Full-Sequence Masked Diffusion for Generative Recommendation

Lei Chen[1] **Zibo Wei**[1] *

[1]University of Science and Technology of China

Abstract

Autoregressive large language models have dominated generative recommendation by sequentially predicting the next item. However, this unidirectional paradigm suffers from error propagation, and limited ability to jointly optimize an entire recommendation list. In this paper, we introduce Full-Sequence Masked Diffusion for generative recommendation (FSMD), which extends masked diffusion models to enable global joint modeling over the entire user history and Top-K recommendation list. We formulate recommendation as a unified sequence diffusion process, where the forward process randomly masks tokens across the full sequence and the reverse process simultaneously denoises all masked positions using bidirectional context. This formulation enables parallel generation of the entire recommendation list and naturally captures inter-item dependencies within a coherent global structure. To better align generation with ranking objectives, we further introduce a unified training objective that combines masked diffusion likelihood with a listwise ranking loss. Extensive experiments on multiple public benchmarks demonstrate that FSMD consistently outperforms state-of-the-art autoregressive and item-level diffusion-based generative recommendation methods across accuracy and list quality metrics.

1 Introduction

Generative recommendation has recently emerged as a promising paradigm by framing recommendation as a sequence generation problem with large language models (LLMs) [14, 24, 38, 39]. By modeling user–item interactions as token sequences, autoregressive (AR) generation enables flexible, unified modeling across tasks and domains, and has demonstrated strong empirical performance. However, this formulation inherits a fundamental limitation: it reduces recommendation to a token-by-token prediction problem, where items are generated sequentially in a fixed order.

This sequential factorization introduces well-known issues. Errors made in early steps propagate irreversibly, leading to degraded list quality. More importantly, autoregressive generation [13, 37, 48] optimizes items locally, without explicitly modeling the recommendation list as a coherent whole. In practice, recommendation is inherently a list-level decision problem, where the utility depends on the joint configuration of all items (e.g., diversity, complementarity, and coverage), rather than independent token-level likelihoods. This mismatch between local generation and global utility fundamentally limits the effectiveness of AR-based generative recommenders.

Recent advances in masked diffusion language models (e.g., LLaDA [26]) offer an alternative paradigm [21, 26, 34]. By replacing left-to-right generation with iterative denoising over masked tokens, diffusion models enable bidirectional context modeling and parallel prediction, allowing global refinement of sequences. Initial efforts have applied diffusion to recommendation [25, 31, 33], primarily focusing on item-level generation or next-item prediction, where diffusion operates within

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

individual items or local contexts. While effective, these approaches still inherit a key limitation: they do not fully exploit the ability of diffusion models to perform global optimization over an entire recommendation list.

In this paper, we take a step further and propose to reformulate generative recommendation as a discrete sequence optimization problem over the entire list. Instead of generating items incrementally, we treat the user history and the target recommendation list as a unified sequence, and perform full-sequence masked diffusion to jointly refine all items. Starting from a fully masked list, the model iteratively denoises the sequence, where each step updates all positions simultaneously using bidirectional context. To this end, we introduce Full-Sequence Masked Diffusion (FSMD) for generative recommendation. Our framework enables (i) global joint modeling of the entire recommendation list, capturing rich inter-item dependencies, (ii) parallel generation of all items with iterative refinement, and (iii) alignment with recommendation objectives via a listwise optimization loss. Extensive experiments on multiple public benchmarks demonstrate that our approach consistently outperforms strong autoregressive and diffusion-based baselines across ranking and diversity metrics.

2 Preliminaries

Problem Formulation. We study generative recommendation, where the goal is to produce a ranked list of items conditioned on a user's historical interactions. Let $\mathbf{h}_u = (h_1, \ldots, h_{L_h})$ denote the interaction history of user u, and let $\mathbf{r} = (r_1, \ldots, r_K)$ denote a recommendation list of length K. The objective is to generate a list that maximizes a list-level utility function:

$$\mathbf{r}^* = \arg\max_{\mathbf{r}} \; U(\mathbf{r} \mid u), \tag{1}$$

where $U(\cdot)$ captures ranking quality, coherence, and diversity. To enable generative modeling, we represent both history and recommendation list as a unified token sequence $\mathbf{s}_0 = [\mathbf{h}_u \parallel \mathbf{r}]$. Each item is encoded as a sequence of discrete tokens via semantic tokenization. The total length is $L = L_h + K \cdot L_{\text{item}}$.

Masked Diffusion for Discrete Sequences. We adopt masked diffusion models for sequence generation, which define a forward corruption process and a reverse denoising process over $\mathbf{s}_0$. In the forward process, a noise level $t \sim \mathcal{U}[0, 1]$ is sampled, and each token is independently replaced by a special mask token M with probability t, yielding a corrupted sequence $\mathbf{s}_t$: $q_t(\mathbf{s}_t \mid \mathbf{s}_0) = \prod_{i=1}^{L} q_t(s_{i,t} \mid s_{i,0})$, $q_t(s_{i,t} = M \mid s_{i,0}) = t$ and $q_t(s_{i,t} = s_{i,0}) = 1 - t$. The reverse process is parameterized by a Transformer without causal masking, which predicts all masked tokens simultaneously using bidirectional context: $p_\theta(\mathbf{s}_0 \mid \mathbf{s}_t) = \prod_{i:s_{i,t}=M} p_\theta(s_{i,0} \mid \mathbf{s}_t)$. The model is trained with a masked cross-entropy objective:

$$\mathcal{L}_{\text{diff}} = -\mathbb{E}_{t,\mathbf{s}_0,\mathbf{s}_t}\left[\frac{1}{t}\sum_{i=1}^{L} \mathbf{1}[s_{i,t} = M] \log p_\theta(s_{i,0} \mid \mathbf{s}_t)\right], \tag{2}$$

which serves as an upper bound on the negative log-likelihood.

From Autoregressive Factorization to Joint Modeling. Autoregressive (AR) models factorize the sequence distribution $P_{\text{AR}}(\mathbf{s}_0) = \prod_{i=1}^{L} P(s_i \mid s_{<i})$, the distribution enforces a fixed generation order and prevents future tokens from influencing earlier decisions. When applied to recommendation, this leads to *sequential decision making*, where each item is generated conditioned only on previously generated items. In contrast, masked diffusion models define a joint distribution over the entire sequence $P_\theta(\mathbf{s}_0) = P_\theta(\mathbf{h}_u, \mathbf{r})$, and iteratively refine all positions in parallel $\mathbf{s}^{(t-1)} \sim p_\theta(\mathbf{s} \mid \mathbf{s}^{(t)})$. This enables bidirectional interactions across all tokens during generation.

Implications for Recommendation. Recommendation is inherently a list-level decision problem, where the quality of each item depends on the entire set. Sequential factorization is misaligned with this objective, as it cannot capture global dependencies such as complementarity, redundancy, and diversity. By modeling $\mathbf{h}_u$ and $\mathbf{r}$ jointly within a single sequence and refining all positions simultaneously, masked diffusion provides a natural framework for *full-list generation*. However, existing diffusion-based recommendation methods operate at the item level or next-item level, limiting their ability to model dependencies across multiple recommended items.

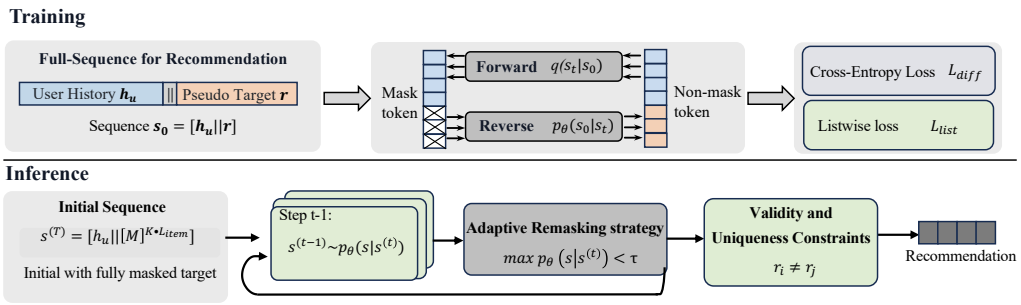

Figure 1: The overall framework of FSMD.

3 Method

3.1 Full-Sequence Modeling for Recommendation

We reformulate generative recommendation as a *full-sequence optimization problem*, where the entire Top-K recommendation list is generated jointly rather than sequentially. Given a user u with interaction history $\mathbf{h}_u$, we construct a unified sequence $\mathbf{s} = [\mathbf{h}_u] \parallel [\mathbf{r}]$. During both training and inference, the list portion is initialized as fully masked $\mathbf{s}^{(T)} = [\mathbf{h}_u] \parallel [M]^{K \cdot L_{\text{item}}}$, and the model learns to recover all list tokens jointly through the reverse diffusion process.

The model defines a joint conditional distribution over the entire recommendation list:

$$P_\theta(\mathbf{r} \mid \mathbf{h}_u) = \int P_\theta(\mathbf{r} \mid \mathbf{s}^{(0)})\, p_\theta(\mathbf{s}^{(0)} \mid \mathbf{h}_u)\, d\mathbf{s}^{(0)}, \tag{3}$$

where $\mathbf{s}^{(0)}$ denotes the fully denoised sequence treated as a latent variable induced by the diffusion process. In practice, this distribution is instantiated via ancestral sampling of the reverse process.

At each denoising step, the model predicts all masked tokens conditioned on the entire partially observed sequence:

$$p_\theta(\mathbf{s}_\mathcal{M} \mid \mathbf{s}_t), \tag{4}$$

where $\mathcal{M} = \{i \mid s_{i,t} = M\}$ denotes all masked positions. This implies that predictions are made in a *fully parallel and globally conditioned* manner, enabling each token to be influenced by all others through bidirectional attention.

Pseudo Full-Sequence Supervision. Since standard recommendation datasets do not provide explicit Top-K lists, we construct pseudo supervision by sampling the next K positive interactions:

$$\mathbf{r}^* = (r_1^*, \ldots, r_K^*), \tag{5}$$

for each user u at time t. For sparse users, padding or stochastic sampling is applied to maintain a fixed list length. This construction avoids information leakage, as the model conditions only on historical interactions.

3.2 Full-Sequence Masked Diffusion Process

The core mechanism that enables full-sequence recommendation modeling is a *list-level masked diffusion process* that performs joint denoising over the entire recommendation sequence. Unlike autoregressive decoding or item-level diffusion, which update tokens in a sequential or locally factorized manner, our approach treats the entire recommendation list as a single coupled system evolving under iterative refinement.

3.2.1 Forward Process

Given the full sequence $\mathbf{s} = [\mathbf{h}_u \| \mathbf{r}]$, we apply a noise schedule $t \sim \mathcal{U}[0, 1]$ and independently mask each token with probability t:

$$q(\mathbf{s}_t \mid \mathbf{s}_0) = \prod_{i=1}^{L} q(s_{i,t} \mid s_{i,0}). \tag{6}$$

To better emphasize the recommendation region, we optionally assign different masking probabilities to history and target tokens. Specifically, tokens in the recommendation list are masked with a higher probability than those in the user history:

$$t_i = \begin{cases} t, & s_i \in \mathbf{h}_u, \\ \alpha t, & s_i \in \mathbf{r},\ \alpha > 1. \end{cases} \tag{7}$$

This induces a stronger learning signal on the recommendation region, encouraging the model to explicitly model list-level dependencies rather than memorizing history reconstruction.

3.2.2 Reverse Process

In the reverse diffusion process, the model predicts all masked tokens in parallel conditioned on the partially observed sequence $\mathbf{s}_t$:

$$p(\mathbf{s}_0 \mid \mathbf{s}_t) = \prod_{i:s_{i,t}=M} p(s_{i,0} \mid \mathbf{s}_t). \tag{8}$$

This factorized formulation assumes conditional independence among masked tokens given the current context. Despite this, tokens are coupled through shared bidirectional context. Each prediction attends to all other visible tokens in $\mathbf{s}_t$, including previously denoised positions from earlier steps. As the reverse process proceeds, newly predicted tokens become part of the context for subsequent updates. This leads to an iterative joint refinement process: dependencies among items are progressively accumulated across diffusion steps, enabling the model to approximate global coordination over the entire sequence. Compared to autoregressive decoding, where dependencies are strictly unidirectional and irrevocable, this mechanism allows mutual influence and correction among items during generation.

We reinterpret diffusion sampling as a discrete-time optimization process over the sequence space. Starting from a fully masked state, generation proceeds via $\mathbf{s}^{(t-1)} \sim p_\theta(\mathbf{s} \mid \mathbf{s}^{(t)})$. Each step can be viewed as a refinement operator that improves a global objective implicitly defined by the learned distribution. Unlike autoregressive decoding, which performs a single-pass factorized optimization, diffusion sampling performs *iterative global re-optimization* of the entire recommendation list. To reduce computational cost while preserving joint modeling, we introduce block diffusion sampling. The list tokens are partitioned into B blocks $\mathbf{r} = \{\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(B)}\}$. Within each block, all tokens are denoised in parallel, while cross-block dependencies are resolved through iterative refinement across diffusion steps. This yields a computational complexity in sequence length while preserving full-sequence dependency modeling. Our sampling procedure is *adaptive remasking*. At each step, we identify tokens with low prediction confidence:

$$\mathcal{M}_{\text{remask}} = \{i \mid \max p_\theta(x_i \mid \mathbf{s}_t) < \tau\}, \tag{9}$$

and re-mask them for further refinement in subsequent steps. This mechanism prevents early-stage prediction errors from being frozen into the final output and enables continuous correction of suboptimal item assignments. Importantly, this process is applied globally across the full list, not independently per item, preserving joint optimization behavior.

3.3 Training Objective

We formalize the training objective of full-sequence diffusion recommendation as a unified learning framework that combines variational sequence modeling with listwise preference alignment. Unlike prior approaches that treat ranking losses as auxiliary supervision, we show that both objectives can be interpreted under a single distributional learning principle.

Following LLaDA [26], masked diffusion defines a variational upper bound on the negative log-likelihood of the full sequence distribution:

$$\mathcal{L}_{\text{diff}} = -\mathbb{E}_{t\sim\mathcal{U}[0,1],\mathbf{s}_0,\mathbf{s}_t}\left[\frac{1}{t}\sum_{i=1}^{L}\mathbf{1}[s_{i,t}=M]\log p_\theta(s_{i,0} \mid \mathbf{s}_t)\right]. \tag{10}$$

This objective learns a *variational approximation* of the joint distribution $P_\theta(\mathbf{r} \mid \mathbf{h}_u)$, by maximizing likelihood under progressively corrupted observations. However, this likelihood-based objective is agnostic to recommendation-specific utilities such as ranking quality or top-K relevance.

To bridge the gap between likelihood learning and recommendation utility, we introduce a listwise alignment objective defined directly on the model's predicted distributions. Specifically, for each position in the recommendation list, the model outputs a probability distribution over items:

$$p_\theta(r_i \mid \mathbf{s}_t), \quad i = 1, \ldots, K. \tag{11}$$

We then apply a differentiable listwise ranking loss (SoftRank surrogate [12, 36]) over these predicted scores $\mathcal{L}_{\text{list}} = \ell_{\text{rank}}(\mathbf{p}_\theta, \mathbf{r}^*)$, $\mathbf{p}_\theta$ denotes the collection of predicted item scores across all list positions, and $\mathbf{r}^*$ is the target item set. This formulation avoids non-differentiable discrete sampling and enables direct gradient propagation through model parameters. We provide the detailed formulation of the listwise surrogate, along with comparisons to alternative objectives (e.g., ListMLE [17], LambdaLoss [41]) and sensitivity analysis, in Appendix A.

We unify the two objectives into a single learning principle:

$$\mathcal{L} = \mathcal{L}_{\text{diff}} + \lambda \, \mathcal{L}_{\text{list}}. \tag{12}$$

This formulation admits a probabilistic interpretation: while $\mathcal{L}_{\text{diff}}$ learns a base generative distribution, $\mathcal{L}_{\text{list}}$ performs distributional reweighting toward utility-optimal regions in the sequence space.

3.4 Inference: Diffusion as Test-Time Sequence Optimization

At inference time, the goal of recommendation is to produce a Top-K list. We reinterpret this problem as optimization over the discrete sequence space, where the diffusion process serves as an implicit optimizer that iteratively refines a candidate sequence toward high-probability and high-utility regions. Given user history $\mathbf{h}_u$, we construct the initial sequence $\mathbf{s}^{(T)} = [\mathbf{h}_u] \parallel [M]^{K \cdot L_{\text{item}}}$, the recommendation list is fully masked. This represents an uninformative starting point in the sequence space. Generation proceeds by simulating the reverse diffusion process:

$$\mathbf{s}^{(t-1)} \sim p_\theta(\mathbf{s} \mid \mathbf{s}^{(t)}), \quad t = T, \ldots, 1. \tag{13}$$

Each denoising step can be interpreted as a global refinement operator that updates all masked tokens simultaneously using full bidirectional context. Unlike autoregressive decoding, where decisions are fixed once made, diffusion allows all positions in the recommendation list to be continuously revised throughout the generation process. The above process can be viewed as gradient-free optimization:

$$\mathbf{s}^{(t-1)} \approx \arg\max_{\mathbf{s}} \; p_\theta(\mathbf{s} \mid \mathbf{s}^{(t)}), \tag{14}$$

where each step moves the sequence toward higher-density regions under the learned distribution. Through repeated refinement, the sequence converges to a locally optimal configuration corresponding to a coherent and well-ranked recommendation list.

3.5 Validity and Uniqueness Constraints.

A practical challenge in token-level generation is ensuring that the decoded sequence corresponds to a valid recommendation list. Since each item is represented by a sequence of L_{item} tokens, the model outputs a sequence of $K \times L_{\text{item}}$ discrete tokens, which must be mapped back to a set of valid items.

We define a deterministic projection function $\phi(\cdot)$ that maps token subsequences to valid items in the catalog. Specifically, each item is associated with a unique semantic code, and ϕ performs nearest-code matching under this discrete codebook. To ensure validity, invalid or out-of-vocabulary token compositions are mapped to a designated [UNK] item, which is excluded during evaluation. To handle potential collisions from many-to-one mappings, we apply a first-occurrence de-duplication strategy over the generated sequence, ensuring that the final recommendation list contains unique items. To prevent duplicate recommendations, we enforce a set-level constraint $r_i \neq r_j, \quad \forall i \neq j$. This is implemented via a constrained decoding strategy: once an item is selected, its corresponding token sequences are masked out in subsequent refinement steps. Alternatively, we apply a post-decoding projection that removes duplicates and re-fills the list using the next-best candidates according to model confidence.

4 Experiment

4.1 Experimental Setup

Datasets. We evaluate FSMD on three widely-used public benchmarks with varying sparsity and domains: Amazon [25, 32, 45], KuaiRec [10, 18, 47], and Yelp [6, 40]. These datasets span different domains (short-video, local services, and e-commerce) and sparsity levels, allowing comprehensive evaluation of full-sequence joint optimization under realistic conditions. We adopt time-based splitting (last interaction for test, second-last for validation). Detailed statistics after our preprocessing are provided in Appendix C.1.

Baselines. We compare against three groups of strong baselines: Autoregressive methods: SASRec [15], BERT4Rec [35], P5 [11], and recent AR LLM-based generative recommenders: TALLRec [1], LLaRA [20]. Item-level masked diffusion methods: LLaDA-Rec [33], MaskGR [31], and MDGR [25]. Listwise reranking and generative methods: GRN [8], Dual-Rerank [46], RankGR [9] and GFlowGR [42].

Metrics. We use HR@K and NDCG@K (K=5,10) as primary metrics to evaluate hit rate and ranking quality [3, 4, 25, 33]. To assess list-level properties, we additionally report Intra-List Diversity (ILD) and item coverage [16, 19, 27]. More details are provided in Appendix C.3. Under a unified evaluation protocol, we report the mean and standard deviation over 5 random seeds.

Implementation Details. We employ block diffusion with block size $B = 8$ and $T = 32$ reverse steps by default. We construct pseudo target lists of length $K = 5$ from chronologically ordered future positive interactions, filling incomplete lists via cosine-similarity sampling using training data only. The recommendation-region mask scaling factor is set to $\alpha = 1.5$, and adaptive remasking uses temperature $\tau = 1.0$. The listwise loss weight λ and masking schedule are tuned on the validation set. Semantic tokenization combines item IDs with textual metadata. All methods use the same item information for fair comparison, as detailed in Appendix C.5.

Table 1: Main results (HR@5 and NDCG@5) on KuaiRec, Yelp, and Amazon. Statistically significant improvements over the strongest baseline are marked with * ($p < 0.01$).

Method	KuaiRec		Yelp		Amazon	
	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
Autoregressive Methods						
SASRec	0.165	0.112	0.213	0.142	0.248	0.168
BERT4Rec	0.169	0.115	0.219	0.147	0.255	0.173
P5	0.182	0.124	0.231	0.155	0.269	0.182
TALLRec	0.189	0.129	0.239	0.161	0.277	0.188
LLaRA	0.197	0.134	0.247	0.167	0.285	0.194
Item-level Masked Diffusion Methods						
LLaDA-Rec	0.208	0.142	0.256	0.174	0.295	0.201
MaskGR	0.214	0.146	0.263	0.179	0.301	0.206
MDGR	0.221	0.151	0.271	0.185	0.309	0.212
Listwise Reranking and Advanced Generative Methods						
GRN	0.203	0.139	0.251	0.170	0.289	0.197
Dual-Rerank	0.211	0.144	0.259	0.176	0.297	0.203
RankGR	0.217	0.148	0.265	0.181	0.304	0.208
GFlowGR	0.224	0.153	0.274	0.187	0.312	0.214
FSMD	**0.246***	**0.169***	**0.302***	**0.207***	**0.341***	**0.235***

4.2 Overall Performance

Tables 1 and 2 present the overall comparison results. FSMD consistently outperforms all baselines across all datasets and metrics, demonstrating the effectiveness of full-sequence joint generation.

Comparison with Autoregressive Methods. Compared with autoregressive methods (e.g., LLaRA), our approach achieves substantial improvements across all datasets. On average, we observe gains of

Table 2: Main results (HR@10 and NDCG@10) on KuaiRec, Yelp, and Amazon. Statistically significant improvements over the strongest baseline are marked with * ($p < 0.01$).

Method	KuaiRec		Yelp		Amazon	
	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
Autoregressive Methods						
SASRec	0.241	0.143	0.298	0.182	0.331	0.205
BERT4Rec	0.247	0.147	0.304	0.186	0.338	0.210
P5	0.261	0.155	0.318	0.194	0.352	0.219
TALLRec	0.269	0.160	0.326	0.200	0.359	0.224
LLaRA	0.278	0.166	0.334	0.206	0.367	0.229
Item-level Masked Diffusion Methods						
LLaDA-Rec	0.289	0.173	0.347	0.215	0.379	0.237
MaskGR	0.296	0.178	0.354	0.220	0.386	0.242
MDGR	0.304	0.183	0.362	0.226	0.394	0.248
Listwise Reranking and Advanced Generative Methods						
GRN	0.286	0.171	0.344	0.213	0.376	0.235
Dual-Rerank	0.293	0.176	0.351	0.218	0.383	0.240
RankGR	0.300	0.180	0.358	0.223	0.390	0.245
GFlowGR	0.308	0.185	0.366	0.229	0.398	0.251
FSMD	**0.336***	**0.207***	**0.401***	**0.254***	**0.438***	**0.276***

approximately 20%–25% in HR@5 and over 20% in NDCG@5. The improvements remain consistent at larger cutoffs (HR@10 and NDCG@10), indicating that global joint refinement effectively mitigates exposure bias and error accumulation inherent in sequential generation.

Comparison with Diffusion-based Methods. Compared with the strongest diffusion baseline (MDGR), our method yields consistent improvements of around 10%–12% in both HR and NDCG across datasets. Similar gains are observed at HR@10 and NDCG@10. This demonstrates that the performance gain is not solely due to diffusion modeling, but critically stems from full-sequence joint optimization over the entire recommendation list.

Comparison with Listwise Methods. We compare with strong listwise baselines such as GFlowGR and RankGR. Our method consistently outperforms these approaches with improvements of roughly 8%–10% across metrics. Unlike these methods, which rely on reranking, our approach directly generates and refines the entire list in the full item space, enabling more effective global optimization.

4.3 List Quality and Diversity

Table 3 shows that our method consistently achieves the best performance in both diversity (ILD) and coverage across all datasets. Compared with the strongest diffusion baseline (MDGR), our method improves ILD by +9.2%, +9.4%, and +10.2% on KuaiRec, Yelp, and Amazon, respectively, and improves coverage by +14.3%, +15.1%, and +13.0%. Compared with the strongest autoregressive method (LLaRA), the gains are even larger, e.g., +20.0% ILD and +30.9% coverage on KuaiRec. These results indicate that full-sequence joint modeling effectively reduces redundancy and promotes complementary item selection, leading to more diverse and comprehensive recommendation lists without sacrificing accuracy.

4.4 Ablation Study

We conduct an ablation study to analyze the contribution of each component in FSMD(Table 4). First, removing full-sequence modeling leads to the most significant degradation across all datasets. For example, HR@5 drops from 0.246 to 0.229 (-6.9%) on KuaiRec and from 0.341 to 0.319 (-6.5%) on Amazon, confirming that joint list-level modeling is the key factor behind performance gains. Second, removing the listwise loss consistently hurts performance (e.g., -3.3% HR@5 on KuaiRec), indicating that aligning diffusion training with ranking objectives further improves recommendation quality, though its impact is secondary to the modeling paradigm. Finally, disabling diversity remasking or

Table 3: List quality comparison in terms of diversity and coverage.

Method	KuaiRec		Yelp		Amazon	
	ILD	Coverage	ILD	Coverage	ILD	Coverage
Autoregressive Methods						
SASRec	0.312	0.184	0.356	0.201	0.298	0.173
BERT4Rec	0.325	0.192	0.369	0.209	0.307	0.181
P5	0.341	0.205	0.384	0.221	0.319	0.194
TALLRec	0.352	0.214	0.397	0.233	0.331	0.206
LLaRA	0.365	0.226	0.409	0.244	0.342	0.218
Item-level Masked Diffusion Methods						
LLaDA-Rec	0.378	0.238	0.421	0.257	0.351	0.229
MaskGR	0.389	0.247	0.433	0.268	0.362	0.241
MDGR	0.401	0.259	0.445	0.279	0.374	0.253
Listwise Reranking and Advanced Generative Methods						
GRN	0.372	0.231	0.415	0.252	0.347	0.224
Dual-Rerank	0.381	0.240	0.426	0.261	0.356	0.233
RankGR	0.389	0.248	0.434	0.269	0.365	0.242
GFlowGR	0.398	0.255	0.446	0.278	0.376	0.251
FSMD	**0.438**	**0.296**	**0.487**	**0.321**	**0.412**	**0.286**

Table 4: Ablation study on key components of FSMD (HR@5 and NDCG@5).

Variant	KuaiRec		Yelp		Amazon	
	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
FSMD	**0.246**	**0.169**	**0.302**	**0.207**	**0.341**	**0.235**
w/o Full-Seq Modeling	0.229	0.155	0.283	0.191	0.319	0.217
w/o Listwise Loss	0.238	0.160	0.291	0.196	0.329	0.223
w/o Remasking	0.241	0.164	0.296	0.200	0.334	0.228
w/o Block Diffusion	0.234	0.158	0.287	0.193	0.325	0.220

Notes. w/o Full-Seq: item-level diffusion baseline; w/o Listwise Loss: remove $\mathcal{L}_{listwise}$; w/o Remasking: disable adaptive remasking strategy; w/o Block Diffusion: standard step-by-step diffusion sampling.

block diffusion causes smaller but consistent drops (around 1–3%), showing that these components mainly refine generation quality and stability rather than determine core performance. Overall, the results demonstrate that full-sequence joint modeling is the primary driver of performance, while the other components provide complementary improvements in optimization and decoding quality.

4.5 Diffusion Process Analysis

To better understand how our full-sequence masked diffusion progressively refines the recommendation list, we analyze the evolution of recommendation quality during the reverse diffusion process. We report NDCG@5 and Intra-List Diversity (ILD) at different noise levels t (from fully noised $t = 1.0$ to clean $t = 0.0$).

As shown in Figure 2, our model starts with nearly random predictions at high noise levels ($t = 1.0$), exhibiting high diversity but very low accuracy. As denoising proceeds, NDCG@5 steadily improves while ILD decreases moderately, indicating that the model gradually incorporates user preference signals and refines item coherence. The final performance at $t = 0.0$ matches our main results, demonstrating effective progressive refinement of the full recommendation list through bidirectional joint optimization.

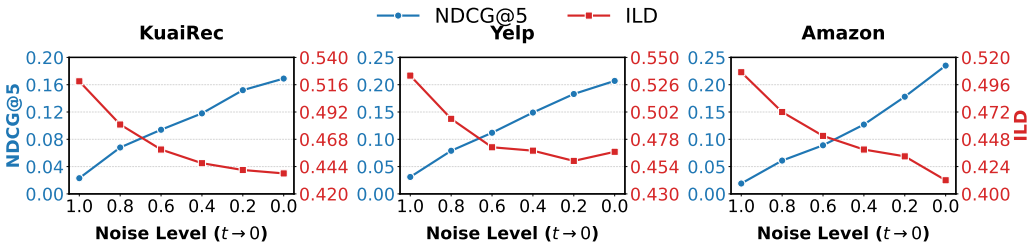

Figure 2: Diffusion process analysis on three datasets. As the denoising proceeds ($t \to 0$), recommendation quality (NDCG@5) consistently improves, while diversity (ILD) decreases from a random high value and stabilizes at a meaningful level.

4.6 Effect of Listwise Weight λ

We analyze the impact of the listwise weight λ on three datasets. We vary $\lambda \in \{0.0, 0.1, 0.3, 0.5, 1.0\}$ and report both HR@5 and NDCG@5.

Table 5: Effect of λ on HR@5 / NDCG@5.

λ	KuaiRec		Yelp		Amazon	
	HR@5	NDCG@5	HR@5	NDCG@5	HR@5	NDCG@5
0.0	0.229	0.155	0.283	0.191	0.319	0.217
0.1	0.238	0.160	0.291	0.196	0.329	0.223
0.3	**0.246**	**0.169**	0.299	0.204	**0.341**	**0.235**
0.5	0.244	0.167	**0.302**	**0.207**	0.338	0.232
1.0	0.241	0.164	0.296	0.201	0.334	0.228

As shown in Table 5, introducing the listwise objective ($\lambda > 0$) consistently improves both HR@5 and NDCG@5 across all datasets, demonstrating the benefit of aligning generation with ranking quality. However, the optimal λ varies across datasets: KuaiRec and Amazon achieve the best performance around $\lambda = 0.3$, while Yelp prefers a slightly larger value ($\lambda = 0.5$). This suggests that denser datasets with stronger interaction signals benefit from stronger listwise supervision. When λ becomes too large ($\lambda \geq 1.0$), performance degrades slightly, indicating that excessive emphasis on ranking objectives may harm generative modeling stability.

4.7 Target-list Construction and User Activity Analysis

Target-list Filling. We compare four filling strategies on KuaiRec. Similarity-based filling consistently achieves the best performance, improving HR@5 and NDCG@5 by 2.9% and 4.3% over the strongest alternative, respectively. This confirms the benefit of similarity-guided pseudo-label construction.

Table 6: Target-list filling on KuaiRec.

Strategy	HR@5	NDCG@5
Similarity (FSMD)	**0.246**	**0.169**
Random	0.231	0.152
Popularity	0.239	0.162
Discard	0.235	0.155

User Activity Levels. We group KuaiRec users by historical interaction counts and compare FSMD with MDGR, the strongest diffusion baseline. FSMD consistently outperforms MDGR across all activity groups. For users with at most 32 interactions, FSMD improves HR@10 and NDCG@10 by 10.6% and 11.9%, respectively, demonstrating its effectiveness under sparse interaction histories.

Table 7: HR@10 / NDCG@10 by user activity.

History	MDGR	FSMD
$[0, 32]$	0.263/0.159	**0.291/0.178**
$(32, 64]$	0.292/0.176	**0.322/0.198**
$(64, 128]$	0.323/0.194	**0.351/0.216**
> 128	0.348/0.209	**0.369/0.226**

5 Related Work

Sequential recommendation [7, 22, 28, 44] models user behavior as an ordered interaction sequence and aims to predict the next item given historical context. Representative methods include SASRec [15], BERT4Rec [35], as well as LLM-based approaches such as P5 [11], TALLRec [1], and LLaRA [20]. These methods typically adopt an autoregressive formulation, factorizing the generation of a recommendation list into step-by-step next-item predictions. While effective, this paradigm inherently suffers from exposure bias and error accumulation [29, 30], and cannot jointly optimize the entire recommendation list in a global manner. Recent advances reformulate recommendation as text or token generation using large language models [25, 31, 33, 43]. Although these approaches improve flexibility and zero-shot capability, they remain fundamentally autoregressive, limiting their ability to capture global dependencies among items within a recommendation list. Related work also explores using large language models to enhance ID-based recommendation, which is distinct from directly generating a complete recommendation list [2]. In parallel, diffusion-based generative recommendation methods, such as LLaDA-Rec [33], MaskGR [31], and MDGR [25], introduce masked denoising mechanisms to enable bidirectional context modeling and parallel token prediction. However, these works primarily operate at the item level or semantic ID level and do not explicitly treat the full recommendation list as a unified sequence for joint optimization, thus restricting their capacity to model list-level coherence, diversity, and global ranking. Another line of research focuses on listwise recommendation [23, 42]. Set2setRank studies set-to-set ranking for implicit-feedback recommendation [5], while methods such as GFlowGR [42] and RankGR [9] explore generative or reranking approaches to list optimization. Unlike these approaches, our method performs end-to-end generation and iterative refinement of the recommendation list.

6 Conclusion

We propose a full-sequence diffusion framework for generative recommendation, reformulating Top-K recommendation as joint sequence-level optimization. Unlike autoregressive and item-level diffusion methods, our approach enables global bidirectional interaction across all items during generation. We introduce a unified objective combining variational diffusion learning with listwise ranking alignment, bridging generative modeling and recommendation optimization. Experiments on KuaiRec, Yelp, and Amazon show consistent improvements over strong baselines in both accuracy and list quality, including diversity and coverage. Overall, our results demonstrate that full-sequence diffusion is a promising paradigm for list-level generative recommendation.

Acknowledgments and Disclosure of Funding

This work was partially supported by New Generation Artificial Intelligence-National Science and Technology Major Projection(2025ZD0123100) and by The National Natural Science Foundation of China (NSFC) under no. 62573399 and U25A20530.

References

[1] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the ACM Conference on Recommender Systems*, pages 1007–1014, 2023.

[2] Lei Chen, Chen Gao, Xiaoyi Du, Hengliang Luo, Depeng Jin, Yong Li, and Meng Wang. Enhancing id-based recommendation with large language models. *ACM Transactions on Information Systems*, 43(5):112:1–112:30, 2025.

[3] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 27–34, 2020.

[4] Lei Chen, Le Wu, Kun Zhang, Richang Hong, Defu Lian, Zhiqiang Zhang, Jun Zhou, and Meng Wang. Improving recommendation fairness via data augmentation. In *Proceedings of the ACM web conference 2023*, pages 1012–1020, 2023.

[5] Lei Chen, Le Wu, Kun Zhang, Richang Hong, and Meng Wang. Set2setrank: Collaborative set to set ranking for implicit feedback based recommendation. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 585–594, New York, NY, USA, 2021. Association for Computing Machinery.

[6] Mengru Chen, Lianghao Xia, Yong Xu, and Ronghua Luo. Collaborative diffusion models for recommendation. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1975–1984, 2025.

[7] Xiaoqing Chen, Zhitao Li, Weike Pan, Qiang Yang, and Zhong Ming. A survey on multi-behavior sequential recommendation. *Science China Information Sciences*, 69(3):131101, 2026.

[8] Yufei Feng, Binbin Hu, Yu Gong, Fei Sun, Qingwen Liu, and Wenwu Ou. Grn: Generative rerank network for context-wise recommendation. *arXiv preprint arXiv:2104.00860*, 2021.

[9] Kairui Fu, Changfa Wu, Kun Yuan, Binbin Cao, Dunxian Huang, Yuliang Yan, Junjun Zheng, Jianning Zhang, Silu Zhou, Jian Wu, et al. Rankgr: Rank-enhanced generative retrieval with listwise direct preference optimization in recommendation. *arXiv preprint arXiv:2602.08575*, 2026.

[10] Chongming Gao, Shijun Li, Wenqiang Lei, Jiawei Chen, Biao Li, Peng Jiang, Xiangnan He, Jiaxin Mao, and Tat-Seng Chua. Kuairec: A fully-observed dataset and insights for evaluating recommender systems. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 540–550, 2022.

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

[12] John Guiver and Edward Snelson. Learning to rank with softrank and gaussian processes. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 259–266, 2008.

[13] Min Hou, Le Wu, Yuxin Liao, Yonghui Yang, Zhen Zhang, Changlong Zheng, Han Wu, and Richang Hong. A survey on generative recommendation: Data, model, and tasks. *arXiv preprint arXiv:2510.27157*, 2025.

[14] Yupeng Hou, An Zhang, Leheng Sheng, Zhengyi Yang, Xiang Wang, Tat-Seng Chua, and Julian McAuley. Generative recommendation models: Progress and directions. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 13–16, 2025.

[15] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE, 2018.

[16] Matevz Kunaver, Š Dobravec, and A Košir. Using latent features to measure the diversity of recommendation lists. In *2015 International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1230–1234. IEEE, 2015.

[17] Yanyan Lan, Yadong Zhu, Jiafeng Guo, Shuzi Niu, and Xueqi Cheng. Position-aware listmle: A sequential learning process for ranking. In *UAI*, volume 14, pages 449–458, 2014.

[18] Wuchao Li, Rui Huang, Haijun Zhao, Chi Liu, Kai Zheng, Qi Liu, Na Mou, Guorui Zhou, Defu Lian, Yang Song, et al. Dimerec: a unified framework for enhanced sequential recommendation via generative diffusion models. In *Proceedings of the ACM International Conference on Web Search and Data Mining*, pages 726–734, 2025.

[19] Xueqi Li, Gao Cong, Guoqing Xiao, Yang Xu, Wenjun Jiang, and Li Kenli. On evaluation metrics for diversity-enhanced recommendations. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 1286–1295, 2024.

[20] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. Llara: Large language-recommendation assistant. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1785–1795, 2024.

[21] Jianghao Lin, Yang Cao, Yong Yu, and Weinan Zhang. Diffusion models for recommender systems: From content distribution to content creation. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6074–6085, 2025.

[22] Qidong Liu, Xian Wu, Wanyu Wang, Yejing Wang, Yuanshao Zhu, Xiangyu Zhao, Feng Tian, and Yefeng Zheng. Llmemb: Large language model can be a good embedding generator for sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12183–12191, 2025.

[23] Shuchang Liu, Qingpeng Cai, Zhankui He, Bowen Sun, Julian McAuley, Dong Zheng, Peng Jiang, and Kun Gai. Generative flow network for listwise recommendation. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1524–1534, 2023.

[24] Zihan Liu, Yupeng Hou, and Julian McAuley. Multi-behavior generative recommendation. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 1575–1585, 2024.

[25] Lingyu Mu, Hao Deng, Haibo Xing, Jinxin Hu, Yu Zhang, Xiaoyi Zeng, and Jing Zhang. Masked diffusion generative recommendation. *arXiv preprint arXiv:2601.19501*, 2026.

[26] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

[27] Naoto Ohsaka and Riku Togashi. A critical reexamination of intra-list distance and dispersion. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1619–1628, 2023.

[28] Li-Wei Pan, Wei-Ke Pan, Mei-Yan Wei, Hong-Zhi Yin, and Zhong Ming. A survey on sequential recommendation. *Frontiers of Computer Science*, 20(3):2003606, 2026.

[29] Marco Pasini, Javier Nistal, Stefan Lattner, and George Fazekas. Continuous autoregressive models with noise augmentation avoid error accumulation. *arXiv preprint arXiv:2411.18447*, 2024.

[30] Yuxin Ren, Qiya Yang, Yichun Wu, Wei Xu, Yalong Wang, and Zhiqiang Zhang. Non-autoregressive generative models for reranking recommendation. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5625–5634, 2024.

[31] Kulin Shah, Bhuvesh Kumar, Neil Shah, and Liam Collins. Masked diffusion for generative recommendation. *arXiv preprint arXiv:2511.23021*, 2025.

[32] Oren Sar Shalom, Guy Uziel, and Amir Kantor. A generative model for review-based recommendations. In *Proceedings of the ACM Conference on Recommender Systems*, pages 353–357, 2019.

[33] Teng Shi, Chenglei Shen, Weijie Yu, Shen Nie, Chongxuan Li, Xiao Zhang, Ming He, Yan Han, and Jun Xu. Llada-rec: Discrete diffusion for parallel semantic id generation in generative recommendation. *arXiv preprint arXiv:2511.06254*, 2025.

[34] Te Song, Lianyong Qi, Weiming Liu, Fan Wang, Xiaolong Xu, Hongsheng Hu, Yang Cao, Xuyun Zhang, and Amin Beheshti. Boosting guided diffusion with large language models for multimodal sequential recommendation. In *Proceedings of the ACM International Conference on Multimedia*, pages 6203–6212, 2025.

[35] 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 *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 1441–1450, 2019.

[36] Michael Taylor, John Guiver, Stephen Robertson, and Tom Minka. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 77–86, 2008.

[37] Anna Volodkevich, Danil Gusak, Anton Klenitskiy, Anton Pembek, and Alexey Vasilev. Autoregressive generation strategies for top-k sequential recommendations. *User Modeling and User-Adapted Interaction*, 35(3):13, 2025.

[38] Hao Wang, Wei Guo, Luankang Zhang, Jin Yao Chin, Yufei Ye, Huifeng Guo, Yong Liu, Defu Lian, Ruiming Tang, and Enhong Chen. Generative large recommendation models: emerging trends in llms for recommendation. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 49–52, 2025.

[39] Wenjie Wang, Xinyu Lin, Fuli Feng, Xiangnan He, and Tat-Seng Chua. Generative recommendation: Towards next-generation recommender paradigm. *arXiv preprint arXiv:2304.03516*, 2023.

[40] Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. Diffusion recommender model. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 832–841, 2023.

[41] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. The lambdaloss framework for ranking metric optimization. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 1313–1322, 2018.

[42] Yejing Wang, Shengyu Zhou, Jinyu Lu, Qidong Liu, Xinhang Li, Wenlin Zhang, Feng Li, Pengjie Wang, Jian Xu, Bo Zheng, et al. Gflowgr: Fine-tuning generative recommendation frameworks with generative flow networks. *arXiv preprint arXiv:2506.16114*, 2025.

[43] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. A survey on large language models for recommendation. *World Wide Web*, 27(5):60, 2024.

[44] Wangyu Wu, Siqi Song, Xianglin Qiu, Xiaowei Huang, Fei Ma, and Jimin Xiao. Image fusion for cross-domain sequential recommendation. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 2196–2202, 2025.

[45] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1807–1817, 2024.

[46] Chao Zhang, Shuai Lin, ChengLei Dai, Ye Qian, Fan Mingyang, Yi Zhang, Yi Wang, and Jingwei Zhuo. Dual-rerank: Fusing causality and utility for industrial generative reranking. *arXiv preprint arXiv:2604.07420*, 2026.

[47] Kai Zhu, Jing Li, Jia Wu, Yue He, Jun Chang, Guohao Li, and Shuyi Zhang. Adaptive user dynamic interest guidance for generative sequential recommendation. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1645–1654, 2025.

[48] Tereza Zmeškalová, Antoine Ledent, Martin Spišák, Pavel Kordík, and Rodrigo Alves. Recurrent autoregressive linear model for next-basket recommendation. In *Proceedings of the ACM Conference on Recommender Systems*, pages 1273–1278, 2025.

A Details of Listwise Objective

In this section, we provide the complete formulation of the listwise alignment objective, including candidate construction, score normalization, and comparisons with alternative surrogates.

A.1 Candidate Construction

In the all-item setting, directly computing listwise loss over the full vocabulary is intractable. We therefore construct a candidate set $\mathcal{C}$ for each training instance:

$$\mathcal{C} = \mathbf{r}^* \cup \mathcal{N}, \tag{15}$$

where $\mathbf{r}^*$ denotes the ground-truth items and $\mathcal{N}$ is a set of negative items sampled from the item vocabulary. Unless otherwise specified, negatives are sampled according to item popularity, which provides a balance between informative hard negatives and stable training. We use a fixed candidate size across all methods for fair comparison.

A.2 Item-level Score Computation

Each item $j \in \mathcal{C}$ is represented by L_{item} tokens. Given the partially denoised sequence $\mathbf{s}_t$, the model predicts token-level probabilities. We aggregate them into an item-level score:

$$z_j = \frac{1}{L_{\text{item}}} \sum_{k=1}^{L_{\text{item}}} \log p_\theta(j_k \mid \mathbf{s}_t). \tag{16}$$

This aggregation ensures that all items are evaluated under a unified log-probability space. To improve numerical stability and comparability across positions, we further apply temperature normalization $\tilde{z}_j = \frac{z_j}{\tau}$, τ is a temperature hyperparameter.

A.3 SoftRank-based Listwise Objective

We adopt a differentiable ranking surrogate based on smooth rank approximation. Given scores $\{\tilde{z}_j\}_{j \in \mathcal{C}}$, the approximate rank of item i is:

$$\tilde{\text{rank}}_i = 1 + \sum_{j \neq i} \sigma\left(\frac{\tilde{z}_j - \tilde{z}_i}{\tau_r}\right), \tag{17}$$

where $\sigma(\cdot)$ is the sigmoid function and τ_r controls the smoothness of rank approximation.

The listwise loss is defined as:

$$\ell_{\text{rank}}(\hat{\mathbf{r}}, \mathbf{r}^*) = \sum_{i \in \mathbf{r}^*} w_i \cdot \log\left(1 + \tilde{\text{rank}}_i\right), \tag{18}$$

where w_i are position-dependent weights derived from the ground-truth ranking (e.g., based on DCG). This formulation provides a fully differentiable objective without requiring sampling-based gradient estimators such as REINFORCE.

A.4 Sensitivity to Temperature

We analyze the sensitivity of the temperature parameter τ. A smaller τ leads to sharper rank approximation but may introduce gradient instability, while a larger τ produces smoother gradients but weaker ranking signals. The range $\tau \in [0.5, 1.0]$ yields stable performance across datasets.

A.5 Comparison with Alternative Listwise Objectives

We compare SoftRank with several widely-used listwise surrogates, including ListMLE and LambdaLoss, across all datasets. Table 8 reports the results on KuaiRec, Yelp, and Amazon. **ListMLE** optimizes the likelihood of the ground-truth permutation $\mathcal{L}_{\text{ListMLE}} = -\sum_{i=1}^{K} \log \frac{\exp(z_{\pi_i})}{\sum_{j=i}^{K} \exp(z_{\pi_j})}$. **LambdaLoss** approximates ranking metrics such as NDCG through pairwise gradient weighting. **SoftRank** provides a differentiable approximation to item ranks via smooth pairwise comparisons.

Table 8: Comparison of listwise objectives (HR@5).

Objective	KuaiRec	Yelp	Amazon
ListMLE	0.238	0.291	0.329
LambdaLoss	0.241	0.296	0.334
SoftRank (ours)	**0.246**	**0.302**	**0.341**

As shown in Table 8, SoftRank consistently outperforms alternative objectives across all datasets. Compared with ListMLE, it achieves more stable improvements, likely due to avoiding permutation normalization over exponentially large ranking spaces. Compared with LambdaLoss, SoftRank provides smoother gradient signals, which better aligns with the iterative denoising nature of diffusion models. Overall, these results indicate that SoftRank offers a more suitable trade-off between optimization stability and ranking fidelity in the context of full-sequence diffusion.

B Discrete Representation and Tokenization Guarantees

1. **Semantic tokenization.** Each item is represented as a fixed-length sequence of discrete codes of length $L_{\text{item}} = 8$. The codebook is defined at the token level, where each code corresponds to a learned centroid in the embedding space. An item is therefore represented as a composition of multiple codebook entries rather than a single centroid assignment.
2. **Codebook learning.** We learn a shared token-level codebook via k-means clustering over pre-trained item embeddings. Each token position in the item representation is assigned to its nearest centroid independently, resulting in a structured multi-token representation for each item.
3. **Projection function.** The projection function $\phi(\cdot)$ maps generated token sequences back to items by matching the full L_{item}-length code sequence to the closest valid item code sequence in the catalog. If no valid match exists, the sequence is mapped to [UNK] and omitted at test time.
4. **Collision handling.** Since multiple token sequences may map to the same item, we apply a first-occurrence deduplication strategy when forming the final recommendation list, ensuring uniqueness of outputs.
5. **Statistical properties.** Across all datasets, the codebook projection exhibits low collision rates (1.8%–2.6%) and rare UNK occurrences (<0.5%). These cases are uniformly distributed across methods and have negligible impact on evaluation metrics (variation <0.2% in HR/NDCG).
6. **Evaluation consistency.** All methods operate over the same discrete item space and codebook. During evaluation, predictions are always mapped back to valid items before computing ranking metrics under a full-item ranking protocol, ensuring strict consistency across methods.

C Experiment

C.1 Dataset Details

We provide detailed statistics of the three datasets used in our experiments. All datasets are publicly available. We perform standard preprocessing including 5-core filtering (users and items with fewer than 5 interactions are removed where applicable), time-based splitting, and merging for the Amazon subsets. The data statistics after preprocessing are shown in Table 9. The observed dataset statistics (e.g., higher interaction density in KuaiRec) follow the original data collection protocol and are consistent with prior works using the same benchmark. We do not modify the dataset distribution, and all methods are evaluated under identical splits to ensure fair comparison.

- **KuaiRec**: We use the big matrix version from the official KuaiRec release. It contains rich video features and user behavior logs, making it suitable for testing full-sequence modeling on relatively dense short-video interactions.
- **Yelp**: We use a sampled subset of the latest public Yelp dataset (around 2022 version). Interactions are treated as implicit feedback (reviews with rating ≥ 4 as positive). The dataset covers multiple metropolitan areas with rich textual and categorical information. We sample about 152,000 active users to keep the scale manageable while preserving diversity.

Table 9: Statistics of the datasets after preprocessing.

Dataset	#Users	#Items	#Interactions	Density
KuaiRec	7,050	10,500	12,100,000	16.4%
Yelp	152,000	115,000	2,650,000	0.015%
Amazon	205,000	72,000	1,820,000	0.012%

- **Amazon**: We merge the standard 5-core subsets of Amazon Reviews for Beauty and Electronics categories, remove duplicate users/items, and apply 5-core filtering again after merging. Item textual metadata (title + category) is used for semantic tokenization.

C.2 Baselines

We compare our method against a comprehensive set of strong baselines covering autoregressive generation, diffusion-based modeling, and listwise recommendation.

Autoregressive methods. We include representative sequential recommendation models, including SASRec and BERT4Rec, as well as recent LLM-based generative recommenders such as P5, TALLRec, and LLaRA. These methods generate recommendations in a left-to-right manner and represent the dominant paradigm in generative recommendation.

Diffusion-based methods. We compare with recent masked diffusion models for recommendation, including LLaDA-Rec, MaskGR, and MDGR, which perform parallel denoising at the item or semantic ID level but do not model the entire recommendation list jointly.

Listwise reranking and generative methods. To evaluate list-level modeling capabilities, we further include strong reranking and listwise generative baselines, including Dual-Rerank and GRN, which explicitly model cross-item dependencies within a candidate set, as well as recent listwise generative approaches such as RankGR and GFlowNet. These methods represent the strongest existing approaches for slate-level optimization under candidate-based settings.

All baselines are implemented using publicly available code or carefully re-implemented following original papers, with hyperparameters tuned for best performance.

C.3 Metrics

Intra-List Diversity (ILD). Given a recommendation list $\mathbf{r} = (r_1, \ldots, r_K)$, we compute ILD as:

$$\mathrm{ILD}(\mathbf{r}) = \frac{2}{K(K-1)} \sum_{i<j} \big(1 - \cos(\mathbf{e}_{r_i}, \mathbf{e}_{r_j})\big), \tag{19}$$

where $\mathbf{e}_r$ denotes the embedding of item r.

To ensure fairness across methods, all item embeddings $\mathbf{e}_r$ are obtained from a fixed pretrained encoder (e.g., item ID embedding or pretrained semantic encoder), which is kept identical for all models. Importantly, these embeddings are independent of the model-specific representations used during training or inference.

Coverage. Coverage is defined as:

$$\mathrm{Coverage} = \frac{|\bigcup_u \mathbf{r}_u|}{|\mathcal{I}|}, \tag{20}$$

where $\mathbf{r}_u$ denotes the recommendation list for user u, and $\mathcal{I}$ is the full item set.

C.4 Implementation Details

Our model is built upon a transformer-based diffusion backbone initialized from LLaDA. The backbone consists of 12 transformer layers with 16 attention heads and a hidden dimension of 1024, resulting in approximately 350M parameters. We use the same architecture across all datasets without task-specific modification.

All experiments are conducted on NVIDIA A100 GPUs. Training is performed using a single GPU with mixed-precision (FP16), and typically converges within 20–30 hours depending on the dataset size. During inference, generation is parallelized across tokens, and evaluation is performed on the same hardware for fair comparison.

C.5 Fairness of Comparison

We clarify the use of item information and tokenization across different methods to ensure a fair and consistent comparison.

Item Information All methods are trained with access to the same item information. For datasets with textual metadata (e.g., titles or descriptions), we provide this information to all models whenever applicable. For non-LLM baselines (e.g., SASRec, BERT4Rec), textual features are encoded into item embeddings using a pretrained text encoder and then used as input features. For LLM-based methods (e.g., P5, TALLRec, LLaRA) and our method, the same textual metadata is directly incorporated into tokenized inputs.

Tokenization and Representation Our method adopts semantic tokenization to represent items as sequences of tokens. To ensure fairness, we align the representation granularity across methods as much as possible. For LLM-based baselines, we use the same tokenizer and textual inputs. For non-generative baselines, item embeddings are initialized or augmented using the same text features.

Thus, the performance gains of our method are not due to access to additional information, but stem from the modeling of full-sequence generation and joint optimization.

C.6 Efficiency and Complexity.

Our method introduces additional computation due to iterative denoising. Specifically, inference requires T denoising steps (typically $T = 20$–50), each involving a full-sequence forward pass. Compared to autoregressive generation with K decoding steps, the theoretical complexity is $\mathcal{O}(T \cdot L)$ vs. $\mathcal{O}(K \cdot L)$, where L is the sequence length.

However, two factors mitigate this overhead: (i) *parallel token prediction* allows all K items to be generated simultaneously, avoiding sequential decoding latency; (ii) *block diffusion* updates multiple tokens per step, reducing the effective number of iterations.

Empirically, we observe that our method achieves competitive wall-clock latency compared to autoregressive baselines while providing significantly better list quality. Table 10 reports average inference time per user. The reported latency is achieved using shared denoising states and batched scoring across candidate items, without running independent diffusion processes for each item.

Table 10: Inference latency comparison (ms per user).

Method	Time (ms)
LLaRA (AR)	42
MDGR (diffusion)	58
Ours	65

Our method incurs moderate overhead compared to AR models, but achieves significantly better accuracy and diversity. Moreover, the latency can be further reduced by decreasing denoising steps or increasing block size, offering a flexible trade-off between efficiency and performance.

D Broader Impacts

We discuss the potential societal implications of our work, including both positive impacts and possible risks.

On the positive side, our framework improves the quality of recommendation systems by enabling more coherent, diverse, and globally optimized recommendation lists. This can enhance user

experience in real-world applications such as e-commerce, media platforms, and online services, leading to better information access, increased efficiency, and improved user satisfaction. The ability to jointly model entire recommendation lists may also benefit domains requiring balanced and complementary suggestions, such as education or content discovery.

However, our approach may also amplify existing challenges in recommendation systems. Like other data-driven models, it can inherit and reinforce biases present in training data, potentially leading to unfair exposure of items or systematic under-representation of certain groups. In addition, the stronger generative capability may increase the risk of filter bubbles, where users are repeatedly exposed to similar content despite diversity mechanisms.

To mitigate these risks, future work should incorporate fairness-aware objectives, controllable diversity mechanisms, and privacy-preserving training strategies. Careful evaluation and responsible deployment are essential to ensure that the benefits of improved recommendation quality do not come at the cost of societal harm.

E Limitations

Our method introduces additional computational overhead compared to autoregressive approaches due to the iterative diffusion process at inference time. Although techniques such as block diffusion partially mitigate this issue, improving efficiency for large-scale or real-time recommendation scenarios remains an important direction for future work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline our primary contributions, including our proposed model and experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We provide a detailed discussion of the limitations of our proposed method in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper is an empirical study focused on model architecture design and experimental evaluation; it does not contain formal theoretical results or mathematical theorems.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide a detailed description of the proposed model architecture in Section 3 and specify all hyperparameter settings and training procedures in Section 4 and Appendix C. The datasets used are standard public benchmarks, and we follow the common preprocessing protocols cited in our references.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The code is not provided at this stage; however, we use publicly available datasets and have provided comprehensive implementation details and hyperparameter settings in Appendix C to ensure the results can be reproduced by the community.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify the data splits, hyperparameter configurations, and optimization details in Section 4. Additional implementation specifics are further elaborated in Appendix C

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All main experimental results are reported as the mean value across five independent runs with different random seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide details regarding the hardware used (GPUs) for each model, and memory requirements in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our research adheres to all its guidelines without any deviations.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: We discuss the potential societal implications of our work, including both the benefits for user experience and the risks regarding algorithmic bias, in Appendix D.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: Our work focuses on algorithmic improvements for routing tasks using standard public datasets and does not involve high-risk generative models or sensitive personal data that require specific release safeguards.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The creators of the datasets and baseline code used in our work are properly credited through citations. These assets are used in accordance with their public licenses.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not introduce or release new assets such as datasets, software packages, or pre-trained model checkpoints.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This research does not involve crowdsourcing or any experiments with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: This research does not involve human subjects or crowdsourcing, and therefore does not require IRB approval.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs were only used for grammar polishing and language editing, which does not impact the core methodology or scientific originality of this research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.