoder and decoder of RQ-VAE are implemented as Multi-Layer Perceptrons (MLPs) with ReLU activation functions. The model is optimized using the AdamW optimizer, employing a learning rate of 0.001 and a batch size of 1024.

Ordered Candidate Construction. To gather LLM preferences, following the LC-Rec setting (Zheng et al., 2024), we first adapt LLaMA-7B into a sequential recommender model using semantic IDs during training. At the inference stage, the recommender employs a beam search with a beam width of 25 for sampling and outputs candidate semantic item IDs based on cumulative probability scores. Due to the lack of constraints during decoding, the generated collection of semantic IDs may contain invalid IDs, and the reference item ID might not exist or be in the first position. Therefore, we correct the model's output to remove invalid IDs and ensure the target item is in the first position. Finally, we select the top 16 semantic IDs as candidates for each sample.

LOHRec Details. We utilize the open-source *Transformers* library to implement our sequential recommenders, adhering to the implementation details provided by Rajput et al. (2024). In the learning-to-rank paradigm of our LOHRec framework, for each user interaction sequence, we use 16 candidate semantic IDs and employ margin ranking loss (Liu et al., 2021) to enable the generative recommendation to learn ordered relationships among them. All the training process is based on the AdamW optimizer, alongside a cosine learning rate scheduler with the warmup.

During inference, the sequential recommender functions as a standard auto-regressive generator and employs a beam search strategy to retrieve multiple possible semantic IDs. Following previous works (Hua et al., 2023), we implement a prefix tree-based constrained decoding approach to ensure that all model outputs for item IDs are valid. The detailed constrained decoding process is elaborated in the Appendix A. Unless otherwise specified, we uniformly set the beam width to 20 across all datasets.

4.3 Evaluation Metrics

To comprehensively evaluate the performance of diverse sequential recommenders, we adopt two widely used evaluation metrics, including top-K Recall (Recall@K) and Normalized Discounted

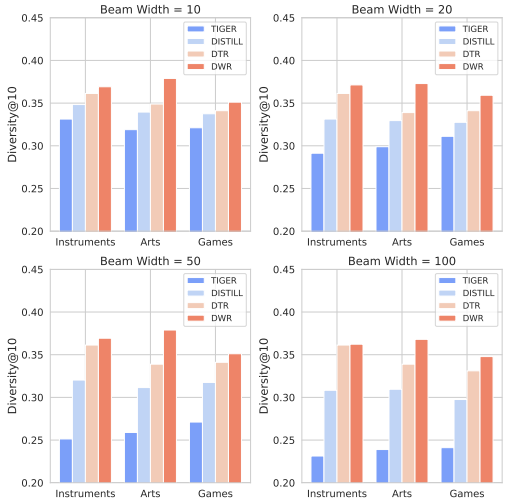

Figure 3: Diversity performance evaluation across TIGER and three variants of LOHRec on the three datasets, with beam width 10, 20, 50, and 100.

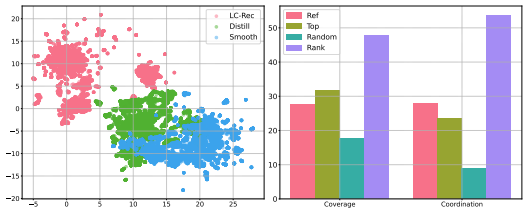

Figure 4: (Left): 2D visualization of embeddings of diverse item semantic IDs via UMAP. (Right): Comparative analysis of different methods for aligning LLM-based sequential recommender.

Cumulative Gain (NDCG@K) with $\mathrm{K} = 5, 10$. In addition, we are also interested in measuring and promoting diversity across recommended items. Here we consider intra-list diversity for the measure. Given the top-K recommended items $\{v_i\}_{i=1}^{\mathrm{K}}$ and the corresponding embedding list $[e_{v_1}, e_{v_2}, \cdots, e_{v_\mathrm{K}}]$, considering the paired item cosine similarity $\cos(e_{v_i}, e_{v_j}) = \frac{e_{v_i}^\top e_{v_j}}{\|e_{v_i}\|\|e_{v_j}\|}$, the intra-list diversity can be formulated as:

$$\text{Diversity@K} = \frac{2}{\mathrm{K}(\mathrm{K}+1)} \sum_{i=1}^{\mathrm{K}} \sum_{j=i}^{\mathrm{K}} 1 - \cos(e_{v_i}, e_{v_j}), \quad (17)$$

which takes values in $[0, 1]$.

4.4 Performance Comparison

With the methodological designs outlined in Section 3, we explore the experimental performance of four variants derived from the LOHRec framework. These variants include generalization through label smoothing (SMOOTH), model distillation (DISTILL), distillation-then-rank (DTR, two stages), and distillation with ranking (DWR, one stage).

Methods	Instruments				Arts				Games			
	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
Caser	.0543	.0710	.0355	.0409	.0379	.0541	.0262	.0313	.0367	.0617	.0227	.0307
HGN	.0813	.1048	.0668	.0744	.0622	.0875	.0462	.0544	.0517	.0856	.0333	.0442
GRU4Rec	.0821	.1031	.0698	.0765	.0749	.0964	.0590	.0659	.0586	.0964	.0381	.0502
BERT4Rec	.0671	.0822	.0560	.0608	.0559	.0713	.0451	.0500	.0482	.0763	.0311	.0401
SASRec	.0751	.0947	.0627	.0690	.0757	.1016	.0508	.0592	.0581	.0940	.0365	.0481
FMLP-Rec	.0786	.0988	.0638	.0704	.0757	.1046	.0541	.0634	.0571	.0930	.0361	.0476
FDSA	.0834	.1046	.0681	.0750	.0734	.0933	.0595	.0660	.0644	.1041	.0404	.0531
S^3-Rec	.0863	.1136	.0626	.0714	.0767	.1051	.0521	.0612	.0606	.1002	.0364	.0491
P5-CID	.0827	.1016	.0708	.0768	.0724	.0902	.0607	.0664	.0506	.0803	.0342	.0437
TIGER	.0863	.1064	.0738	.0803	.0788	.1012	.0631	.0703	.0599	.0939	.0392	.0501
Smooth	.0882	.1086	.0752	.0810	.0807	.1030	.0645	.0717	.0618	.0962	.0415	.0522
	+2.2%	+2.1%	+1.9%	+0.9%	+2.4%	+1.8%	+2.2%	+2.0%	+3.2%	+2.5%	+5.9%	+4.2%
Distill	.0949	.1151	.0814	.0870	.0914	.1113	.0725	.0799	.0680	.1018	.0435	.0540
	+10.0%	+8.2%	+10.3%	+8.3%	+16.0%	+10.0%	+14.9%	+13.7%	+13.5%	+8.4%	+11.0%	+7.8%
DTR	**.1021**	**.1254**	**.0875**	**.0920**	**.0970**	**.1212**	**.0795**	**.0860**	**.0729**	**.1103**	**.0515**	**.0586**
	+18.3%	+22.5%	+18.6%	+14.6%	+23.1%	+19.8%	+26.0%	+22.3%	+21.7%	+17.5%	+31.4%	+17.0%
DWR	.1013	.1230	.0868	.0912	.0945	.1202	.0766	.0842	.0716	.1088	.0499	.0569
	+17.4%	+15.6%	+17.6%	+13.6%	+19.9%	+18.8%	+21.4%	+19.8%	+19.5%	+15.9%	+27.3%	+13.6%

Table 1: Accuracy performance evaluation across all methods on the three datasets. The best results and best results from previous baselines are denoted in bold and underlined, respectively. R@K represents Recall@K and N@K represents NDCG@K.

We present the accuracy performance and diversity performance of the sequential recommendation models in Table 1 and Figure 3, respectively.

Considering these experimental results in Table 1, we observe that our proposed variants can significantly improve the accuracy performance of generative sequential recommenders. Specifically, compared to the representative generative recommender TIGER, all four variants derived from LOHRec outperform across the utilized metrics, demonstrating the strong generalization and potential of our approach. Among them, the accuracy performance of SMOOTH, DISTILL, and DWR increases sequentially, with DTR showing a substantial lead, indicating that LOHRec effectively aligns with the preferences of LLM-based recommenders. We believe that the leading performance of DTR is due to generating more candidate items during the retrieval stage, compared to DWR. Additionally, we observe that DWR outperforms previous state-of-the-art results across all metrics, indicating that by leveraging LLM knowledge, our method achieves the best performance even with comparable model sizes.

On the other hand, by analyzing the experimental results in Figure 3, we observe that the three variants exhibit similar superiority in diversity performance compared to TIGER. Specifically, in addition to setting the beam width to 20, we conducted additional diversity performance experiments with beam widths of 10, 50, and 100. We observed that TIGER's diversity@10 results significantly declined as the beam width increased, indicating that standard generative recommenders are prone to the problem of recommendation homogenization. In contrast, our methods exhibit a slower overall decline in diversity performance with increasing beam width, demonstrating the effectiveness of LOHRec in mitigating recommendation homogenization. Moreover, unlike the leading position of DTR in accuracy performance, we noticed that DWR excels in diversity performance and is able to maintain this advantage even as the beam width increases. We believe this is because DWR integrates the retrieval and ranking processes, resulting in final recommendations that are less homogenized compared to those of DTR that are selected from retrieved candidate items.

4.5 Ablation Study

In this section, we investigate the contributions of each component of DWR to the final recommendation performance. Specifically, DWR is composed of two components: model distillation for generative recommendation (DISTILL) and ranking objective learning ordered correlation (RANK). To investigate the contributions of each component, we independently removed each one. Moreover, we concurrently eliminated both of them, thereby regressing it to the state of standard generative sequential recommendation in an autoregressive manner. As demonstrated in Table 2, compared with

"w/o DISTILL", "w/o RANK", and "w/o All", we have the following observations: (1) Employing hierarchical semantics of LLM-based recommenders for distillation can significantly enhance the overall performance of generative recommendations; (2) Utilizing our learning-to-rank paradigm for LLM preference can significantly enhance the accuracy and diversity of recommended results.

4.6 Further Analyses

Embedding Visualization Analysis. To further investigate the effectiveness of our proposed framework in aligning LLM preferences, we delve into the relationships among embeddings of semantic item IDs encoded by different sequential recommenders. In this work, we utilize UMAP (McInnes et al., 2018) to visualize the item embeddings. Specifically, the 4096-dimensional vectors from the LLM-based recommender LC-Rec and the 384-dimensional vectors from both LOHRec variants SMOOTH and DISTILL are reduced to 2 dimensions for visualization. In Figure 4 (Left), we can observe that compared to SMOOTH, the item representations of DISTILL are closer to the distribution area of LC-Rec, indicating the effectiveness of DISTILL to align with LC-Rec.

LLM Alignment Analysis In addition, we explored the impact of different strategies on learning LLM preferences. In the experiment, we used the top 20 results recommended by LC-Rec as the ground truth and compared the effects of the following four methods: (1) using the next item as the label (Ref); (2) using the top 1 output of LC-Rec as the label (Top); (3) randomly sampling one from the LC-Rec outputs as the label (Random); and (4) using our ranking paradigm to learn the order relationships of the LC-Rec outputs (Rank). Inspired by the commonly used metrics in NLG tasks, ROUGE-1 and ROUGE-L (Lin, 2004), we proposed two metrics, "Coverage" and "Coordination", to measure the alignment effectiveness of our model from the perspectives of the intersection with the LLM recommended set and the longest common subsequence, respectively. As observed in Figure 4 (Right), the Rank method significantly outperforms other methods in terms of alignment performance, indicating the effectiveness of Equation 14 in aligning with LLM preferences.

5 Related Works

Large Language Model For Recommendation. Large language models have gained attention in recommendation systems, with various efforts to use them for modeling user behavior (Kang et al., 2023). LLMs have been employed in diverse recommendation tasks, including rating prediction, sequential recommendation, and direct recommendation(Bao et al., 2023; Zhang et al., 2023). Some efforts also tried to utilize LLMs to model structure relations. However, most approaches directly use LLMs as recommenders, which results in high costs and makes practical application difficult.

Self-Supervised Learning in Recommendation Recently, self-supervised learning has become popular in recommender system research. In collaborative filtering (CF), SSL4Rec (Yao et al., 2021) employs data augmentation on item features and introduces a contrastive pretraining objective to improve learned representations in the two-tower model. In knowledge-aware recommendation, KGCL (Yang et al., 2022) develops a knowledge graph contrastive learning framework to aid denoising and integration between CF learning and knowledge graph encoding. For the socially-aware recommendations, MHCN (Yu et al., 2021) designs a graph infomax task to accommodate cascading semantic information from social graphs, enhancing user representation learning. In the field of sequential recommendation, ICLRec (Chen et al., 2022) conducts clustering and contrastive learning on user intents, and it enhances sequential recommendation by improving the representation of user interests.

6 Conclusion and Future Work

In this paper, we study improving generative recommendation performance via leveraging the ubiquitous ordered correlation and hierarchical semantics in quantized item IDs. Specifically, we demonstrate the effectiveness of LOHRec in aligning the recommendation preferences of lightweight generative recommenders with those of LLM-based recommenders. Additionally, since LOHRec only uses additional tokens from the semantic item ID set, it can be applied to models with incompatible original vocabularies, demonstrating the sound generalizability.

For future work, our approach can be naturally extended to larger parameter scales and even various pre-trained language models. Additionally, our method can be applied to learning more types of sequential information to adapt to specific downstream tasks.

Limitations

Due to the extra GPU memory required for learning to rank candidate results in our framework, we were only able to evaluate our method on lightweight models. Experiments on larger models such as 7B were infeasible, as out-of-memory (OOM) errors occurred even with a batch size of 1. This limits the validation of our approach on large language models (LLMs). Furthermore, although we validated our method on three public datasets, evaluating the method on more and larger datasets is necessary to further demonstrate its generalizability.

Ethics Statement

All datasets used in this research are from public benchmark open-access datasets, which are anonymized and do not pose privacy concerns.

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.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Yongjun Chen, Zhiwei Liu, Jia Li, Julian McAuley, and Caiming Xiong. 2022. Intent contrastive learning for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, pages 2172–2182.

Zeshan Fayyaz, Mahsa Ebrahimian, Dina Nawara, Ahmed Ibrahim, and Rasha Kashef. 2020. Recommendation systems: Algorithms, challenges, metrics, and business opportunities. *applied sciences*, 10(21):7748.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 299–315.

Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 843–852.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. *Preprint*, arXiv:1511.06939.

Karatzoglou A HidasiB et al. 2015. Session-based recommendationswithrecurrentneuralnetworks. *arXiv preprint arXiv:1511.06939*.

Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. 2023. How to index item ids for recommendation foundation models. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 195–204.

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

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do llms understand user preferences? evaluating llms on user rating prediction. *arXiv preprint arXiv:2305.06474*.

Hyeyoung Ko, Suyeon Lee, Yoonseo Park, and Anna Choi. 2022. A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, 11(1):141.

Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*, pages 322–330.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yixin Liu, Zi-Yi Dou, and Pengfei Liu. 2021. Refsum: Refactoring neural summarization. *arXiv preprint arXiv:2104.07210*.

Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 825–833.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. 2024. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36.

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.

Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 565–573.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1434–1443.

Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H Chi, Steve Tjoa, Jieqi Kang, et al. 2021. Self-supervised learning for large-scale item recommendations. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 4321–4330.

Junliang Yu, Hongzhi Yin, Jundong Li, Qinyong Wang, Nguyen Quoc Viet Hung, and Xiangliang Zhang. 2021. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In *Proceedings of the web conference 2021*, pages 413–424.

Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv preprint arXiv:2305.07001*.

Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. 2019. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI*, pages 4320–4326.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. 2024. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1435–1448. IEEE.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1893–1902.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced mlp is all you need for sequential recommendation. In *Proceedings of the ACM web conference 2022*, pages 2388–2399.

A Constrained Decoding via Hierarchy

In general, the autoregressive model samples possible tokens from the whole vocabulary space at each time step during the decoding phase. However, for the sequential recommenders that incorporate generative retrieval paradigm and structured item semantic IDs, directly using such a decoding strategy often generates useless semantic IDs (i.e., no corresponding real item) since there are many invalid codeword combinations, leading to a waste of inference resources and low recommendation performance. Considering the distinct hierarchical structure of semantic item IDs via the residual quantization of RQ-VAE, it is intuitive to leverage these structured characteristics to constrain the sampling process during the decoding phase. Specifically, we first build a prefix tree based on the semantic IDs of all effective items in the dataset. For each node in the tree, we maintain a hash table from codewords to the corresponding child nodes. During the inference phase, we apply a beam search strategy for autoregressive decoding. Particularly, at each time step, we sample from all keys (i.e., all valid codewords currently) of the hash table of the current node, rather than from the entire vocabulary space. The more detailed process of our decoding strategy is shown in Algorithm 1.

B More Constructions of Ordered Prior as Baselines

In this paper, we mainly explore two additional types of ordered information mining methods for the generative sequential recommendation, leveraging the heuristic distance and global statistics, respectively.

B.1 Ordered Priors with Heuristic Distance

As we mentioned above, we use RQ-VAE to construct the semantic item ID, which has a coarse-to-fine hierarchy. Accordingly, we can consider that the earlier codewords in the semantic ID are more important because errors in the earlier parts greatly disrupt the semantic information that follows. Therefore, intuitively, we propose a heuristic method to measure the distance between semantic IDs. Specifically, given two semantic IDs of length D, $v^{(p)} = (c_1^{(p)}, c_2^{(p)}, \cdots, c_D^{(p)})$ and $v^{(q)} = (c_1^{(q)}, c_2^{(q)}, \cdots, c_D^{(q)})$, we measure their differences by considering the matches and weights at corre-

Settings	Instruments			Arts			Games		
	Recall@10	NDCG@10	Div@10	Recall@10	NDCG@10	Div@10	Recall@10	NDCG@10	Div@10
DWR	0.1278	0.0944	0.3711	0.1288	0.0892	0.3740	0.1179	0.0639	0.3553
w/o DISTILL	0.1092	0.0821	0.3423	0.1036	0.0724	0.3490	0.0964	0.0534	0.3161
w/o RANK	0.1169	0.0877	0.3525	0.1176	0.0808	0.3533	0.1090	0.0535	0.3355
w/o All	0.1078	0.0797	0.2913	0.1030	0.0712	0.2971	0.0928	0.0492	0.3116

Table 2: Ablation study results on the development sets of all datasets. Performance changes compared with our best model (DWR) are reported. Div@10 represents the intra-list diversity@10.

Algorithm 1 Beam Search with the Prefix Tree

Input: Top-k k, beam width b, prefix tree T
Output: Sequences $\{S_1, S_2, \ldots, S_k\}$
Initialize with start token $B = \{(\langle s \rangle, \text{root of } T, 1.0)\}$
repeat
  $B' \leftarrow \emptyset$
  for all (sequence s, node n, prob p) $\in B$ **do**
    Get the hash table H from the current node n
    $codewords \leftarrow$ keys of the hash table H
    for all token $t \in codewords$ **do**
      $s' \leftarrow s$ concatenated with t
      Compute probability $P(t|s) \times p$ for the new sequence s'
      $n' \leftarrow$ next node after taking token t from H
      Add $(s', n', P(t|s) \times p)$ to B'
    end for
  end for
  $B \leftarrow$ top b sequences from B' by total probability
until Stopping condition
Sort B by probability in descending order
return Top k sequences from B

sponding positions:

$$\text{Distance} = \sum_{i=1}^{D} 2^{D-i} \mathbb{I}(c_i^{(p)} \neq c_i^{(q)}), \qquad (18)$$

where $\mathbb{I}(\cdot)$ is the indicative function. The range of distances is from 0 to $2^D - 1$. In other words, given a reference semantic ID, we can construct a pseudo-semantic ID list of length $2^D - 1$, ordered by distance from smallest to largest.

B.2 Ordered Priors from Global Statistics

Sequential recommendation models tend to learn local sequential patterns during the training phase. We believe that enhancing the perception of their association with globally ordered information can improve the model's performance. To this end, we first construct the interaction matrix $\mathbf{M} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{V}|}$ between users and items by setting the entry $\mathbf{M}_{u,v} = 1$ if user u has interacted with item v and $\mathbf{M}_{u,v} = 0$ otherwise. With the operation $\mathbf{A}_{\mathcal{G}} = \mathbf{M}^\top \mathbf{M}$, we obtain the initial correlation strength between items in $\mathbf{A}_{\mathcal{G}}$ based on their co-interaction frequency. Then given the target item, we apply *top-k*$(\cdot)$ function to collect the top-k relevant items.

C More Experimental Details

C.1 Dataset Statistics

C.2 Baselines

We adopt the following representative models as baselines for comparison.

- **Caser** (Tang and Wang, 2018): It employs CNN in both horizontal and vertical ways to model high-order MCs for sequential recommendation
- **HGN** (Ma et al., 2019): It uses the hierarchical gating network to capture the long- and short-form interest from the user behavior sequence.

Dataset	#Users	#Items	#Interactions	Sparsity	Avg. Length
Instruments	24,773	9,923	205,153	99.92%	8.32
Arts	45,142	20,957	390,832	99.96%	8.66
Games	50,547	16,860	452,989	99.95%	8.96

Table 3: Statistics of used datasets.

- **GRU4Rec** (Hidasi et al., 2016): It uses GRU with ranking-based loss to model user sequences for the session-based recommendation.
- **BERT4Rec** (Sun et al., 2019): It adopts a bi-directional Transformer-based model and combines it with a mask prediction task for the modeling of item sequences.
- **SASRec** (Kang and McAuley, 2018): It uses a left-to-right Transformer language model to model user behavior sequence and predict the next item.
- **FMLP-Rec** (Zhou et al., 2022): It proposes an all-MLP model with learnable filters, which ensures efficiency and reduces noise signals.
- **FDSA** (Zhang et al., 2019): It focuses on the transformation patterns between item features, modeling both item-level and feature-level sequences separately through self-attention networks.
- **S^3-Rec** (Zhou et al., 2020): utilizes mutual information maximization to pre-train a self-supervised sequential recommendation model, learning the correlation between items and attributions.
- **P5-CID** (Geng et al., 2022; Hua et al., 2023): The authors organize multiple recommendation tasks in a text-to-text format, uniformly modeling these tasks using the T5 model. They then investigate methods for constructing item IDs for sequential recommendation, including sequential indexing and collaborative indexing. In our work, we use P5 with collaborative indexing as the baseline, following the implementation provided by the authors.
- **TIGER** (Rajput et al., 2024): It adopts the generative retrieval paradigm for the sequential recommendation and introduces the quantized semantic IDs based on RQ-VAE to uniquely identify items.

D Case Study

To further qualitatively analyze the effectiveness of our method, we present two cases of recommendation results on the Amazon-Games test set using the sequence recommendation models TIGER and DWR. In Table 4, we observe that the recommendation results generated by TIGER contain an entry "ERROR" highlighted in red, indicating an invalid semantic ID that cannot be mapped to the actual item. Additionally, there is a repeated occurrence of the item "Metroid: Other M" (highlighted in orange), suggesting that TIGER, while avoiding identical semantic IDs, still tends to recommend highly similar semantic IDs that may map to the same item instance. In contrast, DWR ensures valid semantic IDs, with the recommendation results being more accurate and diverse. In Table 5, the recommendation results from TIGER exhibit severe repetition, whereas our recommender maintains good diversity in its recommendations.

Case 1
Input Sequence:
Final Fantasy Crystal Chronicles: Echoes of Time - Nintendo Wii.
My Fitness Coach - Nintendo Wii.
Super Mario Galaxy 2.
Final Fantasy Fables: Chocobo's Dungeon - Nintendo Wii.
Super Mario Galaxy.
Castlevania Judgment.
Kingdom Hearts Re:Chain of Memories.
Final Fantasy X-2.
The Legend of Zelda: Skyward Sword with Music CD.
Summoner.
Beyond Good & Evil.
Target:
Tales Of Symphonia: Dawn of the New World.
Results of TIGER:
Metroid: Other M.
Super Mario Galaxy 2.
Sin and Punishment: Star Successor - Nintendo Wii.
The Last Story - Nintendo Wii.
ERROR.
Super Paper Mario (Nintendo Selects).
Lunar: Dragon Song - Nintendo DS.
Metroid: Other M.
Knights in the Nightmare - Nintendo DS.
Super Smash Bros. Brawl.
Results of LOHRec-LLM:
The Legend of Zelda: Twilight Princess.
Final Fantasy IV.
Metroid: Other M.
The Legend of Zelda: The Wind Waker.
Super Mario Galaxy 2.
Final Fantasy Tactics A2: Grimoire of the Rift.
Rune Factory: Frontier - Nintendo Wii.
Final Fantasy XII.
Tales Of Symphonia: Dawn of the New World.
Dragon Quest IV: Chapters of the Chosen - Nintendo DS.

Table 4: Items recommended by TIGER and LOHRec-DWR trained on Amazon-Games dataset.

Case 2
Input Sequence:
Vacation Quest: Australia - PC.
Mystery P.I. Portrait of a Thief - Nintendo DS.
Amazing Adventures: Around The World - PC.
Mystery P.I.: The Lottery Ticket - PC.
Amazing Adventures The Riddle Of Two Knights.
Mystery P.I.: The London Caper.
Amazing Adventures: The Forgotten Dynasty.
Mahjong Escape: Ancient China - PC.
Bejeweled - PC.
Target:
Escape the Emerald Star - PC/Mac.
Results of TIGER:
Ancient Secrets: Quest for the Golden Key.
Big Fish: Sable Maze 1: Sullivan River and Sable Maze 2: Norwich Caves - PC.
Dark Tales 3:Edgar Allen Poe's The Premature Burial - PC.
Mystery P.I: The Curious Case of Counterfeit Cove.
Amazing Adventures: The Lost Tomb - PC.
Holiday Pack - 3 in 1 - Hidden Object Game [Download].
Chinatown Chronicles - Hidden Objects Game [Download].
Ancient Secrets: Quest for the Golden Key.
Big Fish: Sable Maze 1: Sullivan River and Sable Maze 2: Norwich Caves - PC.
Dark Tales 3:Edgar Allen Poe's The Premature Burial - PC.
Results of LOHRec-LLM:
Mystery P.I: The Curious Case of Counterfeit Cove.
Vacation Quest: Australia - PC.
Mystery P.I.: The London Caper.
Golden Trails: The New Western Rush [Download].
Dark Tales 3:Edgar Allen Poe's The Premature Burial - PC.
100% Hidden Objects [Download].
Awakening: Moonfell Wood.
Mystic Diary: Haunted Island.
Super Market - Hidden Object Game [Download].
Travel Pack - 3 in 1 - Hidden Object Game [Download].

Table 5: Items recommended by TIGER and LOHRec-DWR trained on Amazon-Games dataset.