

LLaDA-Rec: Discrete Diffusion for Parallel Semantic ID Generation in Generative Recommendation

Teng Shi¹, Chenglei Shen¹, Weijie Yu^{2*}, Shen Nie¹,
Chongxuan Li¹, Xiao Zhang¹, Ming He³, Yan Han³, Jun Xu¹

¹Gaoling School of Artificial Intelligence, Renmin University of China, China

²School of Artificial Intelligence and Data Science,

University of International Business and Economics, China

³AI Lab at Lenovo Research, Lenovo Group Limited, China

{shiteng, chengleishen9, nieshen, chongxuanli, zhangx89, junxu}@ruc.edu.cn

yu@uibe.edu.cn, heming01@foxmail.com, hanyan7217@163.com

Abstract

Generative recommendation represents items as semantic identifiers (SIDs), i.e., sequences of discrete tokens, and typically predicts the next item autoregressively. Despite their strong performance, such methods suffer from two limitations: (1) *unidirectional constraints*, where causal attention enforces left-to-right dependencies and limits global semantic modeling; and (2) *error accumulation*, where early mistakes propagate to subsequent tokens and hurt recommendation accuracy. To address these issues, we propose **LLaDA-Rec**, a *discrete diffusion* framework that reformulates next-item recommendation as *parallel SID generation*. By combining bidirectional attention with an adaptive generation order, LLaDA-Rec captures richer inter-item and intra-item dependencies while reducing error accumulation. Our framework includes three key components: (1) a parallel tokenization mechanism for diffusion-compatible SIDs; (2) masking strategies at both the user-history and next-item levels; and (3) an adapted beam search algorithm for adaptive-order discrete diffusion decoding. Experiments on three public datasets show that LLaDA-Rec consistently outperforms both conventional ID-based recommenders and strong generative baselines, highlighting discrete diffusion as a promising paradigm for generative recommendation. Codes are available at <https://github.com/TengShi-RUC/LLaDA-Rec>.

1 Introduction

Recent advances in generative language models (Achiam et al., 2023; Zhao et al., 2023; Liu et al., 2024) have inspired growing interest in applying the generative paradigm to recommendation (Rajput et al., 2023; Zheng et al., 2024; Hou et al., 2025; Zhou et al., 2025). Unlike traditional discriminative recommenders (Kang and McAuley, 2018; Sun et al., 2019; Zhou et al., 2022), generative recommendation formulates recommendation

as a sequence-to-sequence task, generating the next target item autoregressively.

Generative recommendation typically consists of two components: item tokenization and autoregressive generation. In tokenization, each item is mapped to a semantic ID (SID), a sequence of discrete tokens, using methods such as hierarchical clustering (Si et al., 2024) or vector quantization (e.g., RQ-VAE (Rajput et al., 2023; Zheng et al., 2024)). The tokenized user history is then fed into a generative model, which predicts the next item’s tokens autoregressively.

Despite their effectiveness, existing methods still have two key limitations: (1) **Unidirectional Constraint**. Most approaches use autoregressive architectures that generate item tokens left to right, limiting their ability to capture global relationships among all tokens defining an item (Gu et al., 2022) and thus reducing semantic coherence. (2) **Error Accumulation**. During inference, each token is generated based on previously sampled tokens, unlike training with teacher forcing (Raffel et al., 2020). As a result, early errors cannot be corrected and may propagate to subsequent tokens, amplifying their impact (Lin et al., 2025). Figure 1 illustrates these issues.

To address these limitations, we explore a new generative recommendation framework based on discrete diffusion (Li et al., 2025a; Nie et al., 2025; Zhu et al., 2025). During training, the model applies a forward token-masking process and a reverse denoising reconstruction objective, with a bidirectional Transformer predicting masked tokens. This design offers two main advantages: (1) **Bidirectional Understanding**, which improves contextual modeling and captures global relationships among tokens; and (2) **Adaptive Generation Order**, where all tokens are predicted in parallel and low-confidence ones are iteratively re-masked and refined, prioritizing higher-confidence (i.e., easier) tokens and mitigating error accumulation.

*Corresponding author

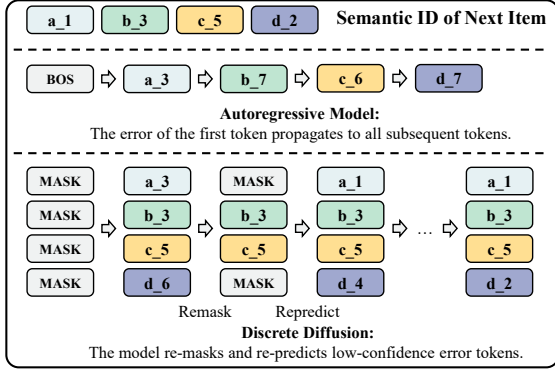


Figure 1: Illustration of the advantages of discrete diffusion over autoregressive generation. Autoregressive models propagate early token errors to later positions, whereas discrete diffusion predicts masked tokens in parallel and iteratively re-masks low-confidence tokens, yielding more accurate results.

Overall, discrete diffusion provides a compelling solution to the limitations of prior generative recommendation methods. Although diffusion models inherently assume fixed-length generation (Nie et al., 2025), this aligns naturally with recommendation, where each item is represented by a fixed number of semantic IDs, making diffusion particularly well suited to this setting.

However, applying discrete diffusion to generative recommendation still presents several challenges: (1) **Mismatch between Residual Quantization (RQ) and Discrete Diffusion.** Since discrete diffusion relies on a bidirectional Transformer where all tokens are equally important, it requires parallel tokenization schemes rather than hierarchical ones such as RQ-VAE. (2) **Beam Search is Not Directly Applicable to Discrete Diffusion.** Unlike standard discrete diffusion models that rely on probabilistic sampling, recommendation tasks require generating the top- k outputs. Beam search, widely used in autoregressive top- k generation (Rajput et al., 2023; Zheng et al., 2024), is designed for fixed left-to-right decoding and thus cannot be directly applied to discrete diffusion with adaptive generation order, requiring specialized decoding strategies. (3) **Differences between Language Modeling and Recommendation.** Original discrete diffusion models are primarily developed for language modeling and must be adapted to capture intra-item semantics and inter-item sequential relations in user histories for recommendation.

To address these challenges, we propose **LLaDA-Rec**, a bidirectional generative recommendation method based on discrete diffusion. Our ap-

proach consists of three main modules: (1) **Parallel Tokenization.** We design Multi-Head VQ-VAE, which encodes each item embedding, splits it into several sub-vectors, and queries them against separate codebooks to produce parallel semantic IDs. (2) **Discrete Diffusion Training.** We employ two masking mechanisms: user-history level masking to model inter-item sequential relationships, and next-item level masking, with the history fixed, to capture intra-item semantics and semantic relationships among tokens within an item. (3) **Discrete Diffusion Inference.** We adapt beam search to the diffusion setting by selecting the position with the highest model confidence at each step and then performing beam expansion at that position. Repeating this process yields the final top- k items.

Our main contributions are as follows:

- (1) We analyze the limitations of existing generative recommendation models, focusing on unidirectional constraints and error accumulation.
- (2) We propose LLaDA-Rec, a generative recommender based on discrete diffusion, which introduces parallel semantic IDs and develops training and inference methods tailored to recommendation.
- (3) Experiments on three datasets demonstrate the effectiveness of LLaDA-Rec, which consistently outperforms both item ID-based methods and semantic ID-based generative recommendation models, and establishes discrete diffusion as a promising new paradigm for generative recommendation.

2 Related Work

Generative Recommendation. Inspired by the success of large language models (LLMs) (Zhao et al., 2023), generative recommendation has received increasing attention (Zhai et al., 2024; Liu et al., 2025a; Hua et al., 2023; Zhou et al., 2025). In this paradigm, each item is represented as a sequence of discrete tokens, called semantic IDs (SIDs), and a generative model predicts the target item’s SIDs from a user’s interaction history. Generative recommendation typically involves two stages: item tokenization and autoregressive generation. Existing tokenization methods mainly include clustering-based approaches, such as SEATER (Si et al., 2024) and EAGER (Wang et al., 2024b), and vector-quantization approaches, including residual quantization methods such as TIGER (Rajput et al., 2023), LETTER (Wang et al., 2024a), LC-Rec (Zheng et al., 2024), and OneRec (Deng et al., 2025; Zhou et al., 2025),

as well as product quantization methods such as RPG (Hou et al., 2025). However, most existing methods combine autoregressive generation with hierarchical quantization, resulting in unidirectional constraints and error accumulation. To address this, we propose parallel semantic IDs and adopt discrete diffusion for generation.

Discrete Diffusion. Discrete diffusion models (Li et al., 2025a; Gu et al., 2022; Ou et al.; Zhu et al., 2025; Nie et al.) have recently emerged as a promising language modeling architecture. Built on a bidirectional Transformer, they are trained with a forward token-masking process and a reverse denoising objective, enabling stronger bidirectional context modeling. During inference, tokens are generated in parallel, while low-confidence tokens are iteratively re-masked and refined, yielding an adaptive generation order. Recent work, such as LLaDA (Nie et al., 2025), LLaDA-V (You et al., 2025), MMaDA (Yang et al., 2025), and LLaDA 1.5 (Zhu et al., 2025), has demonstrated the potential of this paradigm. In this work, we build on discrete diffusion to exploit its strengths in bidirectional modeling and adaptive generation for generative recommendation.

3 Preliminaries and Background

This section presents the problem formulation and the probability distribution of discrete diffusion.

3.1 Problem Formulation

Let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of users and items. Each user $u \in \mathcal{U}$ has a chronological interaction history $\mathcal{H} = \{i_1, i_2, \dots, i_{n-1}\}$, where i_j is the j -th interacted item. The goal is to predict the next item i_n for user u .

In generative recommendation, each item is represented as a sequence of discrete tokens, called semantic IDs (SIDs). The SID of item i is denoted by $s_i = [c_{i,1}, c_{i,2}, \dots, c_{i,M}]$, where M is the SID length and $c_{i,j}$ is the j -th token. Accordingly, a user’s interaction history can be flattened into the token sequence:

$$\mathcal{S}_{\mathcal{H}} = [c_{1,1}, \dots, c_{1,M}, c_{2,1}, \dots, c_{2,M}, \dots, c_{n-1,1}, \dots, c_{n-1,M}]. \quad (1)$$

The next-item prediction task is then formulated as maximizing the conditional probability:

$$\theta^* = \arg \max_{\theta} P_{\theta}(s_n | \mathcal{S}_{\mathcal{H}}), \quad (2)$$

where θ denotes the model parameters and θ^* the optimum.

3.2 Discrete Diffusion Modeling

Discrete diffusion generates the next item through a reverse process, where a mask predictor gradually predicts the M tokens of the next item from a fully masked sequence of M [MASK] tokens. At each step, the model predicts all masked positions in parallel, fixes the most confident predictions, and re-masks the remaining positions for further refinement. Formally:

$$P_{\theta}(s_n | \mathcal{S}_{\mathcal{H}}) = \prod_{t=1}^T \prod_{\substack{1 \leq m \leq M \\ c_{n,m}^t = [\text{MASK}]}} P_{\theta}(c_{n,m} | s_n^t, \mathcal{S}_{\mathcal{H}}), \quad (3)$$

where T is the number of generation steps, s_n^t is the input at step t , and the initial state s_n^1 contains only [MASK] tokens. Here, $c_{n,m}^t$ denotes the m -th token of s_n^t . At each step, s_n^t is concatenated with $\mathcal{S}_{\mathcal{H}}$ and fed into the mask predictor, implemented as a bidirectional Transformer encoder.

Eq. (3) generates tokens in T steps, typically with $T \leq M$. At each step, the mask predictor predicts all masked tokens in parallel, keeps the top- $\frac{M}{T}$ most confident ones, and re-masks the rest. This provides parallel prediction, adaptive generation order, and controllable generation steps. A detailed comparison with autoregressive modeling is provided in App. A.

4 Our Approach: LLaDA-Rec

This section introduces our framework, LLaDA-Rec, shown in Fig. 2. It consists of three modules: (1) **Parallel Tokenization** (§ 4.1), which uses Multi-Head VQ-VAE to generate parallel semantic IDs; (2) **Discrete Diffusion Training** (§ 4.2), which employs user-history level and next-item level masking; and (3) **Discrete Diffusion Inference** (§ 4.3), which adapts beam search to generate the top- k recommended items.

4.1 Parallel Tokenization

To align with autoregressive generation, existing generative recommendation models typically use hierarchical quantization methods such as RQ-VAE (Rajput et al., 2023; Wang et al., 2024a; Zheng et al., 2024) or RQ-Kmeans (Deng et al., 2025; Zhou et al., 2025), where earlier tokens play a more

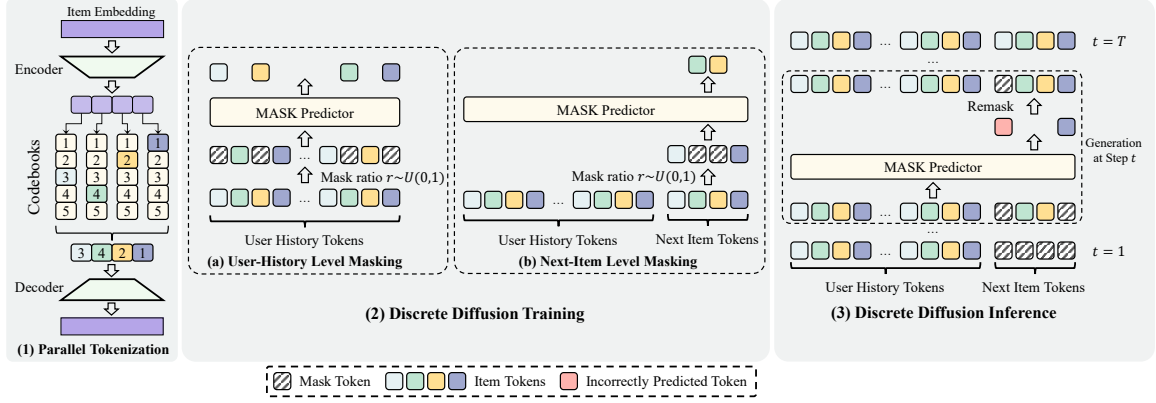


Figure 2: Overall framework of LLaDA-Rec, which consists of three modules: (1) Parallel Tokenization, where Multi-Head VQ-VAE produces parallel semantic IDs; (2) Discrete Diffusion Training, which uses user-history level and next-item level masking; and (3) Discrete Diffusion Inference, which adapts beam search for top- k recommendation.

dominant role because subsequent tokens depend on them. In contrast, the bidirectional Transformer used in discrete diffusion allows full interaction among all tokens, making them equally important in representation and generation. To better match this property, we propose **Multi-Head VQ-VAE**, which removes hierarchical dependencies and treats all tokens equally.

Specifically, we first encode the textual information of item i (e.g., title and description) using a pre-trained embedding model such as Sentence-T5 (Ni et al., 2021), obtaining its semantic representation $\mathbf{v}_i \in \mathbb{R}^D$. We then project $\mathbf{v}_i$ into a latent space for quantization, $\mathbf{z}_i = \text{Encoder}(\mathbf{v}_i)$, where $\mathbf{z}_i \in \mathbb{R}^d$ and $\text{Encoder}(\cdot)$ is an MLP. Next, we split $\mathbf{z}_i$ into M sub-vectors:

$$\mathbf{z}_i = [\mathbf{z}_{i,1}; \mathbf{z}_{i,2}; \dots; \mathbf{z}_{i,M}], \quad (4)$$

where $\mathbf{z}_{i,m} \in \mathbb{R}^{d/M}$ and each sub-vector is independently quantized.

We maintain M codebooks, where the m -th codebook is $\mathcal{C}_m = \{\mathbf{e}_{m,k}\}_{k=1}^K$, with K code embeddings $\mathbf{e}_{m,k} \in \mathbb{R}^{d/M}$. The quantization of the m -th sub-vector is defined as:

$$c_{i,m} = \arg \min_k \|\mathbf{z}_{i,m} - \mathbf{e}_{m,k}\|_2^2, \quad \mathbf{e}_{m,k} \in \mathcal{C}_m, \quad (5)$$

where $c_{i,m}$ is the selected code from the m -th codebook.

After quantizing all M sub-vectors, we obtain the SID of item i as $s_i = [c_{i,1}, c_{i,2}, \dots, c_{i,M}]$, along with the corresponding code embeddings $\{\mathbf{e}_{c_{i,1}}, \mathbf{e}_{c_{i,2}}, \dots, \mathbf{e}_{c_{i,M}}\}$. By concatenating these embeddings, we form the quantized representation $\hat{\mathbf{z}}_i = [\mathbf{e}_{c_{i,1}}; \mathbf{e}_{c_{i,2}}; \dots; \mathbf{e}_{c_{i,M}}]$, which is then fed into a

decoder to reconstruct the original semantic vector $\mathbf{v}_i$, i.e., $\hat{\mathbf{v}}_i = \text{Decoder}(\hat{\mathbf{z}}_i)$, where $\text{Decoder}(\cdot)$ is an MLP.

The VQ-VAE loss includes a reconstruction term $\mathcal{L}_{\text{Recon}}$ and a quantization term $\mathcal{L}_{\text{VQ}}$:

$$\begin{aligned} \mathcal{L}_{\text{Recon}} &= \|\mathbf{v}_i - \hat{\mathbf{v}}_i\|_2^2, \\ \mathcal{L}_{\text{VQ}} &= \sum_{m=1}^M (\|\text{sg}[\mathbf{z}_{i,m}] - \mathbf{e}_{c_{i,m}}\|_2^2 \\ &\quad + \alpha \|\mathbf{z}_{i,m} - \text{sg}[\mathbf{e}_{c_{i,m}}]\|_2^2), \\ \mathcal{L}_{\text{VQ-VAE}} &= \mathcal{L}_{\text{Recon}} + \mathcal{L}_{\text{VQ}}, \end{aligned} \quad (6)$$

where $\text{sg}[\cdot]$ is the stop-gradient operator and α is a hyper-parameter. Here, $\mathcal{L}_{\text{Recon}}$ reconstructs the original semantic vector, while $\mathcal{L}_{\text{VQ}}$ aligns each sub-vector with its corresponding code embedding.

4.2 Discrete Diffusion Training

Original discrete diffusion models are designed for language tasks (Nie et al., 2025) and do not naturally capture the semantic IDs used in recommendation. To bridge this gap, we design two masking mechanisms: (1) User-History level masking to model inter-item sequential dependencies, and (2) Next-Item level masking to learn intra-item token relationships for next-item generation.

4.2.1 Discrete Diffusion Process

The discrete diffusion model consists of two stages: the **forward process** and the **reverse process**. In the forward process, tokens are progressively masked. When the masking ratio is $r = 1$, all tokens are replaced by [MASK]; for $r \in (0, 1)$, each token is independently masked with probability r . In the reverse process, as r decreases from 1 to 0, the model gradually reconstructs the original sequence from the fully masked state.

Based on this framework, we design two masking strategies, user-history level masking and next-item level masking, to capture both inter-item and intra-item dependencies.

4.2.2 User-History Level Masking

We first apply discrete diffusion masking to the user history token sequence $\mathcal{S}_{\mathcal{H}}$ to help the mask predictor capture global dependencies among all tokens in the interaction history.

At each diffusion step $r \in (0, 1)$, each token in $\mathcal{S}_{\mathcal{H}}$ is independently masked with probability r . The resulting sequence, denoted by $\mathcal{S}_{\mathcal{H}}^r$, is fed into the mask predictor to reconstruct the masked tokens. The training loss is:

$$\mathcal{L}_{\text{His-Mask}} = -\mathbb{E}_{r, \mathcal{S}_{\mathcal{H}}, \mathcal{S}_{\mathcal{H}}^r} \left[\frac{1}{r} \sum_{i: \mathcal{S}_{\mathcal{H},i}^r = [\text{MASK}]} \log P_{\theta}(\mathcal{S}_{\mathcal{H},i} | \mathcal{S}_{\mathcal{H}}^r) \right]. \quad (7)$$

4.2.3 Next-Item Level Masking

We then apply discrete diffusion masking to the next-item token sequence while keeping the user history $\mathcal{S}_{\mathcal{H}}$ fully visible. The M tokens of the next item are progressively masked across diffusion steps, enabling the mask predictor to capture semantic relationships among tokens within the same item conditioned on the user history.

At each step $r \in (0, 1)$, each token in the next-item sequence s_n is independently masked with probability r . The resulting sequence, denoted by s_n^r , is concatenated with the history $\mathcal{S}_{\mathcal{H}}$ and fed into the mask predictor to reconstruct the masked tokens. The training objective is:

$$\mathcal{L}_{\text{Item-Mask}} = -\mathbb{E}_{r, s_n, s_n^r} \left[\frac{1}{r} \sum_{i: c_{n,i}^r = [\text{MASK}]} \log P_{\theta}(c_{n,i} | s_n^r, \mathcal{S}_{\mathcal{H}}) \right]. \quad (8)$$

The loss in Eq. (8) is an upper bound on the negative log-likelihood of the conditional distribution in Eq. (2), as shown in prior work (Shi et al., 2024; Ou et al.):

$$-\mathbb{E}[\log P_{\theta}(s_n | \mathcal{S}_{\mathcal{H}})] \leq \mathcal{L}_{\text{Item-Mask}}. \quad (9)$$

Therefore, minimizing Eq. (8) amounts to maximizing the conditional probability in Eq. (2), which justifies the loss design.

4.2.4 Joint Training

To jointly optimize both objectives, we train the mask predictor with Eq. (7) and Eq. (8). This allows the model to capture token dependencies in the user history while predicting the next item conditioned on historical context. The overall loss is:

$$\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{Item-Mask}} + \lambda_{\text{His-Mask}} \mathcal{L}_{\text{His-Mask}} + \lambda_{\text{Reg}} \|\theta\|_2^2, \quad (10)$$

where $\lambda_{\text{His-Mask}}$ balances the two masking losses, and λ_{Reg} controls the strength of L_2 regularization.

4.3 Discrete Diffusion Inference

After training the discrete diffusion model, our goal is to generate the top- k recommended items. This is challenging for two reasons: (1) original discrete diffusion models generate outputs by sampling (Nie et al., 2025), but greedy top-1 sampling cannot produce top- k results; and (2) standard beam search in generative recommendation (Rajput et al., 2023; Zheng et al., 2024) assumes a fixed left-to-right order, whereas discrete diffusion uses an adaptive generation order. To address this, we adapt beam search to discrete diffusion for top- k recommendation. This section presents the proposed inference method.

4.3.1 Initialization

As shown in Eq. (3), generation proceeds in T discrete steps. Let $\mathcal{PG}_t$ denote the set of positions already generated at step $t \in \{1, \dots, T\}$. Initially, $\mathcal{PG}_1 = \emptyset$, and the next-item token sequence is initialized as $s_n^1 = \{[\text{MASK}], \dots, [\text{MASK}]\}$ with M $[\text{MASK}]$ tokens.

Given s_n^t and the user history $\mathcal{S}_{\mathcal{H}}$ at step t , the mask predictor outputs a distribution over the vocabulary for each masked position:

$$P_{\theta}^{t,m}(w | s_n^t, \mathcal{S}_{\mathcal{H}}) \in [0, 1], \quad m \in \{1, \dots, M\} \setminus \mathcal{PG}_t, \quad w \in \{1, \dots, |\mathcal{W}|\}, \quad (11)$$

where $\mathcal{W}$ is the vocabulary, m indexes masked positions, and w indexes candidate tokens.

4.3.2 Generation Position Selection

Unlike left-to-right autoregressive generation, discrete diffusion predicts all $[\text{MASK}]$ positions in parallel at each step. To generate tokens iteratively, we first select the positions to fill at step t . Since M tokens are generated over T steps, we choose the top- $\frac{M}{T}$ unfilled positions with the highest maximum token probabilities:

$$\mathcal{M}_t = \underset{m \in \{1, \dots, M\} \setminus \mathcal{PG}_t}{\text{top-}\frac{M}{T}} \left(\max_{w \in \{1, \dots, |\mathcal{W}|\}} P_{\theta}^{t,m}(w | s_n^t, \mathcal{S}_{\mathcal{H}}) \right), \quad \mathcal{PG}_{t+1} = \mathcal{PG}_t \cup \mathcal{M}_t. \quad (12)$$

Here, $\mathcal{M}_t$ is the set of selected positions at step t , and $\mathcal{PG}_{t+1}$ is updated accordingly.

4.3.3 Beam Search for Discrete Diffusion

We perform beam search over the selected positions in $\mathcal{M}_t$. Let $\mathcal{B}_t$ denote the beam set at step t , and initialize $\mathcal{B}_{t,0} \leftarrow \mathcal{B}_t$ before processing positions in $\mathcal{M}_t$. Denote $\mathcal{M}_t = \{m_1, m_2, \dots, m_{|\mathcal{M}_t|}\}$. For each m_i , we expand the current beams with the top- B candidate tokens and then prune back to the

Table 1: Comparison of different generative recommendation methods.

Methods	Attention Mechanism	Generation Order	Controllable Generation Step
TIGER (Rajput et al., 2023)	Causal	Left2Right	✗
LETTER (Wang et al., 2024a)	Causal	Left2Right	✗
LC-Rec (Zheng et al., 2024)	Causal	Left2Right	✗
RPG (Hou et al., 2025)	Causal	Parallel	✗
LLaDA-Rec (Ours)	Bidirectional	Adaptive	✓

top- B beams:

$$\begin{aligned} \mathcal{B}_{t,i} &\leftarrow \mathcal{B}_{t,i-1} \cup \underset{w \in \{1, \dots, |\mathcal{W}|\}}{\text{top-}B} \left(P_{\theta}^{t,m_i}(w \mid s_n^t, \mathcal{S}_{\mathcal{H}}) \right), \\ \mathcal{B}_{t,i} &\leftarrow \underset{b \in \mathcal{B}_{t,i}}{\text{top-}B} \left(P_{\theta}^t(b \mid s_n^t, \mathcal{S}_{\mathcal{H}}) \right), \quad \mathcal{B}_{t+1} \leftarrow \mathcal{B}_{t,|\mathcal{M}_t|}. \end{aligned} \quad (13)$$

Here, B is the beam size, and $P_{\theta}^t(\cdot)$ denotes the joint probability of beam b at step t . After expansion, the tokens at positions in $\mathcal{M}_t$ are updated to form s_n^{t+1} .

4.3.4 Iterative Generation

In each iteration, $\mathcal{M}_t$ is selected by Eq. (12), while all unselected positions are **remasked** so that the mask predictor can re-evaluate them under the partially generated sequence. Position selection and beam search are repeated until all positions are filled. Finally, the generated sequences are ranked by their overall probabilities, and the top- k sequences are returned as recommendations.

This iterative procedure allows the discrete diffusion model to dynamically refine unselected positions, producing high-quality top- k results with greater flexibility than strict left-to-right autoregressive generation.

4.4 Discussion

Table 1 summarizes the key differences between LLaDA-Rec and representative generative recommenders. Existing autoregressive methods, such as TIGER, LETTER, and LC-Rec, use causal attention and a fixed left-to-right order, which may lead to error accumulation. RPG supports parallel generation, but it remains a one-shot process without iterative refinement. In contrast, LLaDA-Rec uses bidirectional attention, an adaptive generation order, and controllable generation steps, offering greater flexibility and better alignment with recommendation objectives. These advantages contribute to stronger performance.

Detailed discussions on continuous vs. discrete diffusion and comparisons with autoregressive models are provided in App. B.

Table 2: Statistics of the used datasets. “Avg.len” indicates the average number of interactions within each input sequence.

Dataset	#Users	#Items	#Interaction	Sparsity	Avg.len
Scientific	50,985	25,848	412,947	99.969%	8.10
Instrument	57,439	24,587	511,836	99.964%	8.91
Game	94,762	25,612	814,586	99.966%	8.60

5 Experiments

We performed comprehensive experiments to assess the effectiveness of LLaDA-Rec.

5.1 Experimental Setup

5.1.1 Dataset

We evaluate our method on three categories from the Amazon 2023 Review dataset (Hou et al., 2024): “Industrial Scientific” (Scientific), “Musical Instruments” (Instrument), and “Video Games” (Game). Following prior work (Rajput et al., 2023; Hou et al., 2023; Kang and McAuley, 2018), we use chronological user interaction sequences and the leave-one-out protocol. Dataset statistics are summarized in Tab. 2, and additional details are provided in App. C.1.

5.1.2 Baselines

We compare LLaDA-Rec with two groups of baselines.

Item ID-based: GRU4Rec (Hidasi et al., 2016), SASRec (Kang and McAuley, 2018), BERT4Rec (Sun et al., 2019), FMLP-Rec (Zhou et al., 2022), LRURec (Yue et al., 2024), DreamRec (Yang et al., 2023b), and DiffuRec (Li et al., 2023).

Semantic ID-based: VQ-Rec (Hou et al., 2023), TIGER (Rajput et al., 2023), TIGER-SAS (Rajput et al., 2023), LETTER (Wang et al., 2024a), DiffGRM (Liu et al., 2025b), LC-Rec (Zheng et al., 2024), and RPG (Hou et al., 2025).

Detailed descriptions are provided in App. C.2.

5.1.3 Evaluation and Implementation

Following prior studies (Rajput et al., 2023; Zheng et al., 2024), we report top- k Recall and NDCG for $k \in \{1, 5, 10\}$, omitting NDCG@1 since it is identical to Recall@1. Evaluation and implementation details are provided in App. C.3 and App. C.4.

5.2 Overall Performance

Table 3 summarizes the results of all methods on the three datasets. We can find that:

Table 3: Recommendation performance of different methods on the three datasets. R@k and N@k denote Recall@k and NDCG@k, respectively. The best and second-best results are marked in bold and underlined, respectively. Improvements over the second-best methods are statistically significant (paired t -test, $p < 0.05$).

Datasets	Metric	Item ID-based								Semantic ID-based						
		GRU4Rec	SASRec	BERT4Rec	FMLP-Rec	LRURec	DreamRec	DiffuRec	VQ-Rec	TIGER	TIGER-SAS	LETTER	DiffGRM	LC-Rec	RPG	LLaDA-Rec
Scientific	R@1	0.0071	0.0063	0.0045	0.0046	0.0049	0.0052	0.0050	0.0076	0.0084	0.0067	0.0082	0.0082	0.0091	0.0087	0.0098
	R@5	0.0184	0.0240	0.0157	0.0181	0.0169	0.0184	0.0190	0.0248	<u>0.0282</u>	0.0221	0.0273	0.0264	0.0280	0.0257	0.0310
	R@10	0.0272	0.0379	0.0264	0.0300	0.0267	0.0299	0.0310	0.0385	<u>0.0446</u>	0.0356	0.0423	0.0406	0.0434	0.0395	0.0474
	N@5	0.0128	0.0152	0.0100	0.0113	0.0110	0.0118	0.0119	0.0162	0.0183	0.0144	0.0179	0.0175	0.0186	0.0174	0.0203
Instrument	N@10	0.0156	0.0197	0.0134	0.0151	0.0141	0.0155	0.0158	0.0206	<u>0.0236</u>	0.0187	0.0227	0.0221	0.0235	0.0218	0.0256
	R@1	0.0094	0.0089	0.0065	0.0086	0.0071	0.0069	0.0077	0.0099	0.0105	0.0102	0.0114	0.0104	0.0119	0.0118	0.0128
	R@5	0.0297	0.0331	0.0255	0.0299	0.0272	0.0245	0.0283	0.0345	0.0359	0.0342	0.0362	0.0356	0.0379	0.0362	0.0406
	R@10	0.0453	0.0525	0.0412	0.0496	0.0431	0.0423	0.0465	0.0532	0.0566	0.0521	0.0562	0.0565	0.0587	0.0545	0.0623
Game	N@5	0.0196	0.0211	0.0160	0.0193	0.0172	0.0157	0.0179	0.0222	0.0233	0.0223	0.0239	0.0231	0.0251	0.0241	0.0268
	N@10	0.0246	0.0273	0.0211	0.0257	0.0223	0.0214	0.0237	0.0282	0.0300	0.0280	0.0303	0.0298	0.0318	0.0300	0.0337
	R@1	0.0149	0.0128	0.0082	0.0099	0.0134	0.0125	0.0111	0.0150	0.0166	0.0170	0.0169	0.0155	0.0165	0.0159	0.0203
	R@5	0.0461	0.0516	0.0315	0.0395	0.0480	0.0381	0.0425	0.0497	0.0529	0.0548	0.0552	0.0518	0.0567	0.0579	0.0623
	R@10	0.0712	0.0823	0.0530	0.0649	0.0753	0.0611	0.0709	0.0769	0.0823	0.0847	0.0863	0.0811	0.0891	0.0853	0.0942
	N@5	0.0307	0.0323	0.0199	0.0246	0.0308	0.0253	0.0268	0.0325	0.0348	0.0360	0.0362	0.0336	0.0366	0.0397	0.0415
	N@10	0.0387	0.0421	0.0267	0.0328	0.0396	0.0326	0.0359	0.0412	0.0442	0.0457	0.0462	0.0430	0.0471	0.0485	0.0517

Table 4: Ablation results on the three datasets. “w/o” indicates that the corresponding module is removed.

Model	Scientific		Instrument		Game	
	R@5	N@5	R@5	N@5	R@5	N@5
LLaDA-Rec	0.0310	0.0203	0.0406	0.0268	0.0623	0.0415
Tokenizer						
RQ-VAE	0.0293	0.0191	0.0367	0.0244	0.0604	0.0399
RQ-Kmeans	0.0250	0.0165	0.0344	0.0224	0.0552	0.0370
OPQ	0.0237	0.0155	0.0340	0.0229	0.0552	0.0362
Training						
w/o $\mathcal{L}_{\text{His-Mask}}$	0.0255	0.0169	0.0321	0.0209	0.0544	0.0356
w/o $\mathcal{L}_{\text{Item-Mask}}$	0.0264	0.0172	0.0355	0.0231	0.0571	0.0376
Inference						
w/o Beam Search	0.0077	0.0077	0.0091	0.0091	0.0162	0.0162

- (1) Compared with both conventional item ID-based recommenders and generative semantic ID-based methods, LLaDA-Rec achieves state-of-the-art performance, demonstrating the effectiveness of our discrete diffusion framework and Multi-Head VQ-VAE in improving recommendation accuracy.
- (2) Generative recommendation methods using SIDs consistently outperform traditional item ID-based methods, showing that SIDs better capture inter-item correlations and richer semantics, which further improve recommendation quality.
- (3) Parallel SID-based methods, such as RPG and LLaDA-Rec, achieve competitive results across all datasets, showing the effectiveness of parallel SID generation. The stronger performance of LLaDA-Rec over RPG further demonstrates the benefit of incorporating discrete diffusion.

5.3 Ablation Study

We conduct ablation studies to evaluate each module in LLaDA-Rec, as shown in Tab. 4.

Tokenizer. As discussed in § 4.1, we design Multi-Head VQ-VAE to better adapt discrete diffusion with a bidirectional Transformer. We compare it with several SID generation methods, including RQ-VAE (Rajput et al., 2023; Zheng et al., 2024), RQ-Kmeans (Deng et al., 2025), and OPQ (Hou

et al., 2023, 2025). Results show that residual-quantization-based methods consistently underperform, suggesting that they are less compatible with bidirectional Transformers. In RQ, earlier tokens tend to dominate, whereas LLaDA-Rec assumes more balanced token importance, making it better suited to parallel SIDs.

Moreover, even when using RQ-VAE as the tokenizer, our method still outperforms the baseline in most cases, demonstrating the robustness of our framework. Finally, clustering-based methods such as RQ-Kmeans and OPQ perform worse than VAE-based methods, confirming the stronger representational capacity of VAE-based quantization.

Training. In § 4.2, we introduce two masking mechanisms and analyze their corresponding loss functions. Removing either loss degrades performance. The loss $\mathcal{L}_{\text{His-Mask}}$ (Eq. (7)) helps model relationships across items, while $\mathcal{L}_{\text{Item-Mask}}$ (Eq. (8)) captures semantic dependencies among tokens within the same item and supports next-item prediction conditioned on the user history.

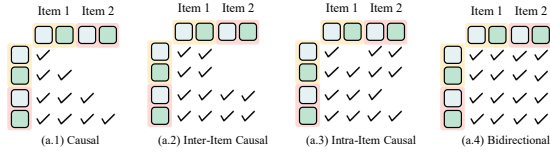
Inference. As discussed in § 4.3, the original diffusion language model (Nie et al., 2025) uses probabilistic sampling and returns only the top-1 result. We adapt beam search to generate top- k items. Replacing beam search with the original greedy strategy substantially degrades performance, highlighting its importance for generative recommendation. Note that greedy search produces only one result, making Recall and NDCG equivalent.

5.4 Experimental Analysis

We further performed experimental analysis to assess the effectiveness of LLaDA-Rec.

5.4.1 Impact of the Attention Mechanism

Our method is built on a bidirectional Transformer, which better captures contextual dependencies in both directions. We compare it with three alter-



(a) Different Attention Mechanism

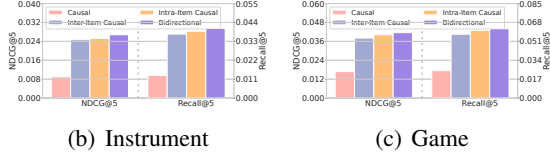


Figure 3: Comparison of different attention mechanisms. (a): Illustration of the attention masks used by each mechanism. (b) and (c): Recommendation performance under different attention mechanisms.

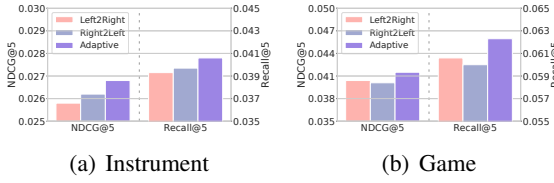


Figure 4: Impact of Generation Order.

native attention mechanisms: (1) causal attention, commonly used in autoregressive models; (2) inter-item causal attention, where attention across items is causal but within each item is bidirectional; and (3) intra-item causal attention, where attention within each item is causal but across items is bidirectional. The corresponding attention masks are shown in Fig. 3(a).

Figure 3 compares their performance. Bidirectional attention performs best, likely due to its stronger contextual modeling ability. In contrast, fully causal attention performs worst because its unidirectional structure limits context utilization. Inter-item causal and intra-item causal attention both achieve competitive results, suggesting that bidirectional attention, either across items or within items, is important for contextual modeling.

5.4.2 Impact of Generation Order

As described in § 4.3, our generation order is determined dynamically by selecting the position with the highest model confidence at each step. We compare this adaptive strategy with fixed left-to-right (left2right) and right-to-left (right2left) orders (Fig. 4). The adaptive strategy consistently performs best, while left2right sometimes gives the worst results, revealing the limitations of this commonly used order. By generating from easier to harder tokens, our method gains a clear advantage.

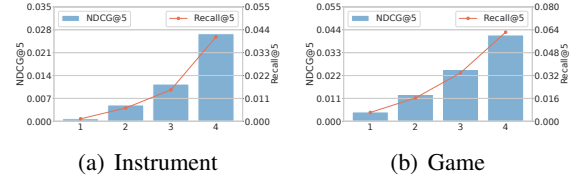


Figure 5: Impact of Generation Steps.

Table 5: Collision and validity rates of SIDs across datasets. Collision rate is the proportion of distinct items sharing the same SID and remains very low. Validity rate is the proportion of generated top- k SIDs that correspond to valid items and stays close to 100% across all k .

Datasets	Collision Rate	Validity Rate			
		$k = 10$	$k = 50$	$k = 100$	$k = 500$
Scientific	0.2476%	99.96%	99.78%	99.64%	98.87%
Instrument	0.2400%	99.99%	99.96%	99.90%	99.28%
Game	0.7770%	99.98%	99.83%	99.65%	98.51%

5.4.3 Impact of Generation Steps

As discussed in § 3.2 and § 4.3, the number of generation steps can be controlled so that the model generates multiple tokens per step. Figure 5 shows that more steps generally lead to better performance. Although fewer steps improve generation efficiency, they also cause a clear performance drop. Achieving a better trade-off between efficiency and performance with fewer steps remains an open question, though recent studies on diffusion language models have begun to explore this direction (Chen et al., 2025; Hayakawa et al., 2024).

5.4.4 Collision and Validity Rate of SIDs

We further examine the collision rate of SIDs and the validity rate of the model-generated top- k SIDs, as reported in Tab. 5. The collision rate measures the proportion of distinct items sharing the same SID. Its extremely low value indicates that SIDs can effectively distinguish items.

The validity rate measures the proportion of generated top- k SIDs that correspond to valid items. Across different k values, the validity rate remains close to 100%, suggesting that the model has successfully learned the mapping between items and their corresponding SIDs.

For additional hyper-parameter analysis results, please refer to App. D.1.

6 Conclusion

This work overcomes the unidirectional constraints and addresses error accumulation in autoregressive-based generative recommendation through discrete

diffusion. We propose LLaDA-Rec, a bidirectional framework with parallel semantic IDs, tailored masking, and adapted beam search for recommendation. It captures both inter-item and intra-item dependencies while reducing error propagation. Experiments on three real-world datasets show that LLaDA-Rec consistently outperforms both traditional ID-based recommenders and state-of-the-art semantic ID-based generative methods.

Limitations

Although LLaDA-Rec achieves strong results, several directions remain for further improvement. First, while discrete diffusion enables parallel SID generation, reducing the number of generation steps may still lead to performance degradation. This indicates that there is room to further improve the efficiency–performance trade-off. Developing methods that maintain strong recommendation quality with fewer steps would make the framework more practical. Second, our current decoding strategy adopts a confidence-guided generation order. While this heuristic is simple and effective, it is still rule-based and may not fully exploit the flexibility of adaptive-order generation. Exploring end-to-end learned decoding strategies may further improve performance. Finally, we evaluate LLaDA-Rec on public benchmark datasets. It would be valuable to further examine its applicability in real-world large-scale recommendation scenarios, especially under practical latency and deployment constraints.

Acknowledgement

This work was funded by the National Key R&D Program of China (2023YFA1008704), the National Natural Science Foundation of China (No. 62472426, 62502091), the Beijing-Tianjin-Hebei Natural Science Foundation Cooperation Project (25JJJC0045). Work partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education, and Beijing Key Laboratory of Research on Large Models and Intelligent Governance.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Tianqi Chen, Shujian Zhang, and Mingyuan Zhou. 2025. Dlm-one: Diffusion language models for one-step sequence generation. *arXiv preprint arXiv:2506.00290*.

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

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*.

Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. 2022. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10696–10706.

Satoshi Hayakawa, Yuhta Takida, Masaaki Imaizumi, Hiromi Wakaki, and Yuki Mitsufuji. 2024. Distillation of discrete diffusion through dimensional correlations. *arXiv preprint arXiv:2410.08709*.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*, pages 1162–1171.

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

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

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.

- Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. 2025a. A survey on diffusion language models. *arXiv preprint arXiv:2508.10875*.
- Wuchao Li, Rui Huang, Haijun Zhao, Chi Liu, Kai Zheng, Qi Liu, Na Mou, Guorui Zhou, Defu Lian, Yang Song, and 1 others. 2025b. Dimerec: a unified framework for enhanced sequential recommendation via generative diffusion models. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pages 726–734.
- Zihao Li, Aixin Sun, and Chenliang Li. 2023. Diffurec: A diffusion model for sequential recommendation. *ACM Transactions on Information Systems*, 42(3):1–28.
- Xinyu Lin, Haihan Shi, Wenjie Wang, Fuli Feng, Qifan Wang, See-Kiong Ng, and Tat-Seng Chua. 2025. Order-agnostic identifier for large language model-based generative recommendation. In *Proceedings of the 48th international ACM SIGIR conference on research and development in information retrieval*, pages 1923–1933.
- Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. 2025a. Generative recommender with end-to-end learnable item tokenization. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 729–739.
- Zhao Liu, Yichen Zhu, Yiqing Yang, Guoping Tang, Rui Huang, Qiang Luo, Xiao Lv, Ruiming Tang, Kun Gai, and Guorui Zhou. 2025b. Diffgrm: Diffusion-based generative recommendation model. *arXiv preprint arXiv:2510.21805*.
- Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*.
- Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. In *The Thirteenth International Conference on Learning Representations*.
- Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *arXiv preprint arXiv:2502.09992*.
- Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *The Thirteenth International Conference on Learning Representations*.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, and 1 others. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315.
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.
- Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. 2024. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167.
- Zihua Si, Zhongxiang Sun, Jiale Chen, Guozhang Chen, Xiaoxue Zang, Kai Zheng, Yang Song, Xiao Zhang, Jun Xu, and Kun Gai. 2024. Generative retrieval with semantic tree-structured identifiers and contrastive learning. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 154–163.
- 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, CIKM '19*, pages 1441–1450, New York, NY, USA. ACM.
- Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024a. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2400–2409.
- Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangan He, and Tat-Seng Chua. 2023. Diffusion recommender model. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*, pages 832–841.
- Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou

- Zhao, and 1 others. 2024b. Eager: Two-stream generative recommender with behavior-semantic collaboration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3245–3254.
- Ling Yang, Ye Tian, Bowen Li, Xinchun Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. 2025. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*.
- Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023a. Diffusion models: A comprehensive survey of methods and applications. *ACM computing surveys*, 56(4):1–39.
- Zhengyi Yang, Jiancan Wu, Zhicai Wang, Xiang Wang, Yancheng Yuan, and Xiangnan He. 2023b. Generate what you prefer: Reshaping sequential recommendation via guided diffusion. *Advances in Neural Information Processing Systems*, 36:24247–24261.
- Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. 2025. Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*.
- Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. 2024. Linear recurrent units for sequential recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 930–938.
- Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, and 1 others. 2024. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *International Conference on Machine Learning*, pages 58484–58509. PMLR.
- Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).
- 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.
- Guorui Zhou, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Qiang Luo, Qianqian Wang, Qigen Hu, Rui Huang, Shiyao Wang, and 1 others. 2025. Onerec technical report. *arXiv preprint arXiv:2506.13695*.
- 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, WWW ’22*, page 2388–2399, New York, NY, USA. Association for Computing Machinery.
- Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and 1 others. 2025. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*.

A Autoregressive and Discrete Diffusion

We provide a detailed comparison between autoregressive generation and discrete diffusion generation in generative recommendation.

A.1 Autoregressive Modeling

Existing generative recommendation methods, such as TIGER (Rajput et al., 2023), mainly follow an autoregressive paradigm, where tokens are generated sequentially from left to right. At each step, the prediction of the current token is conditioned on the user history and all previously generated tokens. Formally, the conditional probability of the next item sequence s_n is factorized as

$$P_{\theta}(s_n | \mathcal{S}_{\mathcal{H}}) = \prod_{m=1}^M P_{\theta}(c_{n,m} | c_{n,<m}, \mathcal{S}_{\mathcal{H}}), \quad (14)$$

where $c_{n,<m}$ denotes the tokens preceding $c_{n,m}$. For the first token, $c_{n,<1}$ is a special [BOS] token.

This formulation requires exactly M decoding steps to generate an item with M tokens. Since each prediction depends on previous outputs, generation follows a fixed order and may suffer from error accumulation, where early mistakes can affect subsequent token predictions.

A.2 Discrete Diffusion Modeling

In contrast, discrete diffusion generates the next item through a reverse process, where a mask predictor gradually predicts the M tokens of the next item starting from a fully masked sequence of M [MASK] tokens. At each step, the model predicts all masked positions in parallel, keeps the most confident predictions, and re-masks the remaining positions for further refinement. Formally,

$$P_{\theta}(s_n | \mathcal{S}_{\mathcal{H}}) = \prod_{t=1}^T \prod_{m=1}^M P_{\theta}(c_{n,m} | s_n^t, \mathcal{S}_{\mathcal{H}}), \quad (15)$$

where T is the number of generation steps, s_n^t is the input sequence at step t , and the initial state s_n^1 consists entirely of [MASK] tokens. Here, $c_{n,m}^t$ denotes the m -th token of s_n^t . At each step, s_n^t is concatenated with the user history $\mathcal{S}_{\mathcal{H}}$ and fed into the mask predictor, which is implemented as a bidirectional Transformer encoder.

Compared with autoregressive generation in Eq. (14), the discrete diffusion formulation in Eq. (15) generates tokens in T steps, where typically $T \leq M$. At each step, the mask predictor predicts all masked tokens in parallel, retains the top- $\frac{M}{T}$ most confident ones, and re-masks the remaining positions. This process enables iterative refinement and a dynamically determined generation order.

A.3 Discussion

The two formulations differ in several important aspects. Autoregressive modeling uses a fixed left-to-right decoding order and causal attention, which restricts each token to attend only to previous tokens. In contrast, discrete diffusion uses bidirectional attention and an adaptive generation order determined by token confidence. This allows the model to capture richer contextual dependencies and reduce the impact of early prediction errors.

Moreover, autoregressive generation requires exactly M decoding steps for an M -token item, while discrete diffusion allows explicit control over the number of generation steps through T . When $T < M$, multiple tokens can be generated at each step, leading to more efficient inference. The re-masking mechanism further allows the model to revisit uncertain positions and refine them iteratively, which is not possible in autoregressive decoding.

Overall, discrete diffusion offers three key advantages for generative recommendation: parallel token prediction, adaptive generation order, and controllable generation steps, making it a flexible alternative to autoregressive modeling.

B Additional Discussion

We provide additional discussion on the advantages of discrete diffusion for recommendation, including its difference from continuous diffusion and its benefits over autoregressive generation.

B.1 Continuous vs. Discrete Diffusion in Recommendation

Continuous diffusion models have been widely used in image generation (Yang et al., 2023a; Rombach et al., 2022) and sequential recommendation (Yang et al., 2023b; Li et al., 2023; Wang et al., 2023; Li et al., 2025b). Operating in continuous spaces, they generate high-quality images or latent representations through a forward noising process and a reverse denoising process. By

contrast, discrete diffusion models (Li et al., 2025a; Nie et al., 2025) are designed to generate sequences of discrete tokens.

In recommendation tasks, continuous-diffusion-based methods (Yang et al., 2023b; Li et al., 2023) typically generate a latent representation and then retrieve items through similarity search. In this setting, generation and retrieval are handled in separate stages. By comparison, our discrete diffusion approach directly generates the semantic IDs of items, removing the extra retrieval stage and unifying generation and retrieval into a single optimization process. This design simplifies inference and improves recommendation performance.

B.2 Advantages over Autoregressive Models

Autoregressive generative recommendation methods (Rajput et al., 2023; Zheng et al., 2024; Wang et al., 2024a; Liu et al., 2025a) generate tokens in a fixed left-to-right order, leading to strong sequential dependencies and potential error accumulation, where early mistakes may affect later predictions. Their attention mechanism is also unidirectional (causal), so each token can only attend to its preceding tokens.

Although RPG (Hou et al., 2025) supports parallel generation of semantic IDs using multi-token prediction (Gloeckle et al., 2024), it performs prediction in a single step and therefore cannot iteratively refine uncertain tokens through re-masking and re-prediction. In addition, tokens in RPG are predicted independently, making it difficult to exploit dependencies across token positions. It also relies on a relatively complex decoding strategy.

In contrast, discrete diffusion adopts an adaptive, confidence-driven generation order that prioritizes high-confidence tokens, which helps reduce the impact of early prediction errors. It further uses bidirectional attention, allowing each token to attend to both preceding and succeeding positions for richer contextual modeling. Moreover, discrete diffusion supports controllable generation steps and can predict multiple tokens at each step.

Overall, bidirectional modeling and adaptive generation order make the generation process better aligned with recommendation objectives, leading to improved accuracy. Table 1 summarizes the main differences between LLaDA-Rec and representative generative recommendation methods.

C Experimental Details

We provide additional experimental details, including the datasets, baselines, evaluation, and implementation details.

C.1 Dataset

We evaluate our method on three categories from the widely used Amazon 2023 Review dataset (Hou et al., 2024): *Industrial Scientific* (Scientific), *Musical Instruments* (Instrument), and *Video Games* (Game). Following prior work (Rajput et al., 2023; Hou et al., 2023), we treat each user’s historical reviews as interaction records and sort them chronologically, with earlier interactions appearing first.

For evaluation, we adopt the widely used leave-one-out protocol (Kang and McAuley, 2018; Rajput et al., 2023). Specifically, for each user sequence, the last item is used for testing, the second-to-last item is used for validation, and the remaining items are used for training.

License. The Amazon 2023 dataset (Hou et al., 2024) is available under the CC BY 4.0 License, and the associated code is available under the MIT License.

C.2 Baselines

We first compared our approach with traditional item ID-based methods.

Item ID-based: (1) **GRU4Rec** (Hidasi et al., 2016) leverages gated recurrent units (GRUs) to model user interaction histories; (2) **SASRec** (Kang and McAuley, 2018) employs a unidirectional Transformer to capture sequential dependencies; (3) **BERT4Rec** (Sun et al., 2019) utilizes a bidirectional Transformer trained with a cloze-style objective; (4) **FMLP-Rec** (Zhou et al., 2022) adopts multi-layer perceptrons (MLPs) with learnable filters for sequential modeling; (5) **LRURec** (Yue et al., 2024) integrates linear recurrent units (LRUs) to efficiently process long-range user interactions. (6) **DreamRec** (Yang et al., 2023b) uses SASRec outputs in a diffusion denoising module, removing negative sampling and training only on positive samples; (7) **DiffuRec** (Li et al., 2023) combines generative diffusion with sequential recommendation via a Transformer approximator to reconstruct target item embeddings.

We also compared our approach with generative recommendation methods based on semantic IDs. **Semantic ID-based:** (8) **VQ-Rec** (Hou et al., 2023) applies product quantization to tokenize

items into semantic IDs, which are then pooled to obtain item representations; (9) **TIGER** (Rajput et al., 2023) utilizes RQ-VAE to generate codebook identifiers, embedding semantic information into discrete code sequences; (10) **TIGER-SAS** (Rajput et al., 2023) derives semantic IDs from SASRec-trained item embeddings rather than text embeddings; (11) **LETTER** (Wang et al., 2024a) develops a learnable tokenizer that incorporates hierarchical semantics, collaborative signals, and code assignment diversity; (12) **Diff-GRM** (Liu et al., 2025b) uses masked discrete diffusion with Parallel Semantic Encoding for parallel semantic ID generation; (13) **LC-Rec** (Zheng et al., 2024) exploits identifiers with auxiliary alignment tasks to associate the generated codes with natural language; (14) **RPG** (Hou et al., 2025) is a lightweight semantic ID-based model that generates long, unordered semantic IDs in parallel via multi-token prediction (Gloeckle et al., 2024).

C.3 Evaluation Metrics

Following prior studies (Rajput et al., 2023; Wang et al., 2024a; Zheng et al., 2024), we evaluate performance using two commonly adopted ranking metrics: top- k *Recall* and top- k *Normalized Discounted Cumulative Gain* (NDCG). We report results for $k \in \{1, 5, 10\}$. Since NDCG@1 is identical to Recall@1, it is omitted from the evaluation.

C.4 Implementation Details

For the parallel tokenization module, we adopt Sentence-T5 (Ni et al., 2021) to encode the title and other textual information of each item into an embedding. We employ $M = 4$ codebooks, each containing $K = 256$ code vectors of dimension $d = 32$. The weight α in Eq. (6) is set to 0.25. The Multi-Head VQ-VAE is trained for 10,000 epochs using the AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of 1×10^{-3} and a batch size of 2,048.

For the discrete diffusion model, the MASK predictor is a bidirectional Transformer encoder with a token embedding dimension of 256 and 8 attention heads per layer. On the Scientific and Instrument datasets, we use a 4-layer encoder, whereas on the Game dataset we use a 6-layer encoder.

The model parameters are randomly initialized and trained using our designed loss function. The weight $\lambda_{\text{His-Mask}}$ in Eq. (10) is tuned over $\{1, 2, 3, 4, 5\}$. The maximum historical length for all datasets is set to 20. We train the Transformer

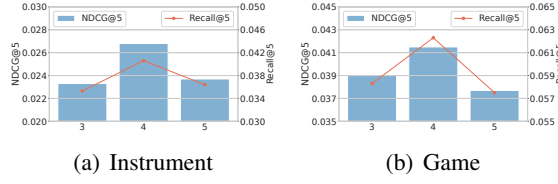


Figure 6: Impact of Semantic ID Length.

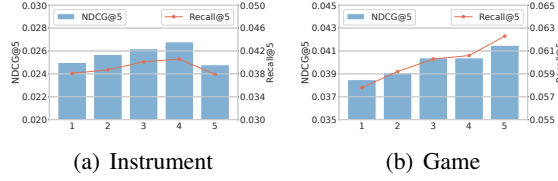


Figure 7: Impact of $\lambda_{\text{His-Mask}}$.

encoder for 150 epochs with early stopping, using the AdamW optimizer. The learning rate is tuned over $\{0.005, 0.003, 0.001\}$ and the weight decay over $\{0.05, 0.005, 0.001\}$. The batch size is set to 1,024.

D Additional Experimental Results

We provide additional experimental results.

D.1 Impact of Hyper-parameters

We investigate the impact of different hyperparameters on the results.

We investigate the influence of semantic ID length M on the performance, as illustrated in Fig. 6. Results indicate that both shorter ($M = 3$) and longer ($M = 5$) SIDs yield inferior performance. Shorter SIDs suffer from greater information loss and a higher collision rate (i.e., different items sharing the same SID), whereas longer SIDs, despite retaining more information, are harder to learn, hinder model convergence, and increase inference latency. Based on these observations, we adopt $M = 4$ in this study. Leveraging longer SIDs to preserve richer information while achieving stronger recommendation performance remains a avenue for future research.

We also analyze the effect of the weight $\lambda_{\text{His-Mask}}$ in Eq. (10), as illustrated in Fig. 7. The results show that moderately increasing $\lambda_{\text{His-Mask}}$ can improve performance, enabling the model to better capture relationships among different tokens in the history. However, setting $\lambda_{\text{His-Mask}}$ too large may hinder the model’s ability to predict the next item conditioned on the given history.