Expressiveness Limits of Autoregressive Semantic ID Generation in Generative Recommendation

Yupeng Hou[1] **Haven Kim**[1] **Clark Mingxuan Ju**[2] **Eduardo Escoto**[1]
Neil Shah[2] **Julian McAuley**[1]
[1]University of California, San Diego [2]Snap Inc.
{yphou,khaven,eduardo,jmcauley}@ucsd.edu,
{mju,nshah}@snap.com

Abstract

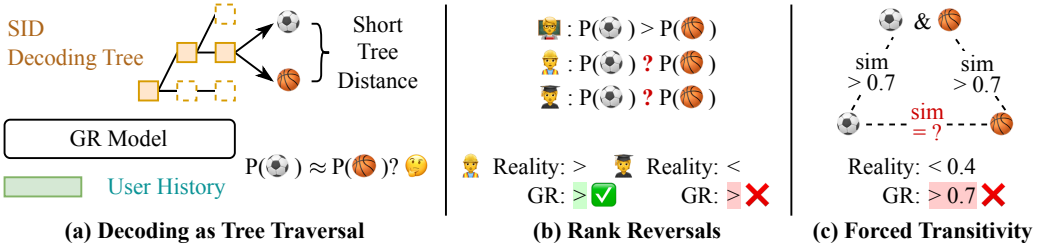
Generative recommendation (GR) models generate items by autoregressively producing a sequence of discrete tokens that jointly index the target item. However, this autoregressive generation process also induces a structured decoding space whose impact on model expressiveness remains underexplored. Specifically, token-by-token generation can be viewed as traversing a decoding tree induced by semantic ID tokens, where leaf nodes correspond to candidate items. We observe that the item probabilities produced by GR models are strongly correlated with this tree structure: items that are close in the tree tend to receive similar probabilities for any given user, making it difficult to distinguish among them based on user-specific preferences. We further show theoretically that such structural correlations prevent GR models from representing even simple patterns that can be well captured by conventional collaborative filtering models. To mitigate this issue, we propose Latte, a simple modification that injects a latent token before each semantic ID, reshaping the decoding space from a single tree into multiple latent-token-conditioned trees. This design creates multiple paths with varying tree distances between items, relaxing tree-induced probability coupling and yielding an average of 3.45% relative improvement on NDCG@10. Our code is available at https://github.com/hyp1231/Latte.

1 Introduction

Generative recommendation (GR) [34, 61, 5, 10] tokenizes items as sequences of discrete tokens (semantic IDs or SIDs [40, 46, 34]). Unlike traditional models that score user-item preferences via representation similarity [11, 21], GR models autoregressively generate SID tokens, scoring an item as the product of its tokens' conditional probabilities. While existing research has largely focused on improving how items are tokenized [42, 66, 16], the autoregressive generation process itself is often taken for granted, despite directly determining how item scores are composed and constrained.

In this work, we take the first step towards understanding the expressive power of this autoregressive token generation process of GR models. Specifically, we investigate whether there exist specific user-item preference patterns that GR models struggle to express. Note that when calculating the probability of an item (*i.e.,* a sequence of tokens), items sharing the same initial tokens also share common terms in the probability calculation, leading to correlated item probabilities. This motivates us to formulate a *decoding tree* view of the autoregressive process, where generating a token corresponds to traversing one level down the tree, and each leaf node represents a candidate item (with the path corresponding to a valid semantic ID). Based on this formulation, we investigate the relationship between learned item probabilities and the decoding tree structure.

Preprint.



(a) Decoding as Tree Traversal **(b) Rank Reversals** **(c) Forced Transitivity**

Figure 1: Illustration of the expressive limitations induced by autoregressive SID generation. (a) Generating a SID can be viewed as traversing a decoding tree, where items sharing longer prefixes have shorter tree distances and tend to receive highly correlated generation probabilities. (b) shows that GR tends to assign consistent relative preference scores across users, while (c) shows that the tree structure can force two items to be similar even when their true similarity is low.

Through empirical analysis, we find that if two items have a short distance in the decoding tree (*i.e.,* they share a long prefix), their predicted probabilities are strongly correlated for any given user. Intuitively, this implies that if a GR model predicts Alice prefers item A over item B (where A and B are close in the tree), it is highly likely to predict that Bob also prefers A over B. This contradicts the fundamental assumption of personalized recommendation that preferences should be user-specific. Driven by this observation, we theoretically demonstrate that there exist simple user-item preference patterns (Section 3.3) and item-item similarity relationships (Section 3.4) that GR models struggle to represent, thereby limiting their expressive power.

To alleviate this issue, we propose **Latte**, a simple yet effective modification to the standard SID generation approach. We introduce a small set of additional tokens, named ***latent tokens***. During training, we inject a randomly sampled latent token before the target semantic ID. The concatenated sequence then serves as the new optimization target. This modification effectively reshapes the decoding structure from a single universal tree to multiple trees rooted at a shared hyper-root, where the second level corresponds to the latent tokens. This design enables multiple paths with varying tree distances between any pair of items, thus relaxing the strong correlation between tree structure and item probabilities in existing GR models. Experiments on benchmarks validate the effectiveness of the proposed method, leading to an average of 3.45% relative improvement on NDCG@10.

Finally, we provide a further exploration of binding latent tokens with inductive biases. Specifically, in a multimodal scenario where each token corresponds to a modality of item features, we bind each latent token to a specific permutation of the semantic ID. Empirically, we show that latent tokens associated with better-performing permutations indeed receive higher selection frequency, leading to improved overall recommendation performance.

2 Preliminaries

Semantic ID. A SID refers to a sequence of discrete tokens that jointly index an item. Formally, a SID can be represented as a tuple $(c^{(1)}, c^{(2)}, \ldots, c^{(m)})$, where m denotes the length of the SID. Each token $c^{(j)}$ is selected from a compact vocabulary $\mathcal{C}^{(j)}$ of size M that is shared across all items.

Generative recommendation. GR models aim to predict the next item i_t with which a user will interact, given the historical interaction sequence $(i_1, i_2, \ldots, i_{t-1})$. In this framework, each item is tokenized into its corresponding semantic ID. Formally, given the semantic IDs for historical interactions $u = (c_1^{(1)}, \ldots, c_{t-1}^{(m)})$, the model is trained to generate the target semantic ID $(c_t^{(1)}, c_t^{(2)}, \ldots, c_t^{(m)})$ in a token-by-token manner. Therefore, the user-item preference score is often defined as the joint probability of generating the target Semantic ID:

$$\mathbb{P}(i_t \mid u) = \prod_{j=1}^{m} \mathbb{P}(c_t^{(j)} \mid c_t^{(1)}, \ldots, c_t^{(j-1)}, u). \tag{1}$$

3 Limitations of Autoregressive SID Generation

In this section, we analyze the expressive limitations of the autoregressive semantic ID generation process in generative recommendation models. We first formulate the generation process as a tree

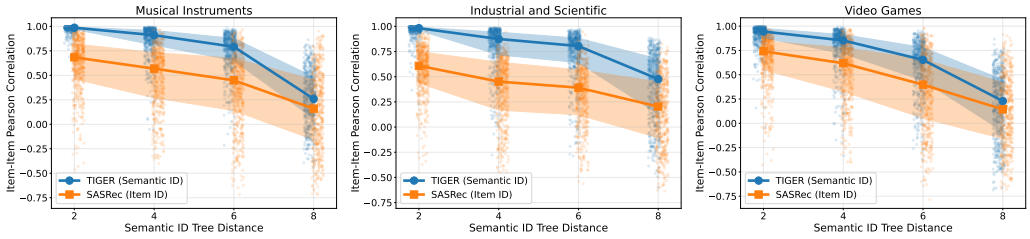


Figure 2: Correlation between tree distance and item generation probability similarity.

traversal procedure in Section 3.1. Next, we empirically demonstrate a strong correlation between the decoding tree structure and item generation probabilities in Section 3.2. Based on these observations, we provide a theoretical analysis showing that such structural correlations constrain the expressive power of generative recommendation (GR) models, causing them to struggle with expressing specific patterns of user-item preferences (Section 3.3) and item-item similarities (Section 3.4).

3.1 Generation as Tree Traversal

Generative recommendation models predict the next item by autoregressively generating a sequence of semantic ID tokens. We can formulate this generation process as traversal along a decoding tree induced by the set of valid semantic IDs. We define the decoding tree as follows:

Definition 3.1 (Decoding Tree)**.** Let $\mathcal{T}$ be a decoding tree of depth m formed by all valid semantic IDs. The root represents the empty sequence, and each node at depth j corresponds to a unique prefix $(c^{(1)}, \ldots, c^{(j)})$. An edge connects a node to its child if the child's prefix extends the parent's by one valid token. Consequently, each leaf node uniquely represents an item.

Under this formulation, the autoregressive generation of a semantic ID is equivalent to traversing from the root to a leaf node in the decoding tree. We adopt this perspective because it links the number of shared prefix tokens between two items to a well-defined distance metric:

Definition 3.2 (Tree Distance)**.** Consider two items i and i' with semantic IDs $(c^{(1)}, \ldots, c^{(m)})$ and $(c'^{(1)}, \ldots, c'^{(m)})$, respectively. Let k denote the length of the longest common prefix shared by these IDs. The tree distance between i and i' is defined as $d_{\mathcal{T}}(i, i') = 2(m - k)$, which corresponds to the number of edges on the unique path connecting the corresponding leaf nodes in $\mathcal{T}$.

3.2 Tree Distance *vs.* Item Generation Probability

Motivation. Based on Equation (1), consider two items i and i' with a tree distance of $2(m - k)$, implying they share a prefix of length k. The item generation probabilities can be factored as follows:

$$\mathbb{P}(i \mid u) = \prod_{j=1}^{k} \mathbb{P}(c^{(j)} \mid c^{(1)}, \ldots, c^{(j-1)}, u) \cdot \prod_{j=k+1}^{m} \mathbb{P}(c^{(j)} \mid c^{(1)}, \ldots, c^{(j-1)}, u), \tag{2}$$

$$\mathbb{P}(i' \mid u) = \prod_{j=1}^{k} \mathbb{P}(c'^{(j)} \mid c'^{(1)}, \ldots, c'^{(j-1)}, u) \cdot \prod_{j=k+1}^{m} \mathbb{P}(c'^{(j)} \mid c'^{(1)}, \ldots, c'^{(j-1)}, u), \tag{3}$$

where $c^{(j)} = c'^{(j)}$ for all $j \leq k$ (the shared prefix). The highlighted terms in Equations (2) and (3) are identical, as they depend solely on the common prefix. This factorization suggests that items with smaller tree distances (longer shared prefixes) will likely exhibit more similar generation probabilities, largely independent of the user's preference.

Empirical validation. To validate this hypothesis, we compare TIGER [34], a representative GR model, with SASRec [21], a standard item ID-based model, across three public datasets (detailed settings are provided in Section 5.1). We uniformly sample 1,024 pairs of items across varying tree distances ($\{2, 4, 6, 8\}$). For each pair (i, i'), we randomly sample 512 users $\{u_k\}_{k=1}^{512}$, compute the generation probabilities $\mathbb{P}(i \mid u_k)$ and $\mathbb{P}(i' \mid u_k)$ for each sampled user, and calculate the Pearson correlation coefficient between these two probability sequences.

Correlation analysis. The empirical results demonstrate that item generation probabilities are highly correlated when items are close in the decoding tree. Furthermore, the strength of this correlation increases monotonically as tree distance decreases. For instance, we observe a Pearson correlation of ~ 1.0 for items with a tree distance of 2, which drops to > 0.8 for a distance of 4. Formally, we encode this observation into the following assumption for our theoretical analysis:

Assumption 3.3. For any pair of items i and i', let $\rho(i, i')$ denote the Pearson correlation coefficient between their generation probabilities $\mathbb{P}(i \mid u)$ and $\mathbb{P}(i' \mid u)$ across the user population. We assume that for any threshold δ, the probability $\mathbb{P}(\rho(i, i') > \delta)$ is monotonically decreasing w.r.t. the tree distance $d_{\mathcal{T}}(i, i')$.

3.3 Limitation on User-Item Preference Modeling

In this section, we analyze how the structural property observed in Section 3.2 constrains the model's ability to express diverse user-item preference patterns. We intuitively formulate the limitation as the inability to express *rank reversals* between items that are close in the tree.

Rank reversals. Let u and u' be two independent users drawn from the population $\mathcal{U}$. We define the rank reversal event $\mathcal{R}(i, i')$ for a pair of items i and i' as the scenario where users have opposite relative preferences:

$$\begin{aligned}\mathcal{R}(i, i') \triangleq &\{\mathbb{P}(i \mid u) > \mathbb{P}(i' \mid u) \wedge \mathbb{P}(i \mid u') < \mathbb{P}(i' \mid u')\} \\ &\cup \{\mathbb{P}(i \mid u) < \mathbb{P}(i' \mid u) \wedge \mathbb{P}(i \mid u') > \mathbb{P}(i' \mid u')\}. \end{aligned} \tag{4}$$

Rank reversals are essential for personalized recommendation; without them, the relative order of i and i' would be identical for all users, reducing to a non-personalized ranking. Specifically, we have the following theorem bounding the probability of rank reversals based on the correlation of generation probabilities.

Theorem 3.4 (Correlation Suppresses Rank Reversals)**.** *Let σ^2 denote the variance of the generation probabilities (assumed equal for i and i', see Section B). The rank reversal probability is bounded by:*

$$\mathbb{P}(\mathcal{R}(i, i')) \leq \frac{4\sigma^2(1 - \rho(i, i'))}{\mu^2 + 2\sigma^2(1 - \rho(i, i'))}, \tag{5}$$

where $\mu = |\mathbb{E}[\mathbb{P}(i \mid u) - \mathbb{P}(i' \mid u)]|$ is the expected difference in generation probabilities and $\rho(i, i')$ is the correlation defined in Theorem 3.3.

A detailed proof is provided in Appendix B.

Implication. Theorem 3.4 implies that as $\rho(i, i') \to 1$, the rank-reversal probability $\mathbb{P}(\mathcal{R}(i, i')) \to 0$. Combining with Theorem 3.3, we conclude that for items with small tree distance $d_{\mathcal{T}}(i, i')$, GR is structurally constrained to assign similar relative ranking of i and i' across all users. This limits the model's ability to capture users whose preferences diverge from the majority trend.

3.4 Limitation on Item-Item Similarity Modeling

Beyond user-item preferences, a powerful recommender system should also capture complex item-item similarities. To analyze this, we adopt a collaborative filtering view [36, 37], where item-item similarity is induced by the inner product of the user-item preference matrices. A critical property of such similarity in real-world scenarios is that it is *not necessarily transitive*. Consider a scenario with three items and two user groups G_A and G_B:

- Item i_1: Preferred by G_A.
- Item i_2: Preferred by both G_A and G_B.
- Item i_3: Preferred by G_B.

In this case, a flexible model should capture that i_1 is similar to i_2 (co-preferred by G_A) and i_2 is similar to i_3 (co-preferred by G_B), while simultaneously reflecting that i_1 and i_3 are dissimilar (disjoint user base). However, the strong correlation between the tree structure and item generation probabilities identified in Section 3.2 hinders the model's ability to represent such intransitive similarity relationships. We formalize this limitation in the following theorem:

Theorem 3.5 (Forced Transitivity)**.** *Based on Assumption 3.3, suppose high similarity (correlation $> \tau$) implies small tree distance $d_{\mathcal{T}} \leq \delta$, and conversely $d_{\mathcal{T}} \leq \delta$ implies correlation $> \tau$. Then, if the model captures similarities for both (i_1, i_2) and (i_2, i_3) (correlation $> \tau$), it is structurally forced to assign correlation $> \tau$ to (i_1, i_3), preventing the representation of their dissimilarity.*

A detailed proof is provided in Appendix C.

Implication. The proof leverages the fact that the tree distance $d_{\mathcal{T}}$ satisfies the ultrametric inequality: $d_{\mathcal{T}}(i_1, i_3) \leq \max(d_{\mathcal{T}}(i_1, i_2), d_{\mathcal{T}}(i_2, i_3))$. This structural property implies that items sharing common similar neighbors in the decoding tree are forced to be close to each other. Consequently, generative recommendation models may struggle to distinguish items that are locally similar to the same set of items but distinct from each other (*e.g.,* sharing different features or preferred by different user groups), effectively limiting the model's expressiveness.

4 Alleviating Expressive Limits

Having analyzed the expressive limitations of standard autoregressive semantic ID generation in GR, we now present **Latte** (Figure 3), a simple yet effective modification designed to relax the constraints imposed by the decoding tree structure and enhance model expressiveness. The core idea is to condition the model to predict an additional latent token prior to generating the semantic ID tokens. This effectively introduces a super-root node that connects multiple copies of the original decoding trees, allowing for multiple paths with varying tree distances between any pair of items (leaf nodes of the decoding tree). Below, we detail the training (Section 4.1) and inference (Section 4.2) processes of Latte, and discuss how this straightforward adjustment improves the expressive power of GR models (Section 4.3).

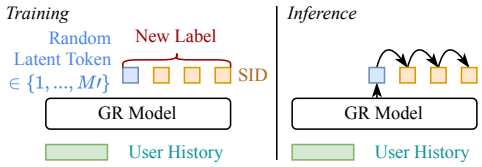


Figure 3: Overall framework of Latte.

4.1 Training: Sampling Latent Tokens

Latte uses a small set of additional discrete tokens, termed latent tokens, denoted as $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_{M'}\}$, where $M' \ll M$ represents the number of latent tokens. During training, for each target semantic ID $(c_t^{(1)}, c_t^{(2)}, \ldots, c_t^{(m)})$, we randomly sample a latent token $\ell \in \mathcal{L}$ and prepend it to the semantic ID, thereby creating an augmented target sequence $(\ell, c_t^{(1)}, c_t^{(2)}, \ldots, c_t^{(m)})$. The GR model is then trained to generate this augmented sequence autoregressively using the standard next-token prediction loss.

4.2 Inference: Generating Latent Tokens

At inference time, we impose no constraints on the selection of latent tokens. Instead, we allow the model to generate autoregressively, initiating generation with a latent token followed by the semantic ID tokens. Consequently, the user-item preference score is reformulated as:

$$\mathbb{P}(i_t \mid u) = \operatorname*{Agg}_{\ell \in \mathcal{L}} \left(\mathbb{P}(\ell \mid u) \cdot \prod_{j=1}^{m} \mathbb{P}(c_t^{(j)} \mid \ell, c_t^{(1:j-1)}, u) \right),$$

where Agg denotes the aggregation operator, which can be instantiated via operations such as summation (sum) or maximization ($\max$). In practice, following prior work [34, 61], we employ beam search during inference to efficiently approximate this aggregation.

4.3 Improved Expressive Power via Latent Tokens

The analysis in Section 3 demonstrates that the fixed tree distance $d_{\mathcal{T}}$ enforces a high correlation between the generation probabilities of structurally close items. Latte effectively alleviates this limitation by introducing latent tokens $\mathcal{L}$, thereby expanding the single decoding tree into a forest of $|\mathcal{L}|$ trees. This structural modification enables dynamic distances between items.

Table 1: Performance comparisons between different methods and the proposed method Latte. The best and second-best results are highlighted in **bold** and underlined font, respectively. "Δ" indicates the performance gain of Latte over the best baseline. "*" denotes statistically significant improvements ($p < 0.05$) over the best baseline according to a paired t-test.

Methods	Instrument				Scientific				Game			
	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
GRU4Rec	0.0324	0.0501	0.0209	0.0266	0.0202	0.0338	0.0129	0.0173	0.0499	0.0799	0.0320	0.0416
BERT4Rec	0.0307	0.0485	0.0195	0.0252	0.0186	0.0296	0.0119	0.0155	0.0460	0.0735	0.0298	0.0386
SASRec	0.0333	0.0523	0.0213	0.0274	0.0259	0.0412	0.0150	0.0199	0.0535	0.0847	0.0331	0.0438
FMLP-Rec	0.0339	0.0536	0.0218	0.0282	0.0269	0.0422	0.0155	0.0204	0.0528	0.0857	0.0338	0.0444
HSTU	0.0343	0.0577	0.0191	0.0271	0.0271	0.0429	0.0147	0.0198	0.0578	0.0903	0.0334	0.0442
FDSA	0.0347	0.0545	0.0230	0.0293	0.0262	0.0421	0.0169	0.0213	0.0544	0.0852	0.0361	0.0448
S^3-Rec	0.0317	0.0496	0.0199	0.0257	0.0263	0.0418	0.0171	0.0219	0.0485	0.0769	0.0315	0.0406
TIGER	0.0370	0.0564	0.0244	0.0306	0.0264	0.0422	0.0175	0.0226	0.0559	0.0868	0.0366	0.0467
LETTER	0.0372	0.0580	0.0246	0.0313	0.0279	0.0435	0.0182	0.0232	0.0563	0.0877	0.0372	0.0473
ActionPiece	0.0383	0.0615	0.0243	0.0318	0.0284	0.0452	0.0182	0.0236	0.0591	0.0927	0.0382	0.0490
PSID	0.0390	0.0602	0.0256	0.0325	0.0278	0.0445	0.0181	0.0235	0.0599	0.0939	0.0391	0.0500
Latte	**0.0401***	**0.0618**	**0.0261***	**0.0331***	**0.0304***	**0.0470***	**0.0196***	**0.0249***	**0.0618***	**0.0958***	**0.0406***	**0.0515***
Δ	+2.82%	+0.49%	+1.95%	+1.85%	+7.04%	+3.98%	+7.69%	+5.51%	+3.17%	+2.02%	+3.84%	+3.00%

Dynamic tree distance. In standard GR, tree distance $d_{\mathcal{T}}(i, i')$ remains constant. In contrast, Latte employs an aggregation operation (*e.g.,* $\mathrm{Agg} = \max$) that allows the generation process for a user u to select a specific latent path. Let $\ell^*(i, u) = \arg\max_\ell \mathbb{P}(\ell \mid u)\mathbb{P}(i \mid \ell, u)$ denote the dominant latent token for generating item i given user u. The "effective" structural distance between i and i' thus becomes context-dependent:

$$d_{\text{eff}}(i, i'; u) = \begin{cases} d_{\mathcal{T}}(i, i') & \text{if } \ell^*(i, u) = \ell^*(i', u), \\ 2(m+1) & \text{if } \ell^*(i, u) \neq \ell^*(i', u). \end{cases} \tag{6}$$

By assigning distinct latent tokens to items, the model can significantly increase the effective distance. This implies that the probability computations for structurally close items diverge at an earlier stage (see Equations (2) and (3)). This capability enables the model to reflect lower correlations even for items with similar SIDs (empirically validated in Section 5.3). Proof can be found in Section C.4.

5 Experiments

5.1 Experimental Setup

Datasets. Following previous works [25, 62], we conduct experiments on three categories of the Amazon Reviews 2023 dataset [13]: **Instruments**, **Scientific**, and **Games**. Each user's review history is grouped into a single sequence and sorted chronologically. We adopt the widely used leave-one-out strategy [21, 34] for data splitting, using the most recent interaction for testing, the second most recent for validation, and the remainder for training. Data statistics are summarized in Table 7.

Evaluation details. We adopt PSID [59] with RQ-Kmeans [19] item tokenization as our base model. Please refer to Section D for detailed implementation and evaluation settings.

5.2 Main Results

We compare the proposed Latte method with all baselines in Table 1. From the results, we can see that Latte consistently outperforms its base model, PSID, and all other baselines across all datasets and metrics. Specifically, Latte achieves an average of 3.45% relative improvement on NDCG@10 with only one simple modification, *i.e.,* generating an additional latent token before semantic IDs, demonstrating the effectiveness of relaxing the constraints discussed in Section 3.

5.3 Tree-Structure Correlation Analysis

To quantitatively compare structural constraints across different models, we analyze the association between the decoding tree structure and the item probabilities. Specifically, we adopt the definition of

Table 3: Performance comparison across different tokenization methods (NDCG@10). Δ denotes Latte's relative improvement over the base model PSID. All improvements are statistically significant ($p < 0.05$) according to the paired t-test.

Tokenization	Model	Instruments	Scientific	Games
OPQ	PSID	0.0313	0.0232	0.0478
	Latte	**0.0318**	**0.0235**	**0.0493**
	Δ	+1.68%	+1.37%	+3.24%
RQ-VAE	PSID	0.0325	0.0241	0.0490
	Latte	**0.0331**	**0.0247**	**0.0502**
	Δ	+2.02%	+2.76%	+2.55%
RQ-KMeans	PSID	0.0325	0.0235	0.0500
	Latte	**0.0331**	**0.0249**	**0.0515**
	Δ	+2.02%	+5.90%	+3.11%

Table 4: Performance comparison of different aggregation methods (NDCG@10).

Aggregation Method	Instruments	Scientific	Games
PSID	0.032462	0.023531	0.049994
Latte (Agg = sum)	**0.033134**	<u>0.024301</u>	<u>0.051476</u>
Latte (Agg = max)	<u>0.033114</u>	**0.024920**	**0.051547**

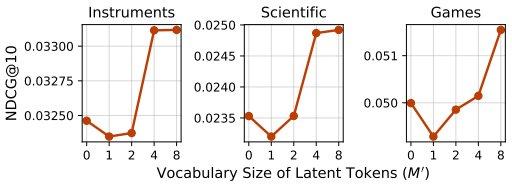


Figure 4: Analysis of model performance w.r.t. the number of latent tokens M'.

item-item similarity introduced in Section 3.2, defined as the Pearson correlation of item generation probabilities across the sampled user population. Since tree distances typically take only a few values (*e.g.,* $\{2, 4, 6, 8\}$), we employ Kendall's rank correlation coefficient [22] for this measurement. This metric is more robust for evaluating monotonic associations between ranked variables. A stronger association (approaching -1 or 1) implies that item similarities are heavily dominated by the decoding tree structure. Conversely, a value closer to 0 is desirable, as it indicates the model's capacity to learn flexible item relationships that are not strictly bound by the underlying tree structure.

As shown in Table 2, Latte consistently achieves lower absolute Kendall correlation values compared to its base model, PSID, while using the exact same semantic IDs. This suggests that generating an additional latent token effectively relaxes the structural constraints imposed by the decoding tree. Notably, there is a large numerical gap between TIGER and PSID, which may be attributed to TIGER's greater tree depth (4 *vs.* 3). However, because these two models use different sets of semantic IDs, they are not directly comparable. We include the TIGER results as a reference only; how to fairly compare structural constraints across models with different semantic IDs remains an open research question.

Table 2: Kendall's rank correlation between tree distance and item-item similarity. **Bold** numbers indicate the fewest constraints.

Model	Instruments (↑)	Scientific (↑)	Games (↑)
TIGER	-0.7170	-0.6137	-0.6462
PSID	-0.6225	-0.4611	-0.6072
Latte	**-0.6030**	**-0.4451**	**-0.5958**

5.4 In-Depth Analysis

5.4.1 Performance w.r.t. Item Tokenization Method

To investigate the generalizability of Latte across different item tokenization strategies, we conduct experiments using three representative tokenizers: OPQ [12, 3, 14], RQ-VAE [34, 42, 66, 62], and RQ-KMeans [19, 5]. As shown in Table 3, Latte consistently outperforms the base model PSID across all tokenization methods and datasets. This result demonstrates the broad applicability of our proposed method. Interestingly, while the base model performs best when paired with RQ-VAE, the introduction of latent tokens allows RQ-KMeans to achieve superior performance. We hypothesize that tokenizers not heavily optimized for the target corpus, such as RQ-KMeans, which lacks the end-to-end training of RQ-VAE, may offer greater optimization headroom.

5.4.2 Performance w.r.t. Inference Aggregation Method

In Section 4.2, we introduced two aggregation methods for combining the scores of generations associated with different latent tokens that point to the same target item: sum and max. We compare their performance in Table 4. We observe that both aggregation methods outperform the base model, while the performance difference between the two is relatively small. These results demonstrate that both strategies are effective and that the choice of aggregation method is robust.

Table 5: Performance comparison on MPD dataset with different modality orders.

Method	Recall@10	NDCG@10
Base: Fixed Modality Order		
playlist → tag → metadata	0.1966	0.1266
playlist → metadata → tag	0.1972	0.1268
tag → playlist → metadata	0.1948	0.1255
tag → metadata → playlist	0.1939	0.1250
metadata → playlist → tag	0.1933	0.1247
metadata → tag → playlist	0.1942	0.1248
Ours: Dynamic Modality Ordering		
Agg = max	<u>0.2060</u>	<u>0.1335</u>
Agg = sum	**0.2107**	**0.1353**

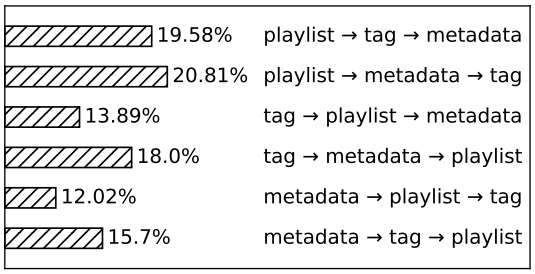


Figure 5: Distribution of modality orders corresponding to the generated permutation tokens.

5.4.3 Performance w.r.t. Latent Token Vocabulary Size

We investigate the impact of latent token vocabulary size on model performance. Specifically, we tune the number of introduced latent tokens $M' \in \{0, 1, 2, 4, 8\}$, where 0 denotes the base model PSID. As shown in Figure 4, we observe that introducing only one or two latent tokens usually leads to a performance drop due to accumulated errors in latent token prediction. However, when the vocabulary size increases to 4 and 8, the models outperform the base model. Overall, these results suggest that a moderate number of latent tokens can effectively improve model performance.

6 Incorporating Inductive Bias into Latent Tokens

In Section 4, we proposed generating latent tokens prior to semantic IDs to enhance model expressiveness. While effective, these latent tokens are sampled uniformly at random during training. In this section, we explore the possibility of anchoring latent tokens to specific inductive biases.

6.1 Latent Tokens as SID Permutation Indicators

We consider a multimodal setting where each token corresponds to a specific modality of item features. Existing works typically adopt a fixed modality order to organize an item's semantic IDs [8, 67, 56, 58], largely based on human heuristics. However, user preferences over different modalities may vary. In addition, identifying a globally optimal modality ordering is non-trivial. While a straightforward approach would involve enumerating all possible permutations and training separate models, such a strategy is computationally exhaustive. We are therefore interested in whether latent tokens can enable the model to adaptively select modality orders for different users, or alternatively, whether this mechanism can facilitate the discovery of globally best orderings in a purely data-driven manner.

Specifically, we bind each latent token to a specific permutation of the SID sequence (representing a unique ordering of modalities). During training, we continue to sample latent tokens uniformly; however, each token now dictates a specific permutation of the SIDs. We permute the SIDs accordingly before concatenating them with the latent token. During inference, the model generates latent tokens autonomously, followed by the SIDs. The only modification is that the generated SIDs are de-permuted back to their original order based on the preceding latent token.

6.2 Experiments and Analysis

Dataset. We evaluate our approach on the Million Playlist Dataset (MPD) [4]. Following prior work [23], we filter out songs lacking tags or metadata and exclude playlists with fewer than six songs. The dataset is split into training, validation, and test sets using an 8:1:1 ratio.

Data processing. We incorporate three modalities: *playlist*, *tag*, and *metadata*. For the playlist modality, we leverage collaborative information from song co-occurrences by training a Word2Vec-style [30, 8] embedding model on the training set. For the remaining modalities, *tags* represent categorical features (*e.g.,* genre), while *metadata* comprises text-rich descriptions. For simplicity,

we follow Doh et al. (2025) [9] and extract embeddings using a pretrained text encoder [53]. Each embedding is clustered into 1,024 groups via k-means clustering, where the centroid indices serve as the semantic ID tokens. Since there are three modalities, we maintain a latent token vocabulary of size $3! = 6$.

Results. We compare our latent token-based method against six baseline models that utilize fixed modality orders. As shown in Table 5, our method consistently outperforms all baselines, demonstrating the effectiveness of introducing modality-order flexibility via latent tokens. In Figure 5, we observe that the model generates a non-uniform distribution of latent tokens. Notably, the two most frequent permutations (both of which prioritize the playlist modality) correspond to the best-performing fixed-order baselines in Table 5. This suggests that latent tokens with inductive bias can automatically identify more effective modality sequences. Overall, these results validate the potential of anchoring latent tokens to specific inductive biases. While the permutation-based approach discussed here is a straightforward example, we believe many other grounding strategies merit future exploration.

7 Related Work

Generative recommendation. Unlike conventional models that represent each item via learnable embedding vectors [21, 11] or feature-based representations [15, 24], generative recommendation tokenizes each item into a sequence of discrete tokens and frames the recommendation task as a sequence generation problem [34, 57, 61, 5, 20]. Existing literature has largely focused on designing item tokenization algorithms [42, 66, 44, 18, 17, 25, 41, 50] or more effectively leveraging heterogeneous features during the tokenization stage [43, 27, 16, 55, 45, 47, 52, 28]. To better understand the underlying mechanisms of GR, recent studies have investigated its cold-start capabilities [54, 7], scaling behavior [26], and generalization capability [6]. In this work, we contribute to this line by identifying inherent expressiveness limitations stemming from the unique decoding process of GR. Specifically, we demonstrate cases where generative models fail to capture even simple user-item preference and item-item similarity patterns.

Expressiveness of recommendation models. As recommender systems enter the era of deep learning, understanding the expressive power of various architectures has emerged as a critical research direction. Existing literature can be broadly categorized into two primary threads. The first focuses on the expressiveness of the underlying backbone architectures upon which recommendation models are built, such as graph neural networks [51, 31, 38, 1]. The second thread examines the dot-product-based scoring functions, like connecting expressiveness with embedding dimensionality [32, 48, 29, 35]. However, unlike traditional two-tower models, generative recommendation produces predictions by decoding sequences rather than retrieving single vectors. Consequently, the insights derived from retrieval-based paradigms may not hold in this new context. In this work, we study the connections between the sequence decoding process and item probability distributions to analyze the expressive limitations of modern generative recommendation models.

8 Conclusion

In this work, we identify inherent expressiveness limitations in generative recommendation models that arise from their unique autoregressive decoding process. We empirically demonstrate that GR models tend to assign similar probabilities to items that are close in the decoding tree induced by semantic ID tokens. We further provide theoretical evidence that such structural correlations prevent GR models from capturing simple user-item preference (rank reversals, see Section 3.3) and item-item similarity (forced transitivity, see Section 3.4) patterns. To mitigate this issue, we propose a simple yet effective modification to the standard GR framework. By generating latent tokens before the semantic IDs, the decoding tree is reshaped to allow multiple paths with varying tree distances between any pair of items, thereby relaxing the structural constraints imposed by the original fixed decoding tree. Extensive experiments on public datasets validate the effectiveness of our proposed method, leading to an average of 3.45% relative improvement of NDCG@10. We further investigate binding latent tokens with inductive biases. In particular, order-indicating latent tokens help the model identify effective modality orderings, resulting in improved overall performance. In future work, we are interested in developing more effective ways to bind latent tokens with inductive biases.

References

[1] Xuheng Cai, Lianghao Xia, Xubin Ren, and Chao Huang. How expressive are graph neural networks in recommendation? In *CIKM*, pages 173–182, 2023.

[2] Francesco Paolo Cantelli. Sui confini della probabilita. In *Atti del Congresso Internazionale dei Matematici: Bologna del 3 al 10 de settembre di 1928*, pages 47–60, 1929.

[3] Ben Chen, Xian Guo, Siyuan Wang, Zihan Liang, Yue Lv, Yufei Ma, Xinlong Xiao, Bowen Xue, Xuxin Zhang, Ying Yang, Huangyu Dai, Xing Xu, Tong Zhao, Mingcan Peng, Xiaoyang Zheng, Chao Wang, Qihang Zhao, Zhixin Zhai, Yang Zhao, Bochao Liu, Jingshan Lv, Xiao Liang, Yuqing Ding, Jing Chen, Chenyi Lei, Wenwu Ou, Han Li, and Kun Gai. Onesearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search. *arXiv preprint arXiv:2509.03236*, 2025.

[4] Ching-Wei Chen, Paul Lamere, Markus Schedl, and Hamed Zamani. Recsys challenge 2018: automatic music playlist continuation. In *RecSys*, pages 527–528, 2018.

[5] Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*, 2025.

[6] Yijie Ding, Zitian Guo, Jiacheng Li, Letian Peng, Shuai Shao, Wei Shao, Xiaoqiang Luo, Luke Simon, Jingbo Shang, Julian McAuley, and Yupeng Hou. How well does generative recommendation generalize? *arXiv preprint arXiv:2603.19809*, 2026.

[7] Yijie Ding, Jiacheng Li, Julian McAuley, and Yupeng Hou. Inductive generative recommendation via retrieval-based speculation. In *AAAI*, 2026.

[8] Seungheon Doh, Keunwoo Choi, and Juhan Nam. Talkplay: Multimodal music recommendation with large language models. *arXiv preprint arXiv:2502.13713*, 2025.

[9] Seungheon Doh, Keunwoo Choi, and Juhan Nam. Talkplay-tools: Conversational music recommendation with llm tool calling. *arXiv preprint arXiv:2510.01698*, 2025.

[10] Ruining He, Lukasz Heldt, Lichan Hong, Raghunandan H. Keshavan, Shifan Mao, Nikhil Mehta, Zhengyang Su, Alicia Tsai, Yueqi Wang, Shao-Chuan Wang, Xinyang Yi, Lexi Baugher, Baykal Cakici, Ed H. Chi, Cristos Goodrow, Ningren Han, He Ma, Rómer Rosales, Abby Van Soest, Devansh Tandon, Su-Lin Wu, Weilong Yang, and Yilin Zheng. PLUM: adapting pre-trained language models for industrial-scale generative recommendations. *arXiv preprint arXiv:2510.07784*, 2025.

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

[12] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. Learning vector-quantized item representation for transferable sequential recommenders. In *WWW*, pages 1162–1171, 2023.

[13] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

[14] Yupeng Hou, Jiacheng Li, Ashley Shin, Jinsung Jeon, Abhishek Santhanam, Wei Shao, Kaveh Hassani, Ning Yao, and Julian McAuley. Generating long semantic ids in parallel for recommendation. In *KDD*, 2025.

[15] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. Towards universal sequence representation learning for recommender systems. In *KDD*, pages 585–593, 2022.

[16] Yupeng Hou, Jianmo Ni, Zhankui He, Noveen Sachdeva, Wang-Cheng Kang, Ed H. Chi, Julian McAuley, and Derek Zhiyuan Cheng. ActionPiece: Contextually tokenizing action sequences for generative recommendation. In *ICML*, 2025.

[17] Wenyue Hua, Shuyuan Xu, Yingqiang Ge, and Yongfeng Zhang. How to index item ids for recommendation foundation models. In *SIGIR-AP*, 2023.

[18] Bowen Jin, Hansi Zeng, Guoyin Wang, Xiusi Chen, Tianxin Wei, Ruirui Li, Zhengyang Wang, Zheng Li, Yang Li, Hanqing Lu, Suhang Wang, Jiawei Han, and Xianfeng Tang. Language models as semantic indexers. In *ICML*, 2024.

[19] Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvesh Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. Generative recommendation with semantic ids: A practitioner's handbook. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 6420–6425, 2025.

[20] Clark Mingxuan Ju, Tong Zhao, Leonardo Neves, Liam Collins, Bhuvesh Kumar, Jiwen Ren, Lili Zhang, Wenfeng Zhuo, Vincent Zhang, Xiao Bai, et al. Semantic ids for recommender systems at snapchat: Use cases, technical challenges, and design choices. *arXiv preprint arXiv:2604.03949*, 2026.

[21] Wang-Cheng Kang and Julian J. McAuley. Self-attentive sequential recommendation. In *ICDM*, 2018.

[22] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[23] Haven Kim, Yupeng Hou, and Julian McAuley. Fusid: Modality-fused semantic ids for generative music recommendation. *arXiv preprint arXiv:2601.08764*, 2026.

[24] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. Text is all you need: Learning language representations for sequential recommendation. In *KDD*, 2023.

[25] Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. Generative recommender with end-to-end learnable item tokenization. In *SIGIR*, pages 729–739, 2025.

[26] Jingzhe Liu, Liam Collins, Jiliang Tang, Tong Zhao, Neil Shah, and Clark Mingxuan Ju. Understanding generative recommendation with semantic ids from a model-scaling view. *arXiv preprint arXiv:2509.25522*, 2025.

[27] Zihan Liu, Yupeng Hou, and Julian McAuley. Multi-behavior generative recommendation. In *CIKM*, 2024.

[28] Xinchen Luo, Jiangxia Cao, Tianyu Sun, Jinkai Yu, Rui Huang, Wei Yuan, Hezheng Lin, Yichen Zheng, Shiyao Wang, Qigen Hu, et al. Qarm: Quantitative alignment multi-modal recommendation at kuaishou. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 5915–5922, 2025.

[29] Aditya Menon, Sadeep Jayasumana, Ankit Singh Rawat, Seungyeon Kim, Sashank Reddi, and Sanjiv Kumar. In defense of dual-encoders for neural ranking. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15376–15400. PMLR, 17–23 Jul 2022.

[30] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[31] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks, 2021.

[32] Naoto Ohsaka and Riku Togashi. Curse of "low" dimensionality in recommender systems. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 537–547. ACM, July 2023.

[33] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. Contrastive learning for representation degeneration problem in sequential recommendation. In *WSDM*, 2022.

[34] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *NeurIPS*, 2023.

[35] Nils Reimers and Iryna Gurevych. The curse of dense low-dimensional information retrieval for large index sizes. In *ACL*, pages 605–611, 2021.

[36] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[37] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2010.

[38] Yifei Shen, Yongji Wu, Yao Zhang, Caihua Shan, Jun Zhang, B Khaled Letaief, and Dongsheng Li. How powerful is graph convolution for recommendation? In *CIKM*, pages 1619–1629, 2021.

[39] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, 2019.

[40] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Prakash Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. Transformer memory as a differentiable search index. In *NeurIPS*, 2022.

[41] Huanjie Wang, Xinchen Luo, Honghui Bao, Zhang Zixing, Lejian Ren, Yunfan Wu, Hongwei Zhang, Liwei Guan, and Guang Chen. Pit: A dynamic personalized item tokenizer for end-to-end generative recommendation. *arXiv preprint arXiv:2602.08530*, 2026.

[42] Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable tokenizer for llm-based generative recommendation. In *CIKM*, 2024.

[43] Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, and Zhenhua Dong. Eager: Two-stream generative recommender with behavior-semantic collaboration. In *KDD*, page 3245–3254, 2024.

[44] Yidan Wang, Zhaochun Ren, Weiwei Sun, Jiyuan Yang, Zhixiang Liang, Xin Chen, Ruobing Xie, Su Yan, Xu Zhang, Pengjie Ren, Zhumin Chen, and Xin Xin. Enhanced generative recommendation via content and collaboration integration. *arXiv preprint arXiv:2403.18480*, 2024.

[45] Yuhao Wang, Junwei Pan, Xinhang Li, Maolin Wang, Yuan Wang, Yue Liu, Dapeng Liu, Jie Jiang, and Xiangyu Zhao. Empowering large language model for sequential recommendation via multimodal embeddings and semantic ids. In *CIKM*, pages 3209–3219, 2025.

[46] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. A neural corpus indexer for document retrieval. In *NeurIPS*, 2022.

[47] Zhipeng Wei, Kuo Cai, Junda She, Jie Chen, Minghao Chen, Yang Zeng, Qiang Luo, Wencong Zeng, Ruiming Tang, Kun Gai, et al. Oneloc: Geo-aware generative recommender systems for local life service. *arXiv preprint arXiv:2508.14646*, 2025.

[48] Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. On the theoretical limitations of embedding-based retrieval, 2025.

[49] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *ICDE*, 2022.

[50] Zhouhang Xie, Bo Peng, Zhankui He, Ziqi Chen, Alice Han, Isabella Ye, Benjamin Coleman, Noveen Sachdeva, Fernando Pereira, Julian McAuley, et al. Agentictagger: Structured item representation for recommendation with llm agents. *arXiv preprint arXiv:2602.05945*, 2026.

[51] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks?, 2019.

[52] Yi Xu, Moyu Zhang, Chenxuan Li, Zhihao Liao, Haibo Xing, Hao Deng, Jinxin Hu, Yu Zhang, Xiaoyi Zeng, and Jing Zhang. Mmq: Multimodal mixture-of-quantization tokenization for semantic id generation and user behavioral adaptation. *arXiv preprint arXiv:2508.15281*, 2025.

[53] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[54] Liu Yang, Fabian Paischer, Kaveh Hassani, Jiacheng Li, Shuai Shao, Zhang Gabriel Li, Yun He, Xue Feng, Nima Noorshams, Sem Park, Bo Long, Robert D Nowak, Xiaoli Gao, and Hamid Eghbalzadeh. Unifying generative and dense retrieval for sequential recommendation. *arXiv preprint arXiv:2411.18814*, 2024.

[55] Wencai Ye, Mingjie Sun, Shaoyun Shi, Peng Wang, Wenjin Wu, and Peng Jiang. Das: Dual-aligned semantic ids empowered industrial recommender system. In *CIKM*, pages 6217–6224, 2025.

[56] Jianyang Zhai, Zi-Feng Mai, Chang-Dong Wang, Feidiao Yang, Xiawu Zheng, Hui Li, and Yonghong Tian. Multimodal quantitative language for generative recommendation. *arXiv preprint arXiv:2504.05314*, 2025.

[57] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, Yinghai Lu, and Yu Shi. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *ICML*, 2024.

[58] Fuwei Zhang, Xiaoyu Liu, Dongbo Xi, Jishen Yin, Huan Chen, Peng Yan, Fuzhen Zhuang, and Zhao Zhang. Multi-aspect cross-modal quantization for generative recommendation, 2025.

[59] Ruohan Zhang, Jiacheng Li, Julian McAuley, and Yupeng Hou. Purely semantic indexing for LLM-based generative recommendation and retrieval. *arXiv preprint arXiv:2509.16446*, 2025.

[60] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI*, 2019.

[61] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *ICDE*, 2024.

[62] Bowen Zheng, Enze Liu, Zhongfu Chen, Zhongrui Ma, Yue Wang, Wayne Xin Zhao, and Ji-Rong Wen. Pre-training generative recommender with multi-identifier item tokenization. *arXiv preprint arXiv:2504.04400*, 2025.

[63] Qiyong Zhong, Jiajie Su, Yunshan Ma, Julian McAuley, and Yupeng Hou. Pctx: Tokenizing personalized context for generative recommendation. *arXiv preprint arXiv:2510.21276*, 2025.

[64] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*, 2020.

[65] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced MLP is all you need for sequential recommendation. In *TheWebConf*, 2022.

[66] Jieming Zhu, Mengqun Jin, Qijiong Liu, Zexuan Qiu, Zhenhua Dong, and Xiu Li. Cost: Contrastive quantization based semantic tokenization for generative recommendation. In *RecSys*, 2024.

[67] Jing Zhu, Mingxuan Ju, Yozen Liu, Danai Koutra, Neil Shah, and Tong Zhao. Beyond unimodal boundaries: Generative recommendation with multimodal semantics. *arXiv preprint arXiv:2503.23333*, 2025.

Table 6: Notations and explanations.

Notation	Explanation
i, i_t	item, item at time t
$(i_1, i_2, \ldots, i_{t-1})$	historical interaction sequence
u	user's historical interactions represented as semantic IDs
$c^{(j)}$	the j-th token in a semantic ID
$(c^{(1)}, c^{(2)}, \ldots, c^{(m)})$	semantic ID, a sequence of discrete tokens indexing an item
$c_t^{(1:j-1)}$	Shorthand for the token sequence $(c_t^{(1)}, c_t^{(2)}, \ldots, c_t^{(j-1)})$
m	the length (depth) of semantic IDs
$\mathcal{C}^{(j)}$	vocabulary for the j-th position in semantic IDs
M	size of the token vocabulary $\mathcal{C}^{(j)}$
$\mathbb{P}(i_t \mid u)$	probability of item i_t given user history u
$\mathbb{P}(c_t^{(j)} \mid c_t^{(1)}, \ldots, c_t^{(j-1)}, u)$	conditional probability of generating the j-th token
$\mathcal{T}$	decoding tree induced by valid semantic IDs
$d_{\mathcal{T}}(i, i')$	tree distance between item i and i' in the decoding tree $\mathcal{T}$
$\rho(i, i')$	Pearson correlation coefficient between generation probabilities of i and i'
σ^2	Variance of item generation probability $\mathbb{P}(i \mid u)$ across users
$\mathcal{R}(i, i')$	Rank reversal event between items i and i' (see Section 3.3)
μ	Expected difference in generation probabilities $\lvert\mathbb{E}[\mathbb{P}(i \mid u) - \mathbb{P}(i' \mid u)]\rvert$
M'	Number of latent tokens in Latte

Table 7: Statistics of the datasets. "Avg. t" denotes the average length of user interaction sequences.

Dataset	#Users	#Items	#Interactions	Avg. t
Instruments	57,439	24,587	511,836	8.91
Scientific	50,985	25,848	412,947	8.10
Games	94,762	25,612	814,586	8.60

A Notation

We summarize the notations used throughout the paper in Table 6 for easy reference.

B Proofs for Section 3.3 (User-Item Preference Limitation)

In this section, we provide the detailed proofs for the claims made in Section 3.3 regarding the limitation of autoregressive SID generation in modeling diverse user preferences.

B.1 Formal Setup

Let u be a random user drawn from the user population $\mathcal{U}$. For a fixed item pair (i, i'), we consider the generation probabilities as random variables:

$$X \triangleq \mathbb{P}(i \mid u), \qquad Y \triangleq \mathbb{P}(i' \mid u), \qquad D \triangleq X - Y.$$

Let $\mu \triangleq \mathbb{E}[D]$ be the expected difference in generation probability (*e.g.,* reflecting global trend difference) and $\rho(i, i') \triangleq \mathrm{Corr}(X, Y)$ be the Pearson correlation coefficient between the generation probabilities of i and i' (as illustrated in Figure 2).

For two independent users $u, u' \overset{\text{iid}}{\sim} \mathcal{U}$, we define the rank reversal event $\mathcal{R}(i, i')$ as the event where the two users have opposite relative preferences for items i and i'. Formally:

$$\mathcal{R}(i, i') \triangleq \Big\{\mathbb{P}(i \mid u) > \mathbb{P}(i' \mid u),\, \mathbb{P}(i \mid u') < \mathbb{P}(i' \mid u')\Big\} \cup \Big\{\mathbb{P}(i \mid u) < \mathbb{P}(i' \mid u),\, \mathbb{P}(i \mid u') > \mathbb{P}(i' \mid u')\Big\}.$$

In terms of the difference variable D, let $D_u = \mathbb{P}(i \mid u) - \mathbb{P}(i' \mid u)$ and $D_{u'} = \mathbb{P}(i \mid u') - \mathbb{P}(i' \mid u')$. Then $\mathcal{R}(i, i') = \{D_u > 0,\, D_{u'} < 0\} \cup \{D_u < 0,\, D_{u'} > 0\}$.

Assumption B.1 (Comparable scale across users)**.** For the fixed pair (i, i'), we assume the variance of generation probabilities is comparable: $\mathrm{Var}(X) = \mathrm{Var}(Y) = \sigma^2$ for some $\sigma^2 > 0$.

B.2 Proof of Rank Reversal Bound

Lemma B.2 (Reversal probability decomposition)**.** *Under the independence of u and u',*

$$\mathbb{P}\big(\mathcal{R}(i,i')\big) = 2\,\mathbb{P}(D>0)\,\mathbb{P}(D<0).$$

Proof. By definition and independence:

$$\begin{aligned}\mathbb{P}\big(\mathcal{R}(i,i')\big) &= \mathbb{P}(D_u>0,\, D_{u'}<0) + \mathbb{P}(D_u<0,\, D_{u'}>0)\\ &= \mathbb{P}(D_u>0)\mathbb{P}(D_{u'}<0) + \mathbb{P}(D_u<0)\mathbb{P}(D_{u'}>0).\end{aligned}$$

Since u and u' are i.i.d., D_u and $D_{u'}$ are identically distributed as D. Thus,

$$\mathbb{P}\big(\mathcal{R}(i,i')\big) = 2\,\mathbb{P}(D>0)\mathbb{P}(D<0).$$

□

Lemma B.3 (Gap variance)**.** *Under Assumption B.1, the variance of the gap D is:*

$$\mathrm{Var}(D) = 2\sigma^2\big(1-\rho(i,i')\big).$$

Proof. Using $\mathrm{Var}(X)=\mathrm{Var}(Y)=\sigma^2$ and $\mathrm{Cov}(X,Y)=\rho(i,i')\,\sigma^2$, we have:

$$\mathrm{Var}(D)=\mathrm{Var}(X-Y)=\mathrm{Var}(X)+\mathrm{Var}(Y)-2\,\mathrm{Cov}(X,Y)=2\sigma^2-2\rho(i,i')\sigma^2=2\sigma^2(1-\rho(i,i')).$$

□

Theorem B.4 (High correlation implies few cross-user rank reversals)**.** *Under Assumption B.1, let $\mu=\mathbb{E}[D]$ and $\rho=\rho(i,i')$. Then:*

$$\mathbb{P}\big(\mathcal{R}(i,i')\big) \le \frac{4\sigma^2(1-\rho)}{\mu^2+2\sigma^2(1-\rho)}.$$

Proof. From Lemma B.2, $\mathbb{P}(\mathcal{R})=2\mathbb{P}(D>0)\mathbb{P}(D<0)\le 2\min\{\mathbb{P}(D>0),\mathbb{P}(D<0)\}$. Without loss of generality, assume $\mu\ge 0$. Then $\mathbb{P}(D<0)$ is the tail probability.

$$\mathbb{P}\big(\mathcal{R}(i,i')\big) \le 2\,\mathbb{P}(D\le 0) = 2\,\mathbb{P}(D-\mu\le-\mu).$$

By Cantelli's inequality [2]:

$$\mathbb{P}(D-\mu\le-\mu)\le\frac{\mathrm{Var}(D)}{\mathrm{Var}(D)+\mu^2}.$$

Calculating the bound:

$$\mathbb{P}\big(\mathcal{R}(i,i')\big)\le\frac{2\,\mathrm{Var}(D)}{\mathrm{Var}(D)+\mu^2}.$$

Substituting $\mathrm{Var}(D)=2\sigma^2(1-\rho)$ from Lemma B.3:

$$\mathbb{P}\big(\mathcal{R}(i,i')\big)\le\frac{4\sigma^2(1-\rho)}{\mu^2+2\sigma^2(1-\rho)}.$$

□

C Proofs for Section 3.4 (Item-Item Similarity Limitation)

In this section, we provide the theoretical justification for the limitation on item-item similarity modeling discussed in Section 3.4.

C.1 Tree Distance as an Ultrametric

Lemma C.1 (Ultrametric inequality)**.** *The tree distance $d_{\mathcal{T}}$ defined in Definition 3.2 satisfies the ultrametric inequality. For any three items $i_1, i_2, i_3 \in \mathcal{I}$:*

$$d_{\mathcal{T}}(i_1, i_3) \leq \max\left(d_{\mathcal{T}}(i_1, i_2), d_{\mathcal{T}}(i_2, i_3)\right). \tag{7}$$

Proof. Let v_1, v_2, v_3 be the leaf nodes corresponding to items i_1, i_2, i_3 in the decoding tree $\mathcal{T}$. The tree distance $d_{\mathcal{T}}(i, i')$ is determined by the depth of the lowest common ancestor (LCA) of the two leaves. specifically, if the tree has depth m, then $d_{\mathcal{T}}(i, i') = 2(m - \text{depth}(\text{LCA}(i, i')))$.

Consider the computed LCAs for the pairs (i_1, i_2) and (i_2, i_3). In any tree structure, the path intersection property dictates that:

$$\text{depth}(\text{LCA}(i_1, i_3)) \geq \min(\text{depth}(\text{LCA}(i_1, i_2)), \text{depth}(\text{LCA}(i_2, i_3))).$$

Substituting the distance definition into this inequality:

$$\begin{aligned}
2(m - \frac{d_{\mathcal{T}}(i_1, i_3)}{2}) &\geq \min\left(2(m - \frac{d_{\mathcal{T}}(i_1, i_2)}{2}), 2(m - \frac{d_{\mathcal{T}}(i_2, i_3)}{2})\right) \\
m - \frac{d_{\mathcal{T}}(i_1, i_3)}{2} &\geq m - \max\left(\frac{d_{\mathcal{T}}(i_1, i_2)}{2}, \frac{d_{\mathcal{T}}(i_2, i_3)}{2}\right) \\
\frac{d_{\mathcal{T}}(i_1, i_3)}{2} &\leq \max\left(\frac{d_{\mathcal{T}}(i_1, i_2)}{2}, \frac{d_{\mathcal{T}}(i_2, i_3)}{2}\right).
\end{aligned}$$

Multiply by 2, we obtain:

$$d_{\mathcal{T}}(i_1, i_3) \leq \max(d_{\mathcal{T}}(i_1, i_2), d_{\mathcal{T}}(i_2, i_3)).$$

□

C.2 Correlation and Item Similarity

Before proving the main theorem, we establish the connection between the Pearson correlation coefficient $\rho(i, i')$ and item-item similarity in collaborative filtering.

In collaborative filtering [36, 37], item-item similarity is commonly modeled as the inner product of item representations derived from the user-item preference matrix. Let $\mathbf{P} \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{U}|}$ denote the matrix where entry $P_{i,u} = \mathbb{P}(i \mid u)$ represents the generation probability of item i for user u. Each row $\mathbf{P}_{i,:}$ captures how different users respond to item i.

The Pearson correlation coefficient between items i and i' is defined as:

$$\rho(i, i') = \text{Corr}(\mathbf{P}_{i,:}, \mathbf{P}_{i',:}) = \frac{\mathbb{E}_u[(P_{i,u} - \bar{P}_i)(P_{i',u} - \bar{P}_{i'})]}{\sigma_i \sigma_{i'}},$$

where $\bar{P}_i = \mathbb{E}_u[P_{i,u}]$ is the mean generation probability and σ_i is the standard deviation. This correlation coefficient is mathematically equivalent to the inner product of the normalized preference vectors:

$$\rho(i, i') = \left\langle \frac{\mathbf{P}_{i,:} - \bar{P}_i}{\sigma_i}, \frac{\mathbf{P}_{i',:} - \bar{P}_{i'}}{\sigma_{i'}} \right\rangle.$$

This formulation reveals that high correlation $\rho(i, i') \approx 1$ indicates that items i and i' have similar user preference patterns, which is connected to item-item similarity in collaborative filtering. Conversely, low or negative correlation suggests dissimilar preference patterns. This connection allows us to interpret the structural constraints imposed by the tree distance (via Assumption 3.3) as constraints on the model's ability to capture flexible item-item similarities.

C.3 Proof of Limitation on Intransitive Similarity

Here we provide the proof for Theorem 3.5 in Section 3.4, building on the established relationship between correlation and item similarity.

Table 8: Best hyperparameters for the base model PSID across three datasets.

Hyperparameter	Instruments	Scientific	Games
Beam size	500	500	500
Learning rate	3×10^{-3}	1×10^{-3}	3×10^{-3}

Table 9: Best hyperparameters for our model Latte across three datasets.

Hyperparameter	Instruments	Scientific	Games
Number of latent tokens M'	4	8	8
Beam size	500	500	500
Learning rate	3×10^{-3}	3×10^{-3}	3×10^{-3}

Proof of Theorem 3.5. By Assumption 3.3, the correlation $\rho(i, i')$ is monotonically related to the tree distance $d_{\mathcal{T}}(i, i')$. Assume the model successfully captures the similarities for the first two pairs:

1. $i_1 \sim i_2 \implies \rho(i_1, i_2) > \tau \implies d_{\mathcal{T}}(i_1, i_2) \leq \delta$.
2. $i_2 \sim i_3 \implies \rho(i_2, i_3) > \tau \implies d_{\mathcal{T}}(i_2, i_3) \leq \delta$.

Using the ultrametric property from Lemma C.1:

$$d_{\mathcal{T}}(i_1, i_3) \leq \max(d_{\mathcal{T}}(i_1, i_2), d_{\mathcal{T}}(i_2, i_3)) \leq \max(\delta, \delta) = \delta.$$

Since $d_{\mathcal{T}}(i_1, i_3) \leq \delta$, by the bidirectional relationship between correlation and tree distance, we have:

$$\rho(i_1, i_3) > \tau.$$

This implies i_1 and i_3 must exhibit a high level of similarity (correlation $> \tau$), making it impossible for the model to treat them as dissimilar (which would require $\rho \leq \tau$). Thus, the tree structure imposes a transitivity constraint on the learned similarities. □

C.4 Expressiveness of Latte

We now show that Latte relaxes the rank-reversal constraint in Theorem 3.4. Recall that for a pair of items (i, i'), the rank-reversal probability in standard GR is bounded by

$$\mathbb{P}(\mathcal{R}(i, i')) \leq B(\rho) \triangleq \frac{4\sigma^2(1-\rho)}{\mu^2 + 2\sigma^2(1-\rho)}, \tag{8}$$

where $\rho = \rho(i, i')$ denotes the correlation between the generation probabilities of i and i' across users. As discussed in Section 3.2, structurally close items in standard GR tend to have large ρ, which makes $B(\rho)$ small and suppresses cross-user rank reversals.

Proposition C.2 (Latte relaxes the rank-reversal constraint)**.** *Consider a pair of structurally close items (i, i') with correlation ρ in standard GR. Suppose Latte uses M' latent tokens and the dominant latent token for each item is approximately uniformly distributed,* i.e.,

$$\mathbb{P}\big(\ell^*(i, u) = \ell^*(i', u)\big) = \frac{1}{M'}.$$

Let ρ_{low} denote the correlation between two items whose generation paths diverge at the latent-token level, with $\rho_{\text{low}} < \rho$. Then the effective correlation between i and i' under Latte satisfies

$$\rho_{\text{Latte}} \approx \frac{1}{M'}\rho + \left(1 - \frac{1}{M'}\right)\rho_{\text{low}} < \rho, \tag{9}$$

for any $M' > 1$. Consequently, when $\mu > 0$, Latte loosens the rank-reversal bound:

$$B(\rho_{\text{Latte}}) > B(\rho). \tag{10}$$

Proof. In standard GR, the tree distance $d_{\mathcal{T}}(i, i')$ between two items is fixed. Thus, for structurally close items, their generation probabilities share a long prefix in the decoding tree, inducing a high correlation ρ across users.

In Latte, the effective generation path also depends on the latent token selected for each item and user. Let

$$\ell^*(i, u) = \arg\max_{\ell} \mathbb{P}(\ell \mid u)\mathbb{P}(i \mid \ell, u)$$

denote the dominant latent token for generating item i given user u. For the pair (i, i'), there are two cases.

First, if $\ell^*(i, u) = \ell^*(i', u)$, the two items are generated under the same latent path. In this case, their effective tree distance remains the original distance $d_{\mathcal{T}}(i, i')$, and their correlation remains close to the standard GR correlation ρ.

Second, if $\ell^*(i, u) \neq \ell^*(i', u)$, the two generation paths diverge immediately after the root through different latent tokens. Their effective distance becomes

$$d_{\text{eff}}(i, i'; u) = 2(m + 1),$$

which is the maximum distance in the latent-augmented decoding tree. By Theorem 3.3, larger tree distance corresponds to lower correlation in generation probabilities. We denote this lower correlation by ρ_{low}, where $\rho_{\text{low}} < \rho$.

Since the latent tokens are approximately uniformly assigned, the probability that two items use the same dominant latent token is $1/M'$, while the probability that they use different dominant latent tokens is $1 - 1/M'$. Therefore, the expected effective correlation under Latte can be written as

$$\rho_{\text{Latte}} \approx \frac{1}{M'}\rho + \left(1 - \frac{1}{M'}\right)\rho_{\text{low}}.$$

Because $M' > 1$ and $\rho_{\text{low}} < \rho$, we have

$$\begin{aligned}\rho_{\text{Latte}} - \rho &= \frac{1}{M'}\rho + \left(1 - \frac{1}{M'}\right)\rho_{\text{low}} - \rho \\ &= \left(1 - \frac{1}{M'}\right)(\rho_{\text{low}} - \rho) < 0. \end{aligned} \tag{11}$$

Thus, $\rho_{\text{Latte}} < \rho$.

It remains to show that this lower correlation gives a looser rank-reversal bound. Taking the derivative of $B(x)$ in Equation (8), we obtain

$$\begin{aligned} B'(x) &= \frac{\partial}{\partial x}\frac{4\sigma^2(1-x)}{\mu^2 + 2\sigma^2(1-x)} \\ &= -\frac{4\sigma^2\mu^2}{\left(\mu^2 + 2\sigma^2(1-x)\right)^2}. \end{aligned} \tag{12}$$

When $\mu > 0$, we have $B'(x) < 0$, so $B(x)$ is strictly decreasing in x. Since $\rho_{\text{Latte}} < \rho$, it follows that

$$B(\rho_{\text{Latte}}) > B(\rho).$$

Therefore, Latte relaxes the rank-reversal constraint imposed by high structural correlation. Intuitively, by allowing structurally close items to take different latent paths, Latte reduces their effective correlation and gives the model more flexibility to express user-specific rank reversals. □

Uniform latent-token generation. In the above analysis, we assume that the latent tokens are approximately uniformly selected during generation. This assumption is motivated by our training design, where latent tokens are uniformly sampled and prepended before the semantic IDs. Under this assumption, when the latent vocabulary size is M', each latent token is selected with probability approximately $1/M'$. Therefore, for two items i and i', the probability that they share the same dominant latent path is approximately $1/M'$, while the probability that they diverge at the latent-token level is approximately $1 - 1/M'$. For the Games dataset with $M' = 8$, the expected generation probability of each latent token is thus $1/8 = 0.125$, as shown in Table 10.

Table 10: Latent-token generation probabilities on the Games dataset under the uniform latent-token generation assumption.

Latent Token	Generation Probability
ℓ_1	0.1244
ℓ_2	0.1253
ℓ_3	0.1248
ℓ_4	0.1243
ℓ_5	0.1259
ℓ_6	0.1254
ℓ_7	0.1251
ℓ_8	0.1248

D Implementation Details

Compared models. We compare Latte with the following baselines: (1) traditional sequential recommendation models that primarily based on item IDs (and feature IDs), including GRU4Rec [11], BERT4Rec [39], SASRec [21], FMLP-Rec [65], HSTU [57], FDSA [60], and S^3-Rec [64]; and (2) generative recommendation models based on semantic IDs, including TIGER [34], LETTER [42], ActionPiece [16], and PSID [59].

We adhere to the exact same experimental settings and directly adopt the reported results for most baselines from prior works [62, 63]. For the base model, PSID [59], we refer to the officially released code[1] to implement the model.

Evaluation. We evaluate all models using the widely adopted Recall@K (R@K) and NDCG@K (N@K) metrics, with $K \in \{5, 10\}$, following prior works [34, 62]. We tune hyperparameters based on NDCG@10 performance on the validation set and select the best-performing checkpoint for testing.

Latte. We build Latte upon our base model, PSID [59]. Specifically, we adopt the architecture of the representative model TIGER [34], maintaining the same configuration: a T5-style encoder-decoder, 4 stacked Transformer blocks in both the encoder and decoder, and an embedding dimension of 128. Regarding tokenization, we follow the recommendations of Ju et al. (2025) [19] to primarily use RQ K-Means, though we also experiment with other tokenizers such as RQ-VAE and OPQ (see Table 3). To prevent semantic ID conflicts, we employ the ESM algorithm as proposed in the PSID paper [59]. The resulting semantic ID length is $m = 3$ with a vocabulary size of $M = 256$ for each token position. The number of introduced latent tokens is tuned from the set $\{2, 4, 8\}$ based on validation performance. For the Latte results reported in Table 1, we use $\max$ as the aggregation method.

Training and inference. We tune the learning rate for both the base model PSID and our model, Latte, from the set $\{1 \times 10^{-3}, 3 \times 10^{-3}\}$. We use a weight decay of 0.05. Models are trained for up to 150 epochs, employing early stopping with a patience of 20 epochs. Hyperparameters are selected based on the best NDCG@10 performance on the validation set, and the best-performing checkpoint is used for testing. During inference, we tune the beam size from $\{50, 100, 500\}$, also based on validation performance. All experiments were conducted on a single NVIDIA A6000 GPU.

Best hyperparameters for both PSID and Latte across the three datasets are summarized in Tables 8 and 9.

E Discussion

Why should semantically similar items not always exhibit correlated generation probabilities? We emphasize that semantic similarity and correlated generation probabilities are related but distinct. Assigning similar semantic IDs to semantically similar items is a common design choice in current GR models, as it allows related items to share token prefixes. However, this design choice becomes limiting when shared prefixes force semantically similar items to have highly correlated generation

[1] `https://github.com/wangshanyw/PurelySemanticIndexing`

Table 11: Statistics of structurally coupled items under PSID with RQ K-Means tokenization. We report the average number of items within tree distance 2 and 4 from each target item, as well as the ratio of test instances whose target item has at least one item at tree distance 2.

Dataset	#Items at Dist. 2	#Items at Dist. 4	Ratio with Dist.-2 Items
Instruments	6.29	142.02	83.98%
Scientific	7.39	153.97	85.25%
Games	6.98	179.41	84.17%

probabilities across users. In practice, users may have opposite preferences toward semantically similar items. For example, in the Games dataset, "Pokémon Scarlet (Switch)" and "Pokémon Sun (3DS)" have a tree distance of 2, meaning that only their last semantic ID token differs. While some users may prefer the newer game and favor Scarlet, others may prefer the older style and favor Sun. Standard GR models struggle to represent such ranking reversals, as discussed in Section 3.3, and tend to assign one item a consistently higher probability than the other for most users. This behavior conflicts with the personalized nature of recommendation, where even semantically similar items should be distinguishable based on user-specific preferences.

How significant is the identified issue? To assess how frequently the identified structural coupling occurs in practice, we analyze the base PSID model with RQ K-Means tokenization, where each item is represented by three tokens. For each target item in the test set, we count the average number of other items within tree distance 2 and 4. As shown in Table 11, each target item is associated with around 6–7 closely coupled items at tree distance 2 and 140–180 items at tree distance 4. We further compute the ratio of test instances whose target item has at least one other item at tree distance 2. The ratio is consistently above 80% across all datasets, indicating that the coupling effect is not a rare corner case, but a pervasive structural issue in autoregressive SID generation.

Is the latent token design necessary or optimal? We do not claim that the latent token design in Latte is necessary or optimal. Rather, our goal is to analyze the expressive limitations of current GR models and show that these limitations can be alleviated by relaxing the rigid decoding tree structure. Latte is one simple instantiation of this idea: by introducing an additional latent token before the semantic ID tokens, it provides multiple latent-conditioned generation paths and therefore reduces the structural coupling induced by a single fixed tree. We view Latte as an initial attempt rather than a final solution. We hope this analysis can motivate future work on more expressive and efficient designs for semantic ID-based generative recommendation.

When does Latte degrade to the base model? Latte alleviates prefix coupling by introducing multiple latent-conditioned generation paths. Although two items remain coupled when they are generated under the same latent token, different latent tokens do not simply reweight the same shared prefix probability. For example, given two latent tokens l_A and l_B, the probabilities of generating the same semantic prefix are

$$P(l_A \mid x)P(\text{prefix} \mid x, l_A),$$
$$P(l_B \mid x)P(\text{prefix} \mid x, l_B).$$

Even ignoring the path weights $P(l_A \mid x)$ and $P(l_B \mid x)$, the conditional prefix probabilities can differ because they are conditioned on different latent tokens, i.e., $P(\text{prefix} \mid x, l_A) \neq P(\text{prefix} \mid x, l_B)$ in general. Thus, Latte provides additional flexibility beyond assigning different weights to the same prefix. It would reduce to the base model **only in the extreme case** where different latent tokens induce identical representations or identical conditional generation distributions.

Does Latte introduce significant inference overhead? Latte does not require $|\mathcal{L}|$ independent beam searches over latent tokens. Instead, it uses a single beam search process, where the model first predicts a latent token and then continues generating the semantic ID tokens. Therefore, introducing latent tokens only adds one extra decoding step, rather than multiplying the inference cost by the number of latent tokens. In practice, the aggregation is performed over the candidates retained by beam search, rather than over all possible latent tokens. To quantify this overhead, we compare the

Table 12: Inference time for one epoch on the Instruments dataset.

Model	Time
TIGER	98s
PSID	66s
Latte ($M' = 4$)	76s
Latte ($M' = 8$)	76s
Latte ($M' = 16$)	78s

Table 13: Comparison with noise-based data augmentation baselines. We report NDCG@10 results.

Model	Instruments	Scientific	Games
PSID	0.0325	0.0235	0.0500
Dropout	0.0322	0.0239	0.0499
Swap	0.0300	0.0208	0.0462
Latte	**0.0331**	**0.0249**	**0.0515**

inference time for one epoch on the Instruments dataset. TIGER uses 4 semantic ID tokens per item, PSID uses 3 semantic ID tokens per item, and Latte uses 1 latent token followed by 3 semantic ID tokens. As shown in Table 12, Latte moderately increases the inference cost compared with its base model PSID, but remains more efficient than TIGER.

Does Latte mainly improve performance by introducing noise? To examine whether Latte's improvement mainly comes from a noise-like regularization effect, we compare it with two data augmentation baselines: Dropout [33] and Swap [49]. During each training epoch, these baselines randomly drop or swap a certain ratio of tokens in the input sequence, with the ratio tuned from $\{0.1, 0.2, 0.3\}$. As shown in Table 13, these augmentation methods can sometimes improve over the base PSID model, but they remain consistently worse than Latte. We further compute Kendall's rank correlation between tree distance and item-item similarity, following the analysis in Table 2. As shown in Table 14, only Latte substantially improves the correlation, indicating that its gains do not merely come from injecting noise, but from alleviating the structural probability coupling discussed in Section 3.

F Limitations

Latte is proposed as a simple example to alleviate the expressive limitations discussed in this paper, rather than as an optimal solution. It still has several known limitations. First, Latte introduces additional inference cost, since it requires one extra decoding step to generate the latent token before the original semantic ID tokens. Although this overhead is moderate in our experiments, it may still matter in latency-sensitive recommendation systems. Second, Latte may degenerate to the base model in extreme cases where different learned latent token embeddings become nearly identical, causing different latent-conditioned paths to induce similar generation distributions. Third, the latent token design increases the flexibility of the decoding process but does not completely remove the structural constraints imposed by semantic ID generation. Items generated under the same latent-conditioned path may still exhibit prefix coupling. Despite these limitations, they do not affect our main findings on the expressive limitations of current GR models, including rank reversal (Section 3.3) and forced transitivity (Section 3.4). We hope these findings motivate future work on more expressive and efficient designs for semantic ID-based generative recommendation.

G Societal Impacts

This work aims to improve the expressiveness of generative recommendation models in capturing user-item preferences and item-item relationships. As our contribution focuses on the modeling paradigm rather than a specific application domain, dataset, or deployment setting, we do not identify additional societal impacts beyond those generally associated with the development and deployment of recommender systems. In practice, such systems may influence information access, user behavior,

Table 14: Kendall's rank correlation between tree distance and item-item similarity for noise-based augmentation baselines. Less negative values indicate weaker structural coupling.

Model	Instruments	Scientific	Games
PSID	-0.6225	-0.4611	-0.6072
Dropout	-0.6248	-0.4687	-0.6025
Swap	-0.6277	-0.4887	-0.6096
Latte	**-0.6030**	**-0.4451**	**-0.5958**

and content exposure, and should therefore be deployed with appropriate consideration of fairness, transparency, privacy, and potential feedback loops.